An efficient extended block Arnoldi algorithm for feedback stabilization of incompressible Navier-Stokes flow problems

M.A. Hamadi, A. Ratnani, Khalide Jbilou

An efficient extended block Arnoldi algorithm for feedback stabilization of incompressible Navier-Stokes flow problems

M.A. Hamadi[a,b], K. Jbilou[a,b], A. Ratnani[a]

[a] *Mohammed VI Polytechnic University, laboratory MSDA, Lot 660, Hay Moulay Rachid, Ben Guerir, 43150 Maroc*
[b] *University of Littoral Côte d'Opale, laboratory LMPA, 50 Rue F. Buisson, BP 699–62228 Calais cedex, France*

Abstract

Navier-Stokes equations are well known in modelling of an incompressible Newtonian fluid, such as air or water. This system of equations is very complex due to the non-linearity term that characterizes it. After the linearization and the discretization parts, we get a descriptor system of index-2 described by a set of differential algebraic equations (DAEs). The two main parts we develop through this paper are focused firstly on constructing an efficient algorithm based on a projection technique onto an extended block Krylov subspace, that appropriately allows us to construct a reduced system of the original DAE system. Secondly, we solve a Linear Quadratic Regulator (LQR) problem based on a Riccati feedback approach. This approach uses numerical solutions of large-scale algebraic Riccati equations. To this end, we use the extended Krylov subspace method that allows us to project the initial large matrix problem onto a low order one that is solved by some direct methods. These numerical solutions are used to obtain a feedback matrix that will be used to stabilize the original system. We conclude by providing some numerical results to confirm the performances of our proposed method compared to other known methods.

Keywords: Algebraic Riccati equations, Feedback Matrix, Krylov subspaces, Linear Quadratic Regulator(LQR), Navier-Stokes equations.

1. Introduction

Navier-Stokes equations (NSEs) are very important in the physics of fluid mechanics. The existence and smoothness of solutions is not yet guaranteed, although these equations are still of interest to engineers and scientists in many technical fields. One of the main reason that makes the solution of NSEs not unique is the chaotically appearing turbulences due to a naturally

Email addresses: amine.hamadi@um6p.ma (M.A. Hamadi), jbilou@univ-littoral.fr (K. Jbilou), ahmed.ratnani@um6p.ma (A. Ratnani)

existing instabilities. In fact, these turbulences cannot be computed or predicted either, that is why we need to seek for stabilization techniques. The stabilization of incompressible flow problems described by Navier-Stokes equations is at the heart of a wide range of engineering applications, since they require a stable and controlled velocity field, which is considered to be the basis for ongoing reaction or production processes. A bench of work based on the theoretical setting has been established by several authors for the stabilization of two and three-dimensional Navier-Stokes equations using a feedback control; see M. Badra [1, 2], V. Barbu et al.,[3, 4], A.V. Fursikov [5] and J. P. Raymond et al., [6, 7, 8]. Other works have been performed for the stabilization of two-dimensional Navier-Stokes equations based on a numerical setting by solving large-scale Linear Quadratic Regulator (LQR) problem using a Riccati-feedback approach; see [9, 10]. In [9] Bansch et al., proposed a generalized low-rank Cholesky factor Newton method to stabilize a flow around a cylinder. The LQR approach that interests us is based on a finite dimensional matrix derived from the discretization of the linearized Navier-Stokes equations around a steady state. After the discretization stage we get a descriptor index-2 system of differential algebraic equations (DAEs) of a high dimension. Another way to deal with this stabilization problem is to choose an appropriate method that allows us to construct a reduced system to the one described by a set of DAEs and then we stabilize the reduced system instead of the original one. This approach is convenient since it is based on the treatment of lower dimensional systems that makes the computation feasible. In [11] the authors use a balanced truncation method to construct an efficient reduced system and they solve the obtained LQR problem associated the reduced system based on a Riccati feedback approach.

Two main parts will be covered in this paper. The first one focuses on describing an efficient method to reduce a large-scale descriptor index-2 system of differential algebraic equations, depicted from a spatial discretization of the linearized Navier-Stokes equations around a steady state. This method is based on a projection technique onto an extended block Krylov subspace, and it allows us to construct a reduced system that has nearly the same response characteristics. A bench of work has been done to build an effective reduced model, such as projection techniques onto suitable Krylov-based subspaces as the rational or extended-rational block Krylov subspaces, see [12, 13, 14, 15, 16, 17, 18]. Another class of methods described in [19, 20, 21] contains balanced truncation methods. Numerous model reduction methods have been explored for Navier-Stokes equations using balanced truncation and proper orthogonal decomposition [22, 11]. A balanced truncation model reduction method for the Ossen equations has been investigated in [23]. For large problems, Krylov subspace methods are more efficient in term of cpu-time and memory requirements which is not the case for the methods based on balanced truncation since they require solving two large Lyapunov matrix equations at each iteration of

the process and also the computation of singular value decompositions. All these methods that we mentioned here work properly for a class of descriptor dynamical systems represented by a set of ordinary differential equations (ODEs). Unfortunately, this is not our case since the dynamical system that we are dealing with is represented by a set of DAEs and therefore these methods are not directly applicable. Hence, one needs a process that ensures a transformation of DAEs into ODEs in an appropriate manner. This will result in a dense projector called the Leray projection and to overcome this problem, we give a simplification on how to avoid this dense projection matrix while performing our process to get a reduced system. The second part of the paper is devoted to solving a derived Linear Quadratic Regulator (LQR) problem using a Riccati feedback approach. The major issue that we have to deal with is to solve a large-scale algebraic Riccati equation [24, 25, 26], which is the key to design a controller represented by a feedback matrix. Our aim is to stabilize the unstable system by using the constructed feedback matrix. We propose an extended block Arnoldi algorithm with appropriate computational requirements. The LQR problem used here is associated to the ODE system that relies on Leray projections appearing after the transformation to a set of an ODEs. We will explain how to avoid an explicit use of the Leray projection while solving the obtained LQR problem.

The remainder of this paper is structured as follows. In Section 2, we describe the incompressible Navier-Stokes equations with the linearization around a steady state, and its descriptor index-2 system of differential algebraic equations that arise after a mixed finite element method. The derivation of the obtained ODE system is also explained. Section 3 deals with the extended block Krylov subspace method that allows us to construct an appropriate reduced model to the ODE system by avoiding the dense projection matrix that appears after the transformation to ODEs. In Section 4, a Riccati feedback approach is explained and we show how to solve the LQR problem associated with the ODE system. This approach is based on solving a large-scale algebraic Riccati equation using an extended block Krylov subspace method. In the last section, we provide some numerical experiments to show the effectiveness of the proposed approaches.

2. Navier-Stokes equations (NSEs) : Linearization and Discretization

Navier-Stokes equations for a viscous, incompressible Newtonian fluid in a bounded domain $\Omega \subset \mathbb{R}^2$ with boundary $\partial\Omega$ are given by

$$\begin{cases} \dfrac{\partial z}{\partial t} - \dfrac{1}{\mathrm{Re}}\Delta z + (z \cdot \nabla)z + \nabla p &= f, \\ \nabla \cdot z &= 0, \end{cases} \tag{1}$$

where for $t \in [0, \infty)$ and $x = [x_1 \quad x_2]^T \in \Omega \subset \mathbb{R}^2$, the vector $z(t,x) = [z_1(t,x), z_2(t,x)] \in \mathbb{R}^2$ refers to the velocity field, $p(t,x) \in \mathbb{R}$ is the pressure field, f is known as the forcing term and

$\mathrm{Re} \in \mathbb{R}^+$ is the Reynolds number. The operators Δ, ∇ and $\nabla\cdot$ are defined as the Laplacien, the Gradient and Divergence operators, respectively. The convective term in our model is a non-linear operator defined as

$$(z \cdot \nabla)z = \begin{bmatrix} z_1 \dfrac{\partial z_1}{\partial x_1} + z_2 \dfrac{\partial z_1}{\partial x_2} \\ z_1 \dfrac{\partial z_2}{\partial x_1} + z_2 \dfrac{\partial z_2}{\partial x_2} \end{bmatrix}.$$

The boundary $\Gamma = \partial\Omega$ can be partitioned as follows

$$\Gamma = \Gamma_{in} \cup \Gamma_{out} \cup \Gamma_{wall} \cup \Gamma_{feed}.$$

We therefore impose the following boundary conditions on the respective parts of the boundary

$$z = \begin{cases} \phi_{feed} & on \quad \Gamma_{feed}, \\ \phi_{in} & on \quad \Gamma_{in}, \\ 0 & on \quad \Gamma_{wall}. \end{cases}$$

The condition given below called, the *do-nothing* condition

$$-\frac{1}{\mathrm{Re}} \nabla z \, n + pn = 0 \quad \text{on} \quad \Gamma_{out},$$

where n denotes the unit outer normal vector to Γ_{out}.

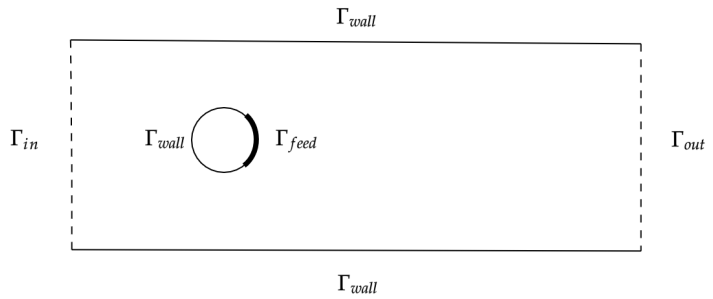

Figure 1: Domain Ω represented by a cylinder wake.

Navier-Stokes equations (NSEs) were derived independently by G.G. Stokes and C.L. Navier in the early 1800's. These equations describe the relationship between the velocity and the pressure of a moving fluid. NSEs represent the conservation of momentum. The fact that the convection term $(z \cdot \nabla)z$ is non-linear is what makes the NSEs complex. For incompressible flows, the second equation of the system (1) is called the continuity equation. In what follows, we present a linearization approach as it is described in [9].

2.1. Linearization

We consider a stationary motion of an incompressible fluid described by the velocity and pressure couple $(w_s(t,x), p_s(t,x))$ that fulfills the stationary Navier-Stokes equations

$$\begin{aligned} -\frac{1}{\mathrm{Re}}\Delta w_s + (w_s \cdot \nabla) w_s + \nabla p_s &= f, \\ \nabla \cdot w_s &= 0. \end{aligned} \tag{2}$$

Here, the same boundary and initial conditions of the first equations are considered. The pair (w_s, p_s) depicts the desired stationary but possibly unstable solution of system (1).

We define the following differences

$$\begin{aligned} v(t,x) &= z(t,x) - w_s(x), \\ \chi(t,x) &= p(t,x) - p_s(t,x). \end{aligned}$$

Replacing in (1) and dropping the non-linear term, we obtain the following linearized Navies-Stokes equations

$$\frac{\partial v}{\partial t} - \frac{1}{\mathrm{Re}}\Delta v + (w_s \cdot \nabla) v + (v \cdot \nabla) w_s + \nabla \chi = 0, \tag{3a}$$

$$\nabla \cdot v = 0, \tag{3b}$$

defined for $t \in [0, \infty)$ and $x \in \Omega \subset \mathbb{R}^2$ with Drichlet boundary conditions

$$v = 0 \quad \text{on} \quad \Gamma_{in} \cup \Gamma_{wall}, \tag{3c}$$

$$v = \phi_{feed} \quad \text{on} \quad \Gamma_{feed}, \tag{3d}$$

a do-nothing condition is described as

$$-\frac{1}{\mathrm{Re}}\nabla v\, n + \chi\, n = 0 \quad \text{on} \quad \Gamma_{out},$$

and the initial condition

$$v(0, \cdot) = 0 \quad \text{in } \Omega.$$

v is defined as perturbation of our flow field z from the desired stationary flow field w_s. A *zero output* for $t \to \infty$ implies that v approximates w_s for $t \to \infty$. As a consequence our flow field achieves the properties of the desired stationary flow field.

2.2. The discrete equations

The choice of an appropriate discretization technique depends on the specific governing equations used, for example (compressible or incompressible flow (our case), mesh type (structured, unstructured). The classical discretization techniques are finite difference, finite element

and finite volume. One of the known methods used to discretize instationary problems, is the method of lines which is based on the replacement of the spatial derivatives in the PDE with algebraic approximations leading to a system of ODEs that approximates the original PDE. In this subsection, we briefly present the main properties of the discrete equations already established in [9]. After using a mixed finite element method, we obtain a system of differential algebraic equations of the form

$$M\,\frac{d}{dt}\mathbf{v}(t) = A\mathbf{v}(t) + G\mathbf{p}(t) + \mathbf{f}(t), \tag{4a}$$

$$0 = G^T\mathbf{v}(t), \tag{4b}$$

where

$\mathbf{v}(t) \in \mathbb{R}^{n_v}$: the nodal vector of the discretized velocity.

$\mathbf{p}(t) \in \mathbb{R}^{n_p}$: the discretized pressure.

90

$\mathbf{f}(t) \in \mathbb{R}^{n_v}$: the forcing term that contains the control.

In what follows, we assume that the forcing term $f(t)$ is given by

$$\mathbf{f}(t) = B\mathbf{u}(t).$$

Moreover, the matrices $M = M^T \succ 0 \in \mathbb{R}^{n_v \times n_v}$ and $A \in \mathbb{R}^{n_v \times n_v}$ are supposed to be large and sparse. They represent the mass matrix and system matrix, respectively. $G \in \mathbb{R}^{n_v \times n_p}$ is a full rank matrix represents the discretized gradient and $B \in \mathbb{R}^{n_v \times n_b}$ is the input matrix. The system matrix $A \in \mathbb{R}^{n_v \times n_v}$ can be decomposed as follows

$$A = -\frac{1}{\mathrm{Re}}L - K - R.$$

More precisely, $-L\mathbf{v}$ represents the discrete Laplacien $\Delta\,\mathbf{v}$, $-K\mathbf{v}$ is the discrete convection resulting from $(w \cdot \nabla)\mathbf{v}$ and $-R\mathbf{v}$ refers to the discrete reaction process of $(\mathbf{v} \cdot \nabla)w$. In the Stokes system, A is symmetric negative definite matrix since there is no role to the matrices K and R. A computational methods based on Krylov projection techniques and interpolatory projection to built a reduced system to a Stokes system have been established respectively in [27, 28]. We add to the system (4) an output function given by

$$y(t) = C\mathbf{v}(t),$$

where $y(t)$ is the output vector and $C^T \in \mathbb{R}^{n_v \times n_c}$ is the output matrix that measures velocity behaviour using information from internal nodes [9]. The system (4) can be rewritten in a

compact form

$$\begin{cases} \underbrace{\begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} \dot{\mathbf{v}}(t) \\ \dot{\mathbf{p}}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{p}(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \mathbf{u}(t), \\ \mathbf{y}(t) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}(t) \\ \mathbf{p}(t) \end{bmatrix}, \end{cases} \tag{5}$$

and we call it a descriptor system since the matrix $\mathbf{M}$ is singular. It uses the following matrix-pencil

$$\left(\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix}, \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix} \right). \tag{6}$$

This matrix pencil has $n_v - n_p$ finite eigenvalues $\lambda_i \in \mathbb{C} \backslash 0$ and $2n_p$ infinite eigenvalues $\lambda_\infty = \infty$, see Theorem 2.1 in [29]. The system (5) is known as an index-2 descriptor dynamical system, for more details about the index of differential algebraic equation, see [30]. For a Reynolds number ($Re \geq 300$), some eigenvalues of the matrix pencil $(\mathbf{A}, \mathbf{M})$ lie in $\mathbb{C}^+$, see [9].

Next, we present a whole process that allows us to reduce such systems. We describe a model reduction technique via a Krylov subspace-based method in order to construct an efficient reduced order system to (5) that has nearly the same response characteristics. To guarantee a well processing of our suggested method, we need to establish a transformation of the system (5) into an ordinary differential equations (ODEs).

2.3. Deriving the ODE system

We first eliminate the discrete pressure $\mathbf{p}$ from (4a) using the following projection operator

$$\Pi = I_n - G\,(G^T M^{-1} G)^{-1}\, G^T M^{-1} \in \mathbb{R}^{n_v \times n_v}.$$

It is easy to check that

$$(\Pi^T)^2 = \Pi^T, \quad \Pi^2 = \Pi, \quad \Pi\, G = 0, \quad \Pi\, M = M\, \Pi^T \text{ and } M^{-1}\, \Pi = \Pi^T M^{-1}.$$

The projection Π^T is an M-orthogonal projection where for x, $y \in \mathbb{R}^{n_v}$ and $M \in \mathbb{R}^{n_v \times n_v}$, the M-inner product is defined by

$$< x, y >_M = (x, My) = y^T M x \quad (M \text{ is a symmetric positive definite matrix}).$$

Notice that

$$\text{null}(\Pi^T) = \text{range}(M^{-1}G) \quad \textit{and} \quad \text{range}(\Pi^T) = \text{null}(G^T).$$

By using all these properties we can show that

$$0 = G^T \mathbf{v}(t) \qquad \text{if and only if} \qquad \mathbf{v}(t) = \Pi^T \mathbf{v}(t). \tag{7}$$

Multiplying (4a) by $G^T\,M^{-1}$ and using (4b), the term $\mathbf{p}$ can be expressed as follows

$$\mathbf{p}(t) = -(G^T\,M^{-1}\,G)^{-1}G^T\,M^{-1}\,A\mathbf{v}(t) - (G^T\,M^{-1}\,G)^{-1}G^T\,M^{-1}\,B\mathbf{u}(t).$$

Replacing $\mathbf{p}$ in (4a) and multiplying by Π yields to the following projected system

$$\mathcal{M}\,\frac{d}{dt}\mathbf{v}(t) = \mathcal{A}\mathbf{v}(t) + \mathcal{B}\mathbf{u}(t), \tag{8a}$$

$$\mathbf{y}(t) = \mathcal{C}\mathbf{v}(t). \tag{8b}$$

Where $\mathcal{A} = \Pi\,A\,\Pi^T$, $\mathcal{M} = \Pi\,M\,\Pi^T$, $\mathcal{B} = \Pi\,B$ and $\mathcal{C} = C\,\Pi^T$. Since the matrix-pencil given by (6) has $n_v - n_p$ finite eigenvalues [29], a decomposition of Π can be made by employing the thin singular value decomposition which leads to the following decomposition

$$\Pi = \Theta_l\Theta_r^T,$$

where Θ_l, $\Theta_r \in \mathbb{R}^{n_v\times(n_v-n_p)}$, are full rank matrices satisfying

$$\Theta_l^T\Theta_r = \mathrm{I}_{(n_v-n_p)}.$$

By inserting this decomposition into (8) and considering a new variable $\tilde{\mathbf{v}}(t) = \Theta_l^T\mathbf{v}(t)$ with $\Theta_r\tilde{\mathbf{v}}(t) = \Theta_r\Theta_l^T\mathbf{v}(t) = \Pi^T\mathbf{v}(t) = \mathbf{v}(t)$, we get the following ODE system

$$M_\Theta\,\frac{d}{dt}\tilde{\mathbf{v}}(t) = A_\Theta\tilde{\mathbf{v}}(t) + B_\Theta\mathbf{u}(t), \tag{9a}$$

$$y(t) = C_\Theta\tilde{\mathbf{v}}(t), \tag{9b}$$

where $M_\Theta = \Theta_r^T M\Theta_r$, $A_\Theta = \Theta_r^T A\Theta_r \in \mathbb{R}^{(n_v-n_p)\times(n_v-n_p)}$, $B_\Theta = \Theta_r^T B \in \mathbb{R}^{(n_v-n_p)\times n_b}$, $C_\Theta = C\Theta_r \in \mathbb{R}^{n_c\times(n_v-n_p)}$. The matrix M_Θ is non-singular due to the fact that M is symmetric and positive definite. Notice that the three systems (4), (8) and (9) are equivalent in the sense that their finite spectrum is the same [31] and also they realize the same transfer function. Before proving this result we give the definition of a transfer function associated to the dynamical system (9), to this end, we apply the Laplace transform given by

$$L(f)(s) := \int_0^\infty e^{-st}f(t)dt,$$

to the system (9), then we get the new system in the frequency domain

$$\begin{cases} sM_\Theta\,\widetilde{\mathbf{V}}(s) &= A_\Theta\,\widetilde{\mathbf{V}}(s) + B_\Theta\,U(s), \\ Y(s) &= C_\Theta\,\widetilde{\mathbf{V}}(s). \end{cases}$$

Where $\widetilde{\mathbf{V}}(s)$, $\mathbf{U}(s)$ and $\mathbf{Y}(s)$ are the Laplace transform of $\tilde{\mathbf{v}}(t)$, $\mathbf{u}(t)$ and $y(t)$ respectively. By eliminating $\widetilde{\mathbf{V}}(s)$ from the two equations, we obtain

$$\mathbf{Y}(s) = F_\Theta(s)\,\mathbf{U}(s),$$

where

$$F_\Theta(s) = C_\Theta(sM_\Theta - A_\Theta)^{-1}B_\Theta, \tag{10}$$

is the transfer function associated to the system (9).

Remark 1. *Let F_m be the transfer function associated to the reduced system. In order to measure the accuracy of the resulting reduced system, we have to compute the error $\|F_\Theta - F_m\|$ with respect to a specific norm. This error can also be used to know how the response of the reduced system is close to that of the original one since $\|\mathbf{Y}(s) - \mathbf{Y}_m(s)\| \leq \|F_\Theta(s) - F_m(s)\|\,\|\mathbf{U}(s)\|$.*

Denote by $X = \Theta_r(sM_\Theta - A_\Theta)^{-1}B_\Theta$, then $F_\Theta = CX$. In addition, X satisfies

$$B_\Theta = (sM_\Theta - A_\Theta)\Theta_l^T X,$$

or equivalently,

$$\Pi B = \Pi(sM - A)\Pi^T X.$$

Due to the facts that $range(\Pi^T) = null(G^T)$ and G is of full rank, we can verify that

$$\begin{bmatrix} sM - A & -G \\ -G^T & 0 \end{bmatrix} \begin{bmatrix} X \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix}.$$

In fact, the relation $range(\Pi^T) = null(G^T)$ guarantees

$$G^T X = 0,$$

and the full rank G leads to

$$\star = (G^T G)^{-1} G^T[(sM - A)X - B],$$

thus, the desired result

$$F_\Theta(s) = C\,X = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} X \\ \star \end{bmatrix} = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} sM - A & -G \\ -G^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} B \\ 0 \end{bmatrix} = F(s), \tag{11}$$

where $F(s)$ is the transfer function associated to the original system (4). The technique used here allows us to solve a saddle point problem instead of solving a linear system depending on the dense matrix Π and its Θ-decomposition as established earlier in [23].

Remark 2. *We notice that instead of reducing the original system (5), we can reduce the ODE system (9) since it has the same transfer functions as it is shown in (11).*

The matrices involved in (9) are dense due to the projector Π and its Θ-decomposition and that is why we need a strategy to avoid using direct computations with these matrices. In the next subsection, we show how to construct a reduced order system to (9) by using the structure of the original system (5) without requiring any explicit computation of the dense matrices $(M_\Theta, A_\Theta, B_\Theta, C_\Theta)$, and this leads to a considerable saving of cost and storage. Our calculations involve the implicit use of the system (9) and this implies solving saddle point problems. Details are given in the next section.

3. A model reduction method to a descriptor index-2 dynamical system

Our goal is to find a reduced system to (9) since it realizes the same transfer function of (5) as we mentioned before. This new system can be constructed using a projection technique onto an extended block Krylov subspace that is defined in the following subsection.

3.1. The extended block Arnoldi algorithm

Multiplying from the left of the first equation of the system (9) by the inverse of M_Θ gives the following system which will be called the Θ-*system*

$$\frac{d}{dt}\tilde{\mathbf{v}}(t) = M_\Theta^{-1}A_\Theta\tilde{\mathbf{v}}(t) + M_\Theta^{-1}B_\Theta\mathbf{u}(t), \tag{12a}$$

$$y(t) = C_\Theta\tilde{\mathbf{v}}(t). \tag{12b}$$

The extended block Krylov subspace associated to the pair $(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$ is defined as follows

$$\begin{aligned}\mathbb{K}_m^{ext}(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta) = \texttt{Range}([&(M_\Theta^{-1}A_\Theta)^{-m}(M_\Theta^{-1}B_\Theta), \ldots, (M_\Theta^{-1}A_\Theta)^{-1}(M_\Theta^{-1}B_\Theta)\\ &(M_\Theta^{-1}B_\Theta), (M_\Theta^{-1}A_\Theta)(M_\Theta^{-1}B_\Theta), \ldots, (M_\Theta^{-1}A_\Theta)^{m-1}(M_\Theta^{-1}B_\Theta)]).\end{aligned}$$

The extended block Arnoldi algorithm for the pair $(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$ is summarized in the following algorithm.

Algorithm 1 The extended block Arnoldi algorithm associated to the Θ *system*

- Inputs: $M_\Theta \in \mathbb{R}^{(n_v-n_p)\times(n_v-n_p)}$, $A_\Theta \in \mathbb{R}^{(n_v-n_p)\times(n_v-n_p)}$, $B_\Theta \in \mathbb{R}^{(n_v-n_p)\times n_b}$ and m.
- Compute $[\mathcal{V}_1^b, \Lambda] = \mathtt{qr}([M_\Theta^{-1}B_\Theta, (M_\Theta^{-1}A_\Theta)^{-1}\, M_\Theta^{-1}B_\Theta])$.
- For $j = 1, \ldots, m$
  1. Set $\mathcal{V}_j^{(1)}$: first n_b columns of $\mathcal{V}_j^b$; $\mathcal{V}_j^{(2)}$: second n_b columns of $\mathcal{V}_j^b$.
  2. $\widetilde{\mathcal{V}}_{j+1} = [(M_\Theta^{-1}A_\Theta)\, \mathcal{V}_j^{(1)}, (M_\Theta^{-1}A_\Theta)^{-1}\, \mathcal{V}_j^{(2)}]$.
  3. Orthogonalize $\widetilde{\mathcal{V}}_{j+1}$ with respect to $\mathcal{V}_1^b, \ldots, \mathcal{V}_j^b$ to get $\mathcal{V}_{j+1}^b$, i.e.,
     for $i = 1, 2, \ldots, j$
     $H_{i,j} = (\mathcal{V}_i^b)^T\, \widetilde{\mathcal{V}}_{j+1}$.
     $\widetilde{\mathcal{V}}_{j+1} = \widetilde{\mathcal{V}}_{j+1} - \mathcal{V}_i^b\, H_{i,j}$.
     end for
  4. $[\mathcal{V}_{j+1}^b,\ H_{j+1,j}] = QR(\widetilde{\mathcal{V}}_{j+1})$.
  5. $\mathcal{V}_{j+1} = [\mathcal{V}_j,\ \mathcal{V}_{j+1}^b]$.

  End For.

The extended block Arnoldi algorithm allows us to construct an orthonormal basis of $\mathbb{K}_m^{ext}(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$ formed by the columns of $\{\mathcal{V}_1^b, \ldots, \mathcal{V}_m^b\}$, where $\mathcal{V}_j^b$ for $(j = 1, \ldots, m)$ are $(n_v - n_p) \times 2n_b$ matrices. We also have some classical algebraic properties given in the following proposition.

Proposition 1. *Let $\mathcal{V}_m = [\mathcal{V}_1^b, \ldots, \mathcal{V}_m^b] \in \mathbb{R}^{2mn_b\times 2mn_b}$ be the matrix generated using the extended block Arnoldi Algorithm 1 to the pairs $(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$, $\mathbb{T}_m = \mathcal{V}_m^T\, M_\Theta^{-1}A_\Theta\, \mathcal{V}_m$. Then we have the following results*

$$M_\Theta^{-1}A_\Theta\, \mathcal{V}_m = \mathcal{V}_{m+1}\, \overline{\mathbb{T}}_m \tag{13}$$

$$= \mathcal{V}_m\, \mathbb{T}_m + \mathcal{V}_{m+1}^b\, T_{m+1,m} E_m^T, \tag{14}$$

where $T_{m+1,m}$ is the last $2n_b \times 2n_b$ block of $\overline{\mathbb{T}}_m \in \mathbb{R}^{2(m+1)n_b\times 2mn_b}$ and E_m^T is the last $2n_b$ columns of the identity matrix I_{2mn_b}.

Proof 1. *Using the fact that $M_\Theta^{-1}A_\Theta\mathbb{K}_m^{ext}(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta) \subset \mathbb{K}_{m+1}^{ext}(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$ and the orthogonality of $\mathcal{V}_m$, there exists a matrix L such that*

$$M_\Theta^{-1}A_\Theta\, \mathcal{V}_m = \mathcal{V}_{m+1}\, L. \tag{15}$$

It has been shown that $\mathbb{T}_m$ is an upper block Hessenberg matrix in [16, 32] and also that $\mathbb{T}_m$ can be computed directly from the columns of the upper block Hessenberg matrix $\mathbb{H}_m$ generated by Algorithm 2. Since $\mathcal{V}_{m+1} = [\mathcal{V}_m, \mathcal{V}^b_{m+1}]$, we have

$$\begin{aligned}
\mathbb{T}_{m+1} &= \mathcal{V}^T_{m+1}\, M_\Theta^{-1} A_\Theta\, \mathcal{V}_{m+1} \\
&= \begin{bmatrix} \mathcal{V}^T_m\, M_\Theta^{-1} A_\Theta\, \mathcal{V}_m & \mathcal{V}^T_m M_\Theta^{-1} A_\Theta\, \mathcal{V}^b_{m+1} \\ (\mathcal{V}^b_{m+1})^T\, M_\Theta^{-1} A_\Theta\, \mathcal{V}_m & (\mathcal{V}^b_{m+1})^T M_\Theta^{-1} A_\Theta \mathcal{V}^b_{m+1} \end{bmatrix} \\
&= \begin{bmatrix} \mathbb{T}_m & \mathcal{V}^T_m M_\Theta^{-1} A_\Theta\, \mathcal{V}^b_{m+1} \\ (\mathcal{V}^b_{m+1})^T\, M_\Theta^{-1} A_\Theta\, \mathcal{V}_m & (\mathcal{V}^b_{m+1})^T M_\Theta^{-1} A_\Theta \mathcal{V}^b_{m+1} \end{bmatrix}.
\end{aligned}$$

We know that $\mathbb{T}_{m+1}$ is also un upper block Hessenberg matrix, then

$$T_{m+1,m}\, E_m^T = (\mathcal{V}^b_{m+1})^T\, M_\Theta^{-1} A_\Theta\, \mathcal{V}_m,$$

and

$$\overline{\mathbb{T}}_m = \mathcal{V}^T_{m+1} M_\Theta^{-1} A_\Theta \mathcal{V}_m = \begin{bmatrix} \mathbb{T}_m \\ T_{m+1,m} E_m^T \end{bmatrix} \in \mathbb{R}^{2(m+1)n_b \times 2mn_b}.$$

Multiplying by $\mathcal{V}^T_{m+1}$ from the left of (15), we obtain $\overline{\mathbb{T}}_m = L$. As a consequence we get the desired result

$$\begin{aligned}
M_\Theta^{-1} A_\Theta\, \mathcal{V}_m &= \mathcal{V}_{m+1}\, \overline{\mathbb{T}}_m \\
&= [\mathcal{V}_m, \mathcal{V}^b_{m+1}] \begin{bmatrix} \mathbb{T}_m \\ T_{m+1,m} E_m^T \end{bmatrix} \\
&= \mathcal{V}_m\, \mathbb{T}_m + \mathcal{V}^b_{m+1}\, T_{m+1,m} E_m^T.
\end{aligned}$$

After constructing the matrix $\mathcal{V}_m$ corresponding to the basis of the extended block Krylov subspace $\mathbb{K}^{ext}_m(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$, we can now built the reduced system by considering the approximation $\tilde{\mathbf{v}}(t) \approx \mathcal{V}_m \mathbf{v}_m(t)$ and by replacing in (9), and then imposing the Petrov-Galerking condition, we obtain the following projected reduced order dynamical system

$$\begin{cases} \mathcal{M}_m \dot{\mathbf{v}}_m(t) &= \mathcal{A}_m\, \mathbf{v}_m(t) + \mathcal{B}_m\, \mathbf{u}(t), \\ y_m(t) &= \mathcal{C}_m\, \mathbf{v}_m(t), \end{cases} \tag{16}$$

with the associated transfer function $F_m(s) = \mathcal{C}_m(s\mathcal{M}_m - \mathcal{A}_m)^{-1}\mathcal{B}_m$, where $\mathcal{M}_m = \mathcal{V}_m^T \mathcal{M} \mathcal{V}_m$, $\mathcal{A}_m = \mathcal{V}_m^T \mathcal{A} \mathcal{V}_m \in \mathbb{R}^{2mn_b \times 2mn_b}$ and $\mathcal{B}_m = \mathcal{V}_m^T \mathcal{B} \in \mathbb{R}^{2mn_b \times n_b}$, $\mathcal{C}_m = C\mathcal{V}_m \in \mathbb{R}^{n_c \times 2mn_b}$.

As we mentioned before, the explicit computation of $\mathcal{V}_m$ is prohibitive in our approach since the j-th block $\mathcal{V}^b_j$ of $\mathcal{V}_m$ relies on Θ_r, which will make our calculations infeasible due to the density of the Θ-decomposition of the projection Π. In what follows, we describe an appropriate process to get a reduced system to (12) by avoiding an explicit computation of $\mathcal{V}_m$.

The main computational issue when we apply the extended block Arnoldi Algorithm 1 to the pair $(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$ is to compute blocks of the form

$$\widetilde{\mathcal{V}}_1 = [M_\Theta^{-1}B_\Theta, (M_\Theta^{-1}A_\Theta)^{-1}M_\Theta^{-1}B_\Theta] \tag{17}$$

$$= [\widetilde{\mathcal{V}}_1^{(1)}, \widetilde{\mathcal{V}}_1^{(2)}], \tag{18}$$

and for $j = 1, \ldots, m$,

$$\widetilde{\mathcal{V}}_{j+1} = [(M_\Theta^{-1}A_\Theta)\,\mathcal{V}_j^{(1)}, (M_\Theta^{-1}A_\Theta)^{-1}\,\mathcal{V}_j^{(2)}] \tag{19}$$

$$= [\widetilde{\mathcal{V}}_{j+1}^{(1)}, \widetilde{\mathcal{V}}_{j+1}^{(2)}], \tag{20}$$

where $\mathcal{V}_j^{(1)}$ and $\mathcal{V}_j^{(2)}$ are the first and second n_b columns of $\mathcal{V}_j^b$, respectively. Our strategy consists in reformulating those blocks onto new ones without an explicit calculation of Θ_r. We set $\widetilde{\mathbb{V}}_m = \Theta_r\widetilde{\mathcal{V}}_m \in \mathbb{R}^{n_v\times 2mn_b}$ where $\widetilde{\mathcal{V}}_m = [\widetilde{\mathcal{V}}_1, \ldots, \widetilde{\mathcal{V}}_m] \in \mathbb{R}^{(n_v-n_p)\times 2mn_b}$ and $\widetilde{\mathbb{V}}_m = [\widetilde{V}_1, \ldots, \widetilde{V}_m] \in \mathbb{R}^{n_v\times 2mn_b}$ satisfying

$$\Pi^T\widetilde{\mathbb{V}}_m = \Theta_r\Theta_l^T\widetilde{\mathbb{V}}_m = \Theta_r\widetilde{\mathcal{V}}_m = \widetilde{\mathbb{V}}_m, \tag{21}$$

We set again $\mathbb{V}_m = \Theta_r\mathcal{V}_m \in \mathbb{R}^{n_v\times 2mn_b}$. All the j-th block $V_j \in \mathbb{R}^{n_v\times 2n_b}$ of $\mathbb{V}_m$ are computed in an appropriate way, which means that we do not include the matrix Θ_r in our computation and also not the block $\mathcal{V}_j^b$. Details are given in Algorithm 2.

$$\Pi^T\mathbb{V}_m = \Theta_r\Theta_l^T\mathbb{V}_m = \Theta_r\mathcal{V}_m = \mathbb{V}_m, \tag{22}$$

The result (21) confirms that $G^T\widetilde{\mathbb{V}}_m = 0$ as it is shown in (7), and consequently we obtain the following relations

- $M_\Theta^{-1}B_\Theta = \widetilde{\mathcal{V}}_1^{(1)}$,
- $M_\Theta\widetilde{\mathcal{V}}_1^{(1)} = B_\Theta$,
- $\Theta_r^T M\Theta_r\widetilde{\mathcal{V}}_1^{(1)} = \Theta_r B$,
- $\Pi M\Pi^T\widetilde{V}_1^{(1)} = \Pi B$,
- $\Pi(M\widetilde{V}_1^{(1)} - B) = 0$,
- $(M\widetilde{V}_1^{(1)} - B) \in null(\Pi) = range(G)$.

Then, the first n_b block-column $\widetilde{V}_1^{(1)}$ of $\widetilde{V}_1 \in \mathbb{R}^{n_v\times 2n_b}$ can be computed by solving the following saddle point problem

$$\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix}\begin{bmatrix} \widetilde{V}_1^{(1)} \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix}.$$

The same process can be used to get $\widetilde{V}_1^{(2)}$ by starting from the linear system $(M_\Theta^{-1}A_\Theta)^{-1}\, M_\Theta^{-1}B_\Theta = \widetilde{\mathcal{V}}_1^{(2)}$. After that, one can use the `qr` function (in MATLAB) to find the block $V_1 = [V_1^{(1)}, V_1^{(2)}] \in \mathbb{R}^{n_v \times 2n_b}$ as described in Algorithm 2. To get the first n_b block-column $\widetilde{V}_{j+1}^{(1)}$ of $\widetilde{V}_{j+1}$, we use the following steps

- $(M_\Theta^{-1}A_\Theta)\mathcal{V}_j^{(1)} = \widetilde{\mathcal{V}}_{j+1}^{(1)}$,
- $M_\Theta \widetilde{\mathcal{V}}_{j+1}^{(1)} = A_\Theta \mathcal{V}_j^{(1)}$,
- $\Theta_r^T M \Theta_r \widetilde{\mathcal{V}}_{j+1}^{(1)} = \Theta_r^T A \Theta_r \mathcal{V}_j^{(1)}$,
- $\Pi M \Pi^T \widetilde{V}_{j+1}^{(1)} = \Pi A V_j^{(1)}$,
- $\Pi(M\widetilde{V}_{j+1}^{(1)} - AV_j^{(1)}) = 0$,
- $(M\widetilde{V}_{j+1}^{(1)} - AV_j^{(1)}) \in null(\Pi) = range(G)$.

Then we have to solve the following saddle point problem

$$\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{V}_{j+1}^{(1)} \\ \star \end{bmatrix} = \begin{bmatrix} AV_j^{(1)} \\ 0 \end{bmatrix}.$$

In the same manner, we can compute the last n_b column $\widetilde{V}_{j+1}^{(2)}$ of $\widetilde{V}_{j+1}$ by starting from this linear system $(M_\Theta^{-1}A_\Theta)^{-1}\, \mathcal{V}_j^{(2)} = \widetilde{\mathcal{V}}_{j+1}^{(2)}$ and following the same previous process.

After showing how to compute the block vectors (17) and (19) without computing neither the matrix $\mathcal{V}_m$ corresponding to the orthonormal basis of $\mathbb{K}_m^{ext}(M_\Theta^{-1}A_\Theta, M_\Theta^{-1}B_\Theta)$ nor Θ-decomposition of Π, we can now present the new extended block Arnoldi algorithm based only on the sparse system matrices of the index-2 system. Here, we have to mention that this algorithm is based on a Gram-Shmidt orthogonalization process, which reconstructs the blocks $\{V_1, \ldots, V_m\}$, such that their columns form an orthonormal matrix $\mathbb{V}_m$ as described in Algorithm 2 step 3.c. This matrix will be used in order to get an efficient reduced system to the index-2 original one (4). Details are given in the next subsections. We summarize all these steps in the following algorithm.

Algorithm 2 The extended block Arnoldi algorithm associated to the index-2 system

Inputs: $M \in \mathbb{R}^{n_v \times n_v}$, $A \in \mathbb{R}^{n_v \times n_v}$, $G \in \mathbb{R}^{n_v \times n_p}$, $B \in \mathbb{R}^{n_v \times n_b}$ and m.

1. solving the first saddle point problems

$$\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{V}_1^{(1)} \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix}, \begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{V}_1^{(2)} \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix},$$

2. Compute $[V_1, \Lambda] = \mathtt{qr}\left([\widetilde{V}_1^{(1)}, \widetilde{V}_1^{(2)}]\right)$, $\mathbb{V}_1 = [V_1]$.

3. For $j = 1, \ldots, m$

   a. Set $V_j^{(1)}$: first n_b columns of V_j; $V_j^{(2)}$: second n_b columns of V_j.

   b. $\widehat{V}_{j+1} = \left(\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{V}_{j+1}^{(1)} \\ \star \end{bmatrix} = \begin{bmatrix} A\, V_j^{(1)} \\ 0 \end{bmatrix}, \begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} \widetilde{V}_{j+1}^{(2)} \\ \star \end{bmatrix} = \begin{bmatrix} M\, V_j^{(2)} \\ 0 \end{bmatrix} \right).$

   c. Orthogonalize $\widehat{V}_{j+1}$ with respect to $V_1, \ldots, V_j$ to get V_{j+1}, i.e.,

      for $i = 1, 2, \ldots, j$

      $H_{i,j} = (V_i)^T\, \widehat{V}_{j+1}$;

      $\widehat{V}_{j+1} = \widehat{V}_{j+1} - V_i\, H_{i,j}$;

      end for

   d. $[V_{j+1},\ H_{j+1,j}] = QR(\widehat{V}_{j+1})$.

   e. $\mathbb{V}_{j+1} = [\mathbb{V}_j,\ V_{j+1}]$.

End For.

As we noticed, the main steps of Algorithm 2 is the solution of a saddle-point problems of $n_v + n_p$ dimension in Step 1 and in Step 3.b, and we are interesting only in the first n_v rows. At each iteration, a direct solver "\", a built-in function on MATLAB, is used to solve these saddle point problems. The new vector V_{j+1} of the matrix $\mathbb{V}_m$ can be computed via the Gram-Shmidt process as we explain in the Step 3.c. The " $\star$ " refers to an $n_p \times n_b$ block that is not taken into account. After m steps of Algorithm 2, we get an orthonormal matrix $\mathbb{V}_m = [V_1, V_2, \ldots, V_m] \in \mathbb{R}^{n_v \times 2mn_b}$ with $V_i \in \mathbb{R}^{n_v \times 2n_b}$. This algorithm built also an upper block Hessenberg matrix $\mathbb{H}_m \in \mathbb{R}^{2mn_b \times 2mn_b}$ whose non zero blocks are the $H_{i,j}$. Notice that each submatrix $H_{i,j}$ $(1 \leq i \leq j \leq m)$ is of order $2n_b \times 2n_b$. A similar algebraic relations to the one given by (13) can be derived using only the sparse matrices M, A and also the matrix $\mathbb{V}_m$ generated by Algorithm 2. We present this result in the following proposition.

Proposition 2. *Let* $\mathbb{V}_m \in \mathbb{R}^{n_v \times 2mn_b}$ *and* $\overline{\mathbb{T}}_m \in \mathbb{R}^{2(m+1)n_b \times 2mn_b}$ *be the orthonormal matrix and the upper block Hessenberg matrix generated by Algorithm2, respectively. Then we have*

$$\begin{aligned} M^{-1}\Pi\, A\mathbb{V}_m &= \mathbb{V}_{m+1}\,\overline{\mathbb{T}}_m \\ &= \mathbb{V}_m\mathbb{T}_m + V_{m+1}T_{m+1,m}E_m^T, \end{aligned}$$

where Π *is the projection matrix defined earlier.*

Proof 2. *Multiplying from the left the relation (13) by* Θ_r*, and using the fact that* $\mathbb{V}_m = \Theta_r\mathcal{V}_m$*, we get*

$$\Theta_r M_\Theta^{-1}\Theta_r^T A\Theta_r\,\mathcal{V}_m = \Theta_r\mathcal{V}_{m+1}\,\overline{\mathbb{T}}_m, \tag{23}$$

$$\Theta_r M_\Theta^{-1}\Theta_r^T A\mathbb{V}_m = \mathbb{V}_{m+1}\,\overline{\mathbb{T}}_m. \tag{24}$$

On the other hand, we know that $\Pi M = M\Pi^T$ *by definition of* Π*, and by using the fact that* $\Pi M\Theta_r = M\Theta_r$ *by the* Θ*-decomposition, we obtain the following relations*

$$\begin{aligned} \Pi M\Theta_r &= M\Theta_r, \\ \Theta_l\Theta_r^T M\Theta_r &= M\Theta_r, \\ \Theta_l M_\Theta &= M\Theta_r, \\ M^{-1}\Theta_l &= \Theta_r M_\Theta^{-1}, \\ \Theta_r M_\Theta^{-1}\Theta_r^T &= M^{-1}\Pi. \end{aligned}$$

Replacing the last relation in the formula (24), we get the desired result.

Notice that from Step 1 of Algorithm 2, we have

$$[V_1, \Lambda] = \mathtt{qr}\left([v, w]\right), \tag{25}$$

where $\Lambda \in \mathbb{R}^{2n_b \times 2n_b}$ is an upper triangular matrix defined by

$$\Lambda = \begin{bmatrix} \Lambda^{(1,1)} & \Lambda^{(1,2)} \\ 0 & \Lambda^{(2,2)} \end{bmatrix},$$

and v, w are the solutions of the following saddle point problems

$$\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix}\begin{bmatrix} v \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix} \; \textit{and} \; \begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix}\begin{bmatrix} w \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix}.$$

We notice that

$$\begin{bmatrix} M & G \\ G^T & 0 \end{bmatrix}\begin{bmatrix} v \\ \star \end{bmatrix} = \begin{bmatrix} B \\ 0 \end{bmatrix} \Leftrightarrow \Pi M\Pi^T v = \Pi B, \;\; (\text{with } \Pi^T v = v),$$

and from (25) we get

$$[v, w] = [V_1^{(1)}, V_1^{(2)}] \begin{bmatrix} \Lambda^{(1,1)} & \Lambda^{(1,2)} \\ 0 & \Lambda^{(2,2)} \end{bmatrix},$$

thus

$$v = V_1^{(1)} \Lambda^{(1,1)},$$

and then

$$\mathbb{V}_m^T M^{-1} \Pi B = \mathbb{V}_m^T V_1^{(1)} \Lambda^{(1,1)} = \begin{bmatrix} I_{n_b} \\ 0_{n_b} \\ \vdots \\ 0_{n_b} \end{bmatrix} \Lambda^{(1,1)}. \tag{26}$$

We have mentioned before that in order to reduce the original system (5), we can construct a reduced system from the Θ system (9) since they realize the same transfer function as it is shown in (11). At the iteration m, we approximate $\tilde{\mathbf{v}}(t)$ by $\mathcal{V}_m \hat{\mathbf{v}}_m(t)$ where $\mathcal{V}_m$ is the matrix corresponding to the orthonormal basis of $\mathbb{K}_m^{ext}(M_\Theta^{-1} A_\Theta, M_\Theta^{-1} B_\Theta)$. By injecting the approximation of $\tilde{\mathbf{v}}(t)$ in the system (9) and enforcing the Petrov-Galerkin condition, we get the following reduced system

$$\begin{cases} \dot{\hat{\mathbf{v}}}_m(t) &= \mathcal{V}_m^T M_\Theta^{-1} A_\Theta \mathcal{V}_m \hat{\mathbf{v}}_m(t) + \mathcal{V}_m^T M_\Theta^{-1} B_\Theta \mathbf{u}(t), \\ y_m(t) &= C_\Theta \mathcal{V}_m \hat{\mathbf{v}}_m(t). \end{cases} \tag{27}$$

We know that $\mathbb{T}_m = \mathcal{V}_m^T M_\Theta^{-1} A_\Theta \mathcal{V}_m$ which can be computed only from the upper block Hessenberg matrix $\mathbb{H}_m$ generated by Algorithm 2 as we mentioned before, also $C_\Theta \mathcal{V}_m = C \Theta_r \mathcal{V}_m = C \mathbb{V}_m$, and by using the fact that $M_\Theta^{-1} B_\Theta \in \mathbb{K}_m^{ext}(M_\Theta^{-1} A_\Theta, M_\Theta^{-1} B_\Theta)$ which confirms that $\mathcal{V}_m \mathcal{V}_m^T M_\Theta^{-1} B_\Theta = M_\Theta^{-1} B_\Theta$, then we can prove

$$\mathcal{V}_m^T M_\Theta^{-1} B_\Theta = \mathbb{V}_m^T M^{-1} \Pi B.$$

Finally, we get the following reduced order LTI dynamical system

$$\begin{cases} \dot{\hat{\mathbf{v}}}_\mathbf{m}(\mathbf{t}) &= \mathbb{T}_m \hat{\mathbf{v}}_m(t) + \mathbb{B}_m \mathbf{u}(t), \\ y_m(t) &= \mathbb{C}_m \hat{\mathbf{v}}_m(t), \end{cases} \tag{28}$$

where $\mathbb{B}_m = \Big[I_{n_b}, 0_{n_b}, \ldots, 0_{n_b} \Big]^T \Lambda^{(1,1)} \in \mathbb{R}^{2mn_b \times n_b}$ as it is mentioned in (26), and $\mathbb{C}_m = C \mathbb{V}_m \in \mathbb{R}^{n_c \times 2mn_b}$.

The reduced transfer function is given by

$$F_m(s) = \mathbb{C}_m (s I_{2mn_b} - \mathbb{T}_m)^{-1} \mathbb{B}_m.$$

Another way to construct a reduced system is by considering the system (9) without inverting the matrix M_Θ. We again approximate $\tilde{\mathbf{v}}(t)$ by $\mathcal{V}_m\hat{\mathbf{v}}_m(t)$ where $\mathcal{V}_m$ is a matrix described in the previous sections, then we get the following system

$$\begin{cases} \mathcal{V}_m^T M_\Theta \mathcal{V}_m \dot{\hat{\mathbf{v}}}_\mathbf{m}(\mathbf{t}) &= \mathcal{V}_m^T A_\Theta \mathcal{V}_m \hat{\mathbf{v}}_m(t) + \mathcal{V}_m^T B_\Theta \mathbf{u}(t), \\ y_m(t) &= C_\Theta \mathcal{V}_m \hat{\mathbf{v}}_m(t), \end{cases} \tag{29}$$

using the fact that $\mathbb{V}_m = \Theta_r \mathcal{V}_m$, we get the following reduced system

$$\begin{cases} \mathbb{M}_m \dot{\hat{\mathbf{v}}}_\mathbf{m}(\mathbf{t}) &= \mathbb{A}_m \hat{\mathbf{v}}_m(t) + \mathbb{B}_m \mathbf{u}(t), \\ y_m(t) &= \mathbb{C}_m \hat{\mathbf{v}}_m(t), \end{cases} \tag{30}$$

with the associated reduced transfer function

$$F_m(s) = \mathbb{C}_m(s\mathbb{M}_m - \mathbb{A}_m)^{-1}\mathbb{B}_m,$$

where $\mathbb{M}_m = \mathbb{V}_m^T M \mathbb{V}_m$, $\mathbb{A}_m = \mathbb{V}_m^T A \mathbb{V}_m$, $\mathbb{B}_m = \mathbb{V}_m^T B$ and $\mathbb{C}_m = C\mathbb{V}_m$.

In Algorithm 2 we gave a description of the process to get the matrix $\mathbb{V}_m$ without any explicit computation of $\mathcal{V}_m$ or Θ_r.

Remark 3. *The two reduced dynamical systems (28) and (30) are considered efficient reduced systems compared to the original one represented by the index-2 system (5), but numerically the first reduced system is more economical since its system matrices ($\mathbb{T}_m\,\mathbb{B}_m$, $\mathbb{C}_m$) could be computed appropriately and without requiring matrix-vector products with A and M which is the case for the second reduced system represented by the system matrices ($\mathbb{M}_m, \mathbb{A}_m, \mathbb{B}_m$, $\mathbb{C}_m$).*

4. Solving the LQR problem based on a Riccati feedback approach

The linear quadratic regulator is a well-known classical method for constructing controlled feedback gains. This feedback allows the design of stable and efficient closed-loop systems. We used the transformation explained in Subsection 2.3 that allows us to deal with an LQR problem governed by an ODE instead of an LQR problem governed by an DAE. Following that, a classical LQR theory can be applied to solve the new problem based on a Riccati feedback approach. The main issue with this approach is the solution of a generalized algebraic Riccati equation (GARe). We mentioned earlier that all calculations are performed using the structure of the original DAE system and not that of the ODE one due to the density of projection Π and its Θ-decomposition which can make our calculations infeasible.

4.1. The LQR problem associated to the ODE system

The LQR problem consists in minimizing the following cost functional

$$\mathcal{J}(\tilde{\mathbf{v}}, \mathbf{u}(t)) := \frac{1}{2}\int_0^\infty (\tilde{\mathbf{v}}^T C_\theta^T C_\theta \tilde{\mathbf{v}} + \mathbf{u}(t)^T \mathbf{u}(t))\ \mathrm{dt}, \tag{31}$$

subject to the ODE system (9) constraints defined earlier in Section 2. According to the LQR approach, the optimal control that minimizes the functional coast (31) subject to the dynamical constraints (9) is given by

$$\mathbf{u}_\star(t) = -\underbrace{B_\theta^T X_\theta M_\theta}_{:=K_\Theta}\tilde{\mathbf{v}}, \tag{32}$$

where $X_\theta \in \mathbb{R}^{(n_v-n_p)\times(n_v-n_p)}$ is the unique symmetric semi-definite positive stabilizing solution of the following generalized algebraic Riccati equation (GARe)

$$\mathcal{R}(X_\Theta) := A_\Theta^T X_\Theta\, M_\Theta + M_\Theta\, X_\Theta\, A_\Theta - M_\Theta\, X_\Theta\, B_\Theta B_\Theta^T X_\Theta\, M_\Theta + C_\Theta^T C_\Theta = 0. \tag{33}$$

The unique solution X_Θ can be computed using an extended block Krylov subspace method. This solution is the main ingredient to construct the feedback matrix $K_\Theta \in \mathbb{R}^{n_b\times(n_v-n_p)}$ that asymptotically stabilizes the ODE system (9). However, solving the GARe (33) is not recommended in our process due to the presence of Θ-decomposition of the projection Π. In the next subsections, we describe how to solve such an algebraic equation (33) without using the Θ-decomposition in our computations.

4.2. Solving the generalized algebraic Riccati equation

Our goal is to solve the GARe (33) without any explicit computation of the dense matrices $(M_\Theta, A_\Theta, B_\Theta, C_\Theta)$. This statement intended to the fact that those matrices rely on Θ_r, and a direct use of them can make our calculations impractical due to the density of Θ-decomposition of the projection Π. A multiplication from the left and right of (33) by Θ_l and Θ_l^T, respectively, gives the following result

$$\Pi A^T \Theta_r X_\Theta \Theta_r^T M \Pi^T + \Pi M \Theta_r X_\Theta \Theta_r^T A \Pi^T - \Pi M \Theta_r X_\Theta \Theta_r^T B B^T \theta_r X_\theta \theta_r^T M \Pi^T + \Pi C^T C \Pi^T = 0.$$

Setting $X = \Theta_r X_\Theta \Theta_r^T$, and using the fact that $\Pi M = M\Pi^T$, we get

$$\Pi A^T X \Pi M + M \Pi^T X A \Pi^T - M \Pi^T X B B^T X \Pi M + \Pi C^T C \Pi^T = 0.$$

Since $X\Pi = \Theta_r X_\Theta \Theta_r^T \Theta_l \Theta_r^T = \Theta_r X_\Theta \Theta_r^T = X$, same to $\Pi^T X = X$, then we obtain the following final result

$$\Pi A^T X M + M X A \Pi^T - M X B B^T X M + \Pi C^T C \Pi^T = 0. \tag{34}$$

If we set $K = B^T XM$ the feedback matrix associated to (34), then the relation between K_Θ and K is given as

$$K_\Theta = B^T \Theta_r X_\Theta \Theta_r^T M \Theta_r = B^T X M \Theta_r = K \Theta_r.$$

In what follows, we describe an appropriate process to compute the unique solution $X = X^T \succeq 0$, by avoiding an explicit computation of Θ_r or the solution X_Θ. Multiplying GARe (33) from the left and the right by the inverse of M_Θ, we get

$$M_\Theta^{-1} A_\Theta^T X_\Theta + X_\Theta A_\Theta M_\Theta^{-1} - X_\Theta B_\Theta B_\Theta^T X_\Theta + M_\Theta^{-1} C_\Theta^T C_\Theta M_\Theta^{-1} = 0.$$

Then we apply the extended block Arnoldi Algorithm 1 to the pair $(M_\Theta^{-1} A_\Theta^T, M_\Theta^{-1} C_\Theta^T)$. The same process described in Section 3 is followed here. We set again $\mathbb{V}_m = \Theta_r \mathcal{V}_m \in \mathbb{R}^{n_v \times 2mn_c}$ satisfying

$$\Pi^T \mathbb{V}_m = \Theta_r \Theta_l^T \mathbb{V}_m = \Theta_r \mathcal{V}_m = \mathbb{V}_m. \tag{35}$$

As we mentioned before, the orthonormal matrix $\mathbb{V}_m$ can be constructed using the Algorithm 2 without any explicit computation of $\mathcal{V}_m$ or Θ_r. After m iterations of the process, we can use Proposition 2 to prove that

$$M^{-1}\Pi A^T \mathbb{V}_m = \mathbb{V}_{m+1} \overline{\mathbb{T}}_m \tag{36a}$$

$$= \mathbb{V}_m \mathbb{T}_m + V_{m+1} T_{m+1,m} E_m^T. \tag{36b}$$

We seek for a low rank approximate solution to the GARe (34) under the following form

$$X_m = \mathbb{V}_m Y_m \mathbb{V}_m^T, \tag{37}$$

where $Y_m \in \mathbb{R}^{2mn_c \times 2mn_c}$ is the unique solution of a low-dimensional Riccati equation defined below. Replacing the approximation (37) in the equation (34) and multiplying from the left and right by the inverse of M, we obtain

$$M^{-1}\Pi A^T \mathbb{V}_m Y_m \mathbb{V}_m^T + \mathbb{V}_m Y_m \mathbb{V}_m^T A \Pi^T M^{-1} - \mathbb{V}_m Y_m \mathbb{V}_m^T B B^T \mathbb{V}_m Y_m \mathbb{V}_m^T + M^{-1}\Pi C^T C \Pi^T M^{-1} = 0,$$

which gives

$$\mathbb{T}_m Y_m + Y_m \mathbb{T}_m^T - Y_m \mathbb{V}_m^T B B^T \mathbb{V}_m Y_m + \mathbb{V}_m^T M^{-1} \Pi C^T C \Pi^T M^{-1} \mathbb{V}_m = 0.$$

When we apply the extended block Arnoldi Algorithm 1 to the pair $(M_\Theta^{-1} A_\Theta^T, M_\Theta^{-1} C_\Theta^T)$, we can notice that $M_\Theta^{-1} C_\Theta^T = \mathcal{V}_1^{(1)} \Lambda^{(1,1)}$ resulting from the use of `qr` function in Step 2 and then

$$\Theta_r M_\Theta^{-1} C_\Theta^T = \Theta_r \mathcal{V}_1^{(1)} \Lambda^{(1,1)},$$
$$\Theta_r M_\Theta^{-1} \Theta_r^T C^T = V_1^{(1)} \Lambda^{(1,1)}.$$

We already proved that $\Theta_r M_\Theta^{-1}\Theta_r^T = M^{-1}\Pi$, which gives

$$M^{-1}\Pi\, C^T = V_1^{(1)}\Lambda^{(1,1)}, \tag{38}$$

$$\mathbb{V}_m^T M^{-1}\Pi\, C^T = \mathbb{V}_m^T V_1^{(1)}\Lambda^{(1,1)} = \begin{bmatrix} I_{n_c} \\ 0_{n_c} \\ \vdots \\ 0_{n_c} \end{bmatrix} \Lambda^{(1,1)}. \tag{39}$$

Finally, we obtain the following low-dimensional Riccati equation

$$\mathbb{T}_m Y_m + Y_m \mathbb{T}_m^T - Y_m \mathbb{V}_m^T BB^T \mathbb{V}_m Y_m + \mathbb{V}_m^T V_1^{(1)}\Lambda^{(1,1)} C^T (\mathbb{V}_m^T V_1^{(1)}\Lambda^{(1,1)} C)^T = 0, \tag{40}$$

which is solved by a direct method such as `care` in MATLAB.

Let $R(X_m)$ be the residual corresponding to the approximation X_m given by

$$\begin{aligned} R(X_m) &= M^{-1}\Pi A^T X_m + X_m A\Pi^T M^{-1} && \text{(41a)} \\ &- X_m BB^T X_m + M^{-1}\Pi C^T C\Pi^T M^{-1}. && \text{(41b)} \end{aligned}$$

In order to stop the iterations, we need to compute the residual $R(X_m)$ given by (41) without involving X_m, since it becomes expensive as m increases. The next result shows how to compute the residual norm of $R(X_m)$ without involving the approximate solution, which is given only in a factored form at the end of the process.

Theorem 1. *Let $\mathbb{V}_m \in \mathbb{R}^{2mn_c \times 2mn_c}$ be an orthonormal matrix generated by Algorithm2. Let $X_m = \mathbb{V}_m Y_m \mathbb{V}_m^T$ be the approximate solution of the GARe (34), then the residual norm is given by*

$$\|R(X_m)\| = \|T_{m+1,m} E_m^T Y_m\|, \tag{42}$$

where $E_m = [0_{2n_c \times 2(m-1)n_c}, I_{2n_c}]^T$ and $\|\|$ is the abbreviation of $\|\|_2$.

Proof 3. *According to (36) and (41), we have*

$$\begin{aligned} R(X_m) &= M^{-1}\Pi A^T X_m + X_m A\Pi^T M^{-1} - X_m BB^T X_m + M^{-1}\Pi C^T C\Pi^T M^{-1} \\ &= M^{-1}\Pi A^T \mathbb{V}_m Y_m \mathbb{V}_m^T + \mathbb{V}_m Y_m \mathbb{V}_m^T A\Pi^T M^{-1} - \mathbb{V}_m Y_m \mathbb{V}_m^T BB^T \mathbb{V}_m Y_m \mathbb{V}_m^T + M^{-1}\Pi C^T C\Pi^T M^{-1} \\ &= (\mathbb{V}_m \mathbb{T}_m + V_{m+1} T_{m+1,m} E_m^T) Y_m \mathbb{V}_m^T + \mathbb{V}_m Y_m (\mathbb{T}_m^T \mathbb{V}_m^T + E_m T_{m+1,m}^T V_{m+1}^T) \\ &- \mathbb{V}_m Y_m \mathbb{V}_m^T BB^T \mathbb{V}_m Y_m \mathbb{V}_m^T + M^{-1}\Pi C^T C\Pi^T M^{-1}. \end{aligned}$$

Using the fact that $M^{-1}\Pi\, C = V_1^{(1)}\Lambda^{(1,1)}$ as it is described in (38), we get

$$R(X_m) = [\mathbb{V}_m, V_{m+1}] \begin{bmatrix} \mathbb{T}_m Y_m + Y_m \mathbb{T}_m^T + E_1\Lambda^{(1,1)}(E_1\Lambda^{(1,1)})^T & (T_{m+1,m} E_m^T Y_m)^T \\ T_{m+1,m} E_m^T Y_m & 0 \end{bmatrix} \begin{bmatrix} \mathbb{V}_m \\ V_{m+1} \end{bmatrix}.$$

Since Y_m is the symmetric solution of the low-dimensional Riccati equation, then

$$R(X_m) \quad = \quad \mathbb{V}_{m+1} \begin{bmatrix} 0 & (T_{m+1,m}E_m^T Y_m)^T \\ T_{m+1,m}E_m^T Y_m & 0 \end{bmatrix} \mathbb{V}_{m+1}^T,$$

and finally we get the desired result

$$\|R(X_m)\| = \|T_{m+1,m}E_m^T Y_m\|. \tag{43}$$

We can check weather we get the desired convergence by verifying the test $\|R(X_m)\| < \epsilon$. Fortunately, the residual $\|R(X_m)\|$ can be computed in a suitable way as described in the theorem above, without computing the approximate solution X_m. We take the advantage of X_m as a symmetric positive semi-definite, so it can be decomposed into a product of two matrices of low-rank as $X_m = ZZ^T$, where Z is a matrix of rank less than or equal to $2m$. The benefit from this decomposition is that we just need to store Z in order to compute the approximate solution X_m. Let $Y_m = U\Sigma V$, the SVD decomposition of Y_m where Σ is the matrix of the singular values of Y_m sorted in decreasing order. Let `dtol` some tolerance and define U_r, V_r as the first r columns respectively of U and V corresponding to the r singular values of magnitude greater than `dtol`. In the numerical experiments, we set dtol=10^{-12}. Setting $\Sigma_r = [\sigma_1, \cdots, \sigma_r]$, we get $Y_m \approx U_r \Sigma_r V_r^T$, and it follows that

$$X_m \approx Z_m Z_m^T, \tag{44}$$

with $Z_m = \mathbb{V}_m U_r (\Sigma_r)^{1/2}$.

The iterations were stopped when the relative residual norm was less than $tol = 10^{-8}$

$$\frac{\|R(X_m)\|}{\|M^{-1}\Pi C^T C \Pi^T M^{-1}\|} < 10^{-8}. \tag{45}$$

We mentioned before that all our results are obtained without any explicit computation of Π, so to compute $M^{-1}\Pi C^T$ in an appropriate manner we use the formula (38), and then

$$M^{-1}\Pi C^T C \Pi^T M^{-1} = V_1^{(1)} \Lambda^{(1,1)} (V_1^{(1)} \Lambda^{(1,1)})^T, \tag{46}$$

where $V_1^{(1)}$ is the first n_c block-column of $V_1 \in \mathbb{R}^{n_v \times 2n_c}$ and $\Lambda^{(1,1)} \in \mathbb{R}^{n_c \times n_c}$ is the block $(1,1)$ of the upper triangular matrix $\Lambda \in \mathbb{R}^{2n_c \times 2n_c}$ previously described in (25). The following algorithm summarizes all the results explained above.

Algorithm 3 Extended block Arnoldi Riccati algorithm (EBARA)

- Inputs: M, $A \in \mathbb{R}^{n_v \times n_v}$, $G \in \mathbb{R}^{n_v \times n_p}$, $B \in \mathbb{R}^{n_v \times n_b}$, $C \in \mathbb{R}^{n_c \times n_v}$, tolerance ϵ, dtol, number of iteration m_{max}.
- Outputs: the approximate solution $X_m \approx Z_m Z_m^T$.
- For $m = 1, \cdots, m_{max}$
- Use Algorithm 2 to compute $\mathbb{V}_m$ an orthonormal matrix and compute $\mathbb{T}_m$ the block Hessenberg matrix.
- Solve the low-dimensional Riccati equation (40) using the MATLAB function `care`.
- Compute the relative residual norm (45) using (43) and (46), and if it is less than ϵ, then
  1. Compute the SVD of $Y_m = U\Sigma V$ where $\Sigma = diag[\sigma_1, \cdots, \sigma_{2m}]$.
  2. Determine r such that $\sigma_{r+1} < \text{dtol} \leq \sigma_r$, set $\Sigma_r = diag[\sigma_1, \cdots, \sigma_r]$ and compute $Z_m = \mathbb{V}_m U_r (\Sigma_r)^{1/2}$.

  end if.
- End For

5. Numerical experiments

In this section, we present some numerical results to confirm the performance of the proposed approaches. All the experiments were carried out using MATLAB R2018a on a computer with Intel ® core i7 at 2.3GHz and 8Gb of RAM. The MATLAB programs representing the two algorithms (Algorithm 2, Algorithm 3) are available in `https://lmpa.univ-littoral.fr/index.php?page_id=8`. Our method is applied to a discretized Navier-Stokes equations as described in Section 2. We first show how our method allows us to build an efficient reduced model by presenting the transfer functions of the original and reduced systems with the associated error. Then we investigate the numerical solution of the GARe (34) using Algorithm 3 and as we mentioned earlier this numerical solution is actually the key to construct the matrix feedback used to stabilize the unstable system. All the data was provided from [9]. Some information on this data are depicted in Table 1. The state dimension n_v refers to the dimension of the discretized velocity field, and n_p is the dimension of the discretized pressure field, also sparsity of each matrix A and M is given, i.e., the ratio of the number of non-zero elements to the total number of elements in the matrix. We used different dimensions of n_v and n_p

corresponding to three levels. Note that the norm used here is the $\mathcal{H}_\infty$ norm and it expressed

Table 1: The matrix dimensions for different levels

Level	n_v	n_p	full model $(n_v + n_p)$	sparsity of A & M
1	4796	672	5468	$4.6 \cdot 10^{-3}$ | $2.3 \cdot 10^{-3}$
2	12292	1650	13942	$1.8 \cdot 10^{-3}$ | $9.05 \cdot 10^{-4}$
3	28914	3784	32698	$7.79 \cdot 10^{-4}$ | $3.89 \cdot 10^{-4}$

as $\|F - F_m\|_\infty = \sup_{\omega \in \mathbb{R}} \|F(j\omega) - F_m(j\omega)\|_2$. To compute this norm we use the following functions from `lyapack` [33]

1. `lp_lgfrq` : Generates a set of logarithmically distributed frequency sampling points $\omega \in [10^{-5}, 10^5]$.
2. `lp_gnorm` : Computes a vector which contains the 2-norm $\|F - F_m\| = \sigma_{max}(F(i\omega) - F_m(i\omega))$ for each sampling points $\omega \in [10^{-5}, 10^5]$ and $i = \sqrt{-1}$.

Example 1 For this example, we show the frequency response of the original and reduced systems. We considered the three models from Table 1 where we associate level 1 with Reynolds number Re = 300, level 2 with Re = 400 and level 3 with Re = 500. For the three models we used $m = 120$ and then the dimension of the reduced system is $2 \times m \times n_b = 480$. For a Reynolds number Re < 100, Navier-Stokes flow starts to behave like a Stokes flow, and this comes from the fact that the convection term $(z \cdot \nabla)z$ in (1) doesn't have an important impact. Figures 2, 3 and 4 illustrate the obtained results comparing the original transfer function and its approximation for the three levels. We also plotted the error-norm between the two transfer functions. The computed error norm $\|F - F_m\|_\infty$ was $1.37 \cdot 10^{-5}$ for level 1, $\|F - F_m\|_\infty = 9.82 \cdot 10^{-5}$ for level 2 and $\|F - F_m\|_\infty = 6.5 \cdot 10^{-4}$ for level 3.

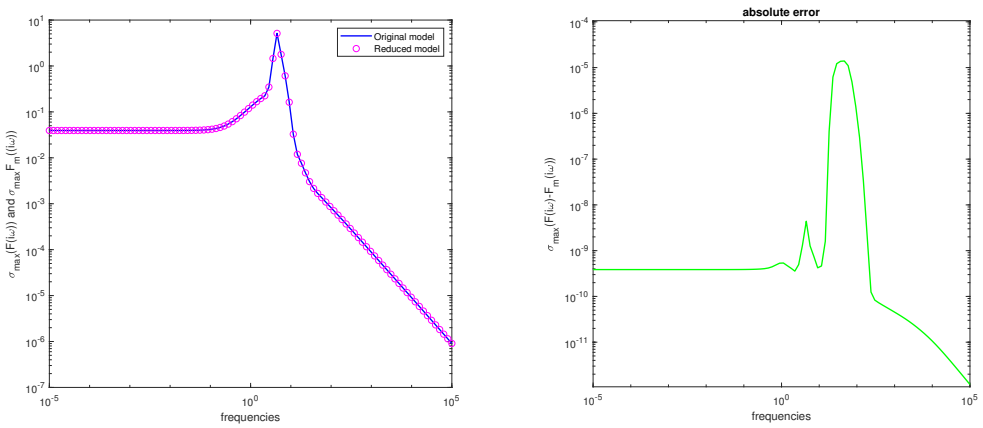

Figure 2: Level 1 with Re= 300: Bode plot (left) and the error norms versus frequencies (right).

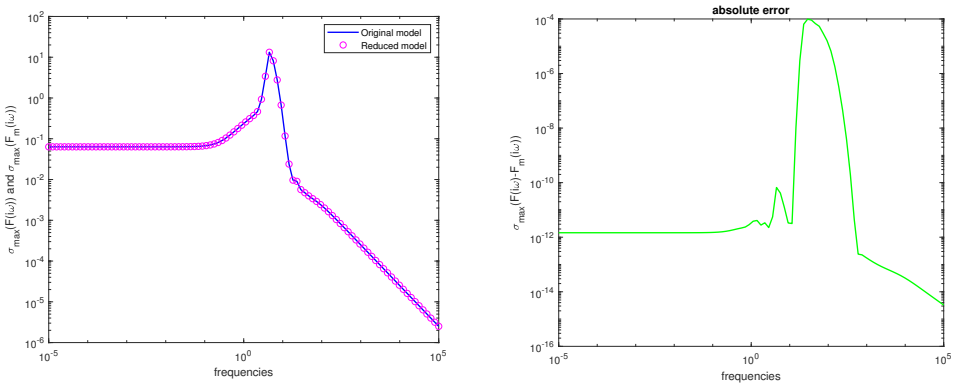

Figure 3: Level 2 with Re= 400: Bode plot (left) and the error norms versus frequencies (right).

Example 2 To investigate the efficiency of Algorithm 2, we compare our method to a common and deployed model order reduction method knows as the Balanced Truncation (BT). The main challenge in the BT is to solve larges-scale Lyapunov equations in order to obtain the system Gramians that will be used to generate a reduced model. The BT algorithm is available at the M-M.E.S.S. toolbox, see [34]. The authors used a different data from those presented in Table 1. We chose from their data two level of discretization and we summarize in Table 2 some information. For the level 1 we used $m = 70$ and $m = 75$ for level 2. The tolerance truncation is set to 10^{-5}. In Figure 5, we plotted the norms $\|F(j\omega)\|_2$ and its approximation $\|F_m(j\omega)\|_2$ for different values of the frequency $\omega \in [10^{-5}, 10^5]$ of the two methods (our method and BT). As can be seen, we have obtained a perfect match between the original transfer function and its approximation for both methods. We show the obtained error-norms $\|F(j\omega) - F_m(j\omega)\|_2 = \sigma_{max}(F(j\omega) - F_m(j\omega))$ for different values of the frequency ω with the Reynold number Re=300

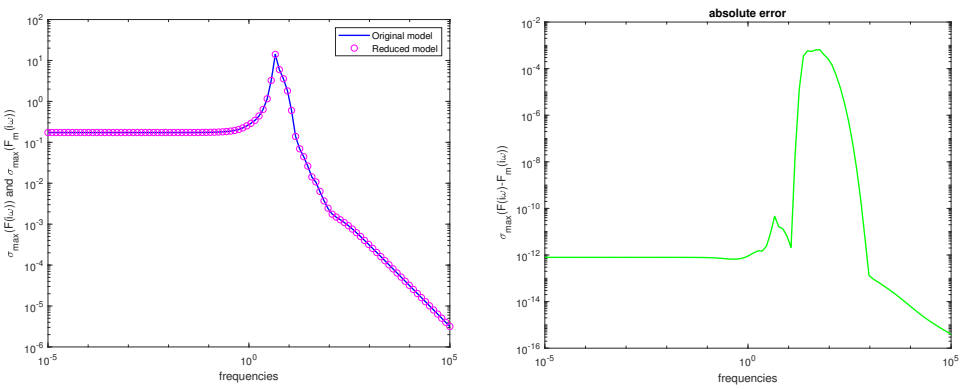

Figure 4: Level 3 with Re= 500: Bode plot (left) and the error norms versus frequencies (right).

Table 2: The matrix dimensions for different levels

Level	n_v	n_p	full model $(n_v + n_p)$
1	3142	453	3595
2	8268	1123	9391

in Figure 6 and with Re=400 in Figure 7. Here, you can notice that the error of our Algorithm increases rapidly when $\omega \in (1, 10^3)$, we tried to alleviate this problem by increasing the number of iterations "m", but this choice increased the computing time and also increased the error when $\omega \in (10^{-5}, 1)$ from 10^{-10} to 10^{-4}, and this also applies when $\omega \in (10^3, 10^5)$. This is why we stick with the first choice and do not increase the number of iterations 'm'. We present in Table 3 the execution time of our algorithm and that based on BT described in [34].

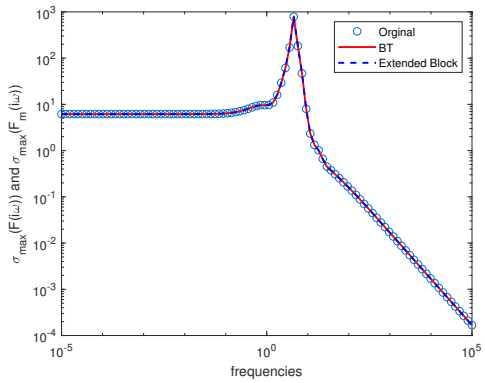

Figure 5: Level 2 with Re= 300: Bode plot

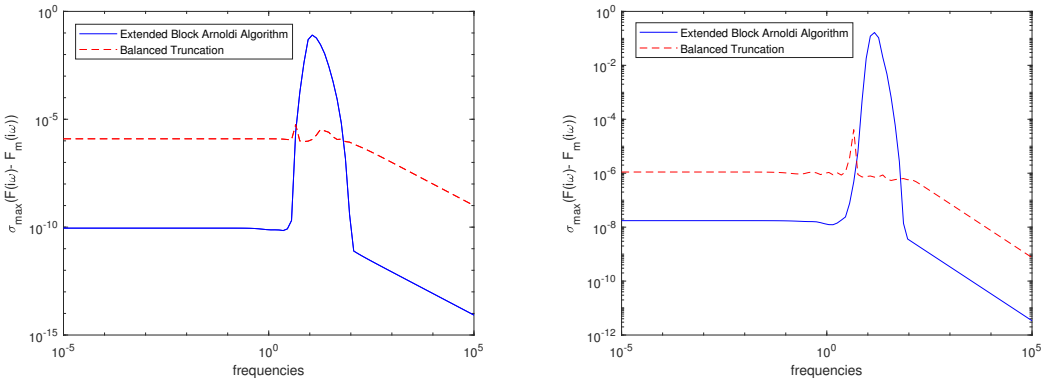

Figure 6: The error-norms versus frequencies using level 1 with Re=300 (left) and with Re=400 (right).

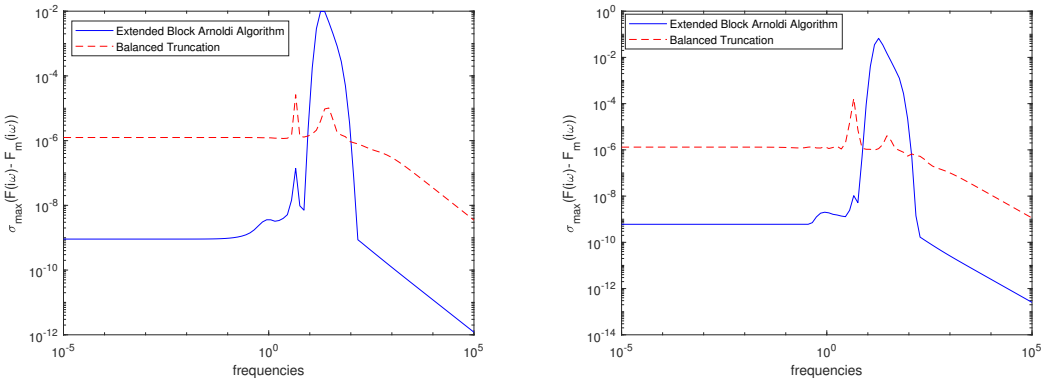

Figure 7: The error-norms versus frequencies using level 2 with Re=300 (left) and with Re=400 (right).

Table 3: The CPU-time (in seconds) required for both methods

Reynolds number	Re=300		Re=400	
	Algorithm 2	BT	Algorithm 2	BT
Level 1	4.55	13.92	4.17	15.47
Level 2	19.71	51.09	17.78	56.34

Example 3 In this example, we investigate the extended block Arnoldi Riccati algorithm (EBARA, Algorithm 3) for solving generalized algebraic Riccati equations (GARe) (34) which is needed to compute the matrix feedback of our initial problem. We use matrices corresponding to the level 1 of discretization in Table 1 with different Reynolds numbers Re $= 300, 400$ and 500. We have established a comparison between our Algorithm 3 and Algorithm 2 (a generalized low-rank Cholesky factor Newton method) described in [9]. We have summarized in Table 5

the number of iterations, ADI and Newton iterations as well as the cpu-time needed to reach the convergence of both methods

Table 4: The obtained results of both methods

Methods	Algorithm 3			Algorithm 2 in [9]		
	# of iter.	cpu-time(sec)	Rel. res.	Newton & ADI iter.	CPU-time(sec)	Rel. res.
Re=300	77	136.77	$8.43e^{-08}$	8 & 245	323.48	$9.10e^{-09}$
Re=400	94	378.82	$8.36e^{-08}$	20 & 301	1124.43sec	$2.27e^{-07}$
Re=500	109	524.27	$8.14e^{-08}$	-	>1800	-

Stabilizing the unstable system

We recall here the matrix feedback K required to stabilize our original system (5). The control vector is given by

$$\mathbf{u}(t) = -K\mathbf{v}(t) \quad \text{where} \quad K = B^T X_m M.$$

The matrix X_m is the approximate solution to GARe (34). We use the relation (44) that allows us to store X_m in a efficient way, then the feedback matrix has the following form

$$K = B^T Z_m \, Z_m^T M.$$

The Reynolds number chosen here Re = 400 and 500 makes our original system (4) unstable as we mentioned earlier. We plug in the input $\mathbf{u}(t)$ in the unstable original system (4) to get the stabilized system described as follows

$$M \frac{d}{dt}\mathbf{v}(t) = (A - BK)\mathbf{v}(t) + G\mathbf{p}(t), \tag{47a}$$

$$0 = G^T \mathbf{v}(t), \tag{47b}$$

$$y = C\,\mathbf{v}(t). \tag{47c}$$

To show the effectiveness of the constructed feedback matrix K, we establish a time domain response simulation. In all examples (before and after stabilization) we use the same constant unit as input actuation. We use matrices corresponding to level 1 of discretization in Table 1 and we set $m = 120$. For each Re = 400 and 500, we first present the time domain response of

the original and reduced systems and then we plot the time domain response associated with the stabilized system (47) and its reduced one. It is important to notice that while we perform the reduction process to the stabilized system (47), using the extended block Krylov subspace method described in Section 3, we have to solve at each iteration the following saddle point problem

$$\underbrace{\begin{bmatrix} A - BK & G \\ G^T & 0 \end{bmatrix}}_{\widehat{\mathbf{A}}} \begin{bmatrix} w \\ \star \end{bmatrix} = \begin{bmatrix} z \\ 0 \end{bmatrix}, \tag{48}$$

Notice that the product BK is dense and this in fact what makes the block $(1,1)$ of $\widehat{\mathbf{A}}$ dense too. To avoid this problem of density that can make our computation infeasible, we rewrite the saddle point problem (48) in a low-rank form

$$\left(\underbrace{\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix}}_{\mathbf{A}} - \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} K & 0 \end{bmatrix}}_{\mathbf{K}} \right) \begin{bmatrix} w \\ \star \end{bmatrix} = \begin{bmatrix} z \\ 0 \end{bmatrix},$$

and then we use the *Sherman-Morrison-Woodbury formula* [35]

$$(\mathbf{A} - \mathbf{BK})^{-1} = \mathbf{A^{-1}} + \mathbf{A^{-1}B}(\mathbf{I} - \mathbf{KA^{-1}B})^{-1}\mathbf{K}\,\mathbf{A^{-1}}.$$

Besides solving the small dense matrix $(\mathbf{I_{n_b}} - \mathbf{KA^{-1}B})$ with right hand side $\mathbf{K}$ we need to solve $\mathbf{A^{-1}B}$ and $\mathbf{A}$ with the right hand side $[z,\, 0]^T$, and this can be done easily by adding the n_b columns $\mathbf{B}$ to the matrix $[z,\, 0]^T$, and then instead of solving the problem (48) with $\widehat{\mathbf{A}}$ one can solve the following saddle point problem

$$\begin{bmatrix} A & G \\ G^T & 0 \end{bmatrix} \begin{bmatrix} w \\ \star \end{bmatrix} = \begin{bmatrix} z & B \\ 0 & 0 \end{bmatrix},$$

using "\", a built-in MATLAB function.

In Figures 8 and 11, we can see that for both cases Re = 400 and 500, the time domain simulation of the original and reduced systems show stability and a good accuracy of the reduced output compared to the original one. However, after $t = 30s$ some oscillations appear due to the instability of our original system. We can also see from the right parts of Figure 8 and Figure 11 that the error-norm $\|y - y_m\|$ increases as the time increases and this is due to the fact that our reduced system loses its accuracy caused by the instability that characterizes the original system. The performance of the matrix feedback allows us to stabilize the unstable system. This is shown in Figures 9, 10, 12 and 13 using two different Reynolds number Re = 400 and

Re = 500. In the left side of these figures we display the time domain responses of the original and reduced stabilized systems of 1st input to 1st output in Figures 9 and 12 with Re = 400 and Re = 500 respectively, and also of 2nd input to 2nd output in Figures 10 and 13 with Re = 400 and Re = 500 respectively. One can notice that the figures illustrate a good accuracy of the reduced output compared to the original one. Moreover, it can be seen that after few oscillations that end in $t = 5s$, the output of the stabilized system stabilize at constant values. On the right hand side of Figures 9, 10, 12 and 13, we show the error in the outputs for the same inputs and we notice that after the stabilization, the error $\|y - y_m\|$ does not increase as the time increases which was not the case before stabilization. This proves the accuracy of our method of constructing a feedback matrix for stabilization using the EBARA Algorithm 3.

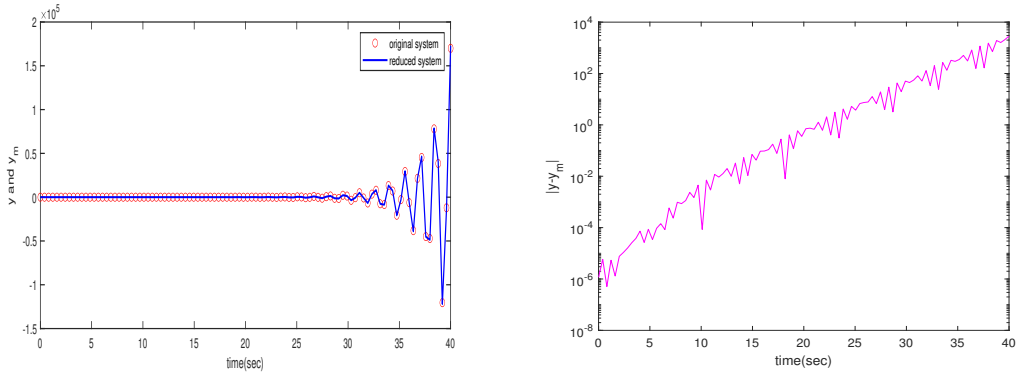

Figure 8: Left: time domain response simulation of the unstable system with Re=400. Right: the error norm $\|y - y_m\|$.

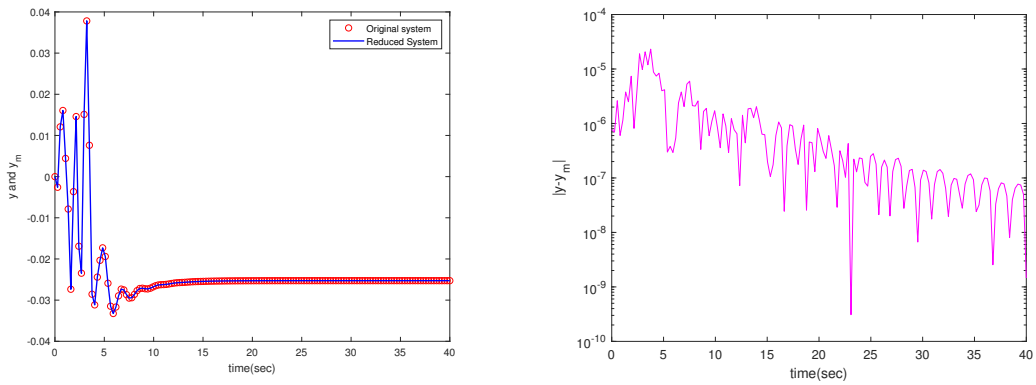

Figure 9: Left: time domain response for the stabilized system (input 1 to output 1) original and reduced systems. Right: the error norm $\|y - y_m\|$.

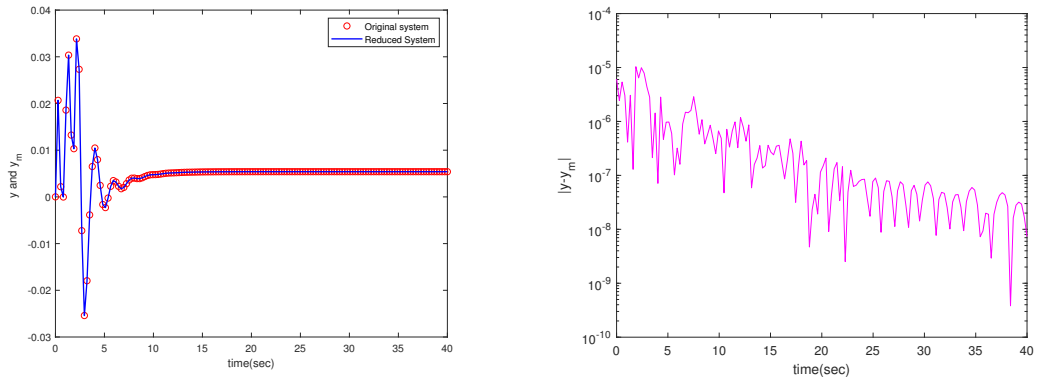

Figure 10: Left: time domain response for the stabilized system (input 2 to output 2) original and reduced systems. Right: the error norm $\|y - y_m\|$.

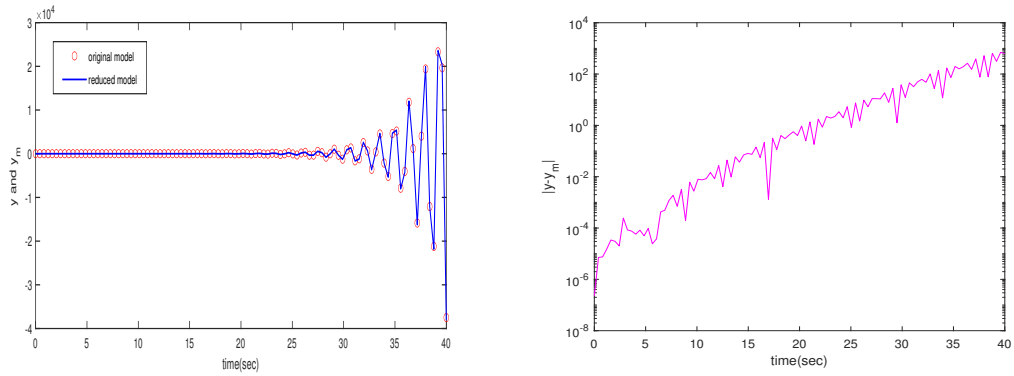

Figure 11: Left: time domain simulation for the unstable system with Re=500. Right: the error norm $\|y - y_m\|$.

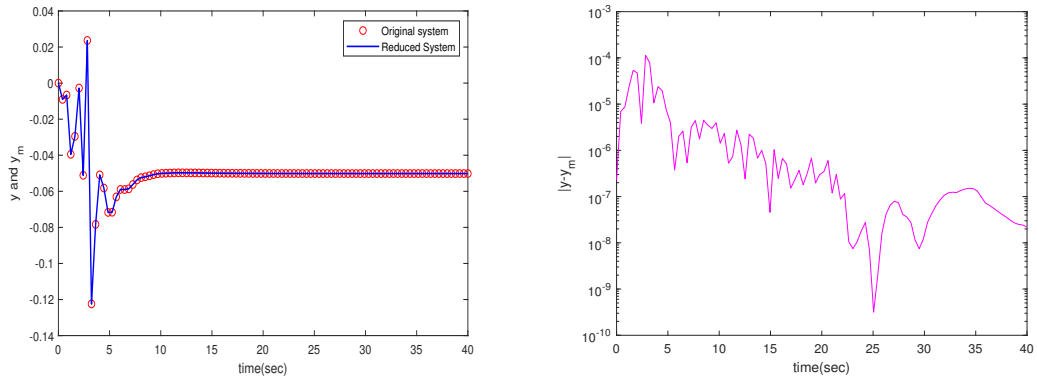

Figure 12: Left: time domain response for the stabilized system (input 1 to output 1) of original and reduced systems. Right: the error norm $\|y - y_m\|$.

Conclusion

Navier-stokes equations (NSEs) are considered as the pillars of fluid mechanics. A spatial discretization of the linearized NSEs around a steady state leads to a high dimension descriptor

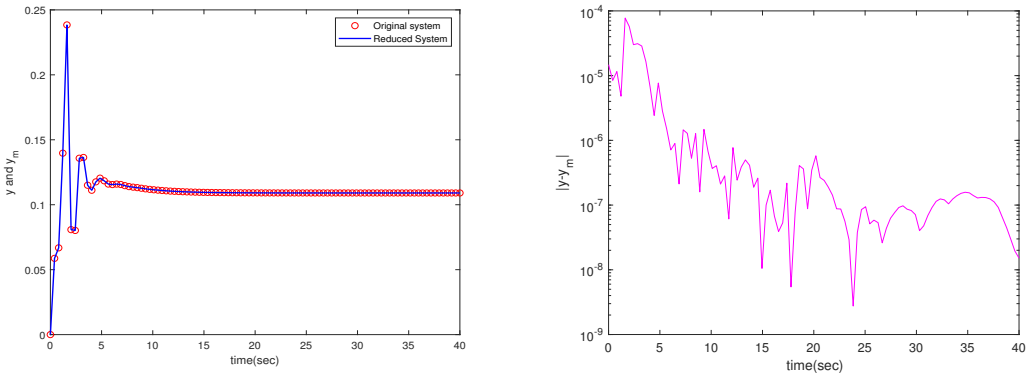

Figure 13: Left: time domain response for the stabilized system (input 2 to output 2) of original and reduced systems. Right: the error norm $\|y - y_m\|$.

system of index-2 presented by a set of differential algebraic equations (DAEs). In this paper, we proposed a projection Krylov-based method to reduce this large dimension system. Our proposed method is based essentially on an extended block Arnoldi algorithm that allows us to build an efficient reduced system with a reasonable cost of computations. The system of NSEs lost its stability when Reynolds numbers are large and then we need stabilization techniques. One of the methods for stabilization that we used here is by solving an LQR problem based on a Riccati feedback approach. We suggested an extended Krylov-based method to solve the obtained large-scale algebraic Riccati equation and the obtained numerical solution is the key to design a controller described by a feedback matrix.

References

[1] M. Badra, Feedback stabilization of the 2-d and 3-d navier-stokes equations based on an extended system, ESAIM. COCV., 15 (2009) 934–968.

[2] M. Badra, Lyapunov function and local feedback boundary stabilization of the navier-stokes equations, SIAM J. Cont. Opti., 48 (2009) 1797–1830.

[3] V. Barbu, Feedback stabilization of the navier-stokes equations, ESAIM. COCV., 9 (2003) 197–205.

[4] V. Barbu, I. Lasiecka, R. Triggiani, Tangential boundary stabilization of navier-stokes equations, American Mathematical Society, (2006).

[5] A. V. Fursikov, Stabilization for the 3d navier-stokes system by feedback boundary control, Disc. Contin. Dyna. Syst., 10(1-2) (2004) 289–314.

[6] J. P. Raymond, Feedback boundary stabilization of the two-dimensional navier-stokes equations, SIAM J. Cont. Opti., 45(3) (2006) 790–828.

[7] J. P. Raymond, Feedback boundary stabilization of the three-dimensional incombressible navier-stokes equations, J. Math. Pures Appl., 87(6) (2007) 627–669.

[8] J. P. Raymond, Stokes and navier-stokes equations with nonhomogeneous boundary conditions, Annales de l'I.H.P Analyse non linéaire, 24(6) (2007) 921–951.

[9] E. Bansch, P. Benner, J. Saak, H. K. Weichelt, Riccati-based boundary feedback stabilization of incompressible navier-stokes flows, SIAM J. Sci. Comput., 37(2) (2015) A832–A858.

[10] E. Bansch, P. Benner, Stabilization of incompressible flow problems by riccati-based feedback, in constrained optimization and optimal control for partial differential equations, Internat. Ser. Numer. Math., 160 (2012) 5–20.

[11] P. Benner, J. Saak, M. M. Uddin, Balancing based model reduction for structured index-2 unstable descriptor systems with application to flow control, Numer. Alge. Cont. Opti., 6(1) (2016) 1–20.

[12] H. Barkouki, A. H. Bentbib, K. Jbilou, An extended nonsymmetric block lanczos method for model reduction in large scale dynamical systems, Calcolo, 55(1) (2018) 1–23.

[13] M. Frangos, I. M. Jaimoukha, Adaptive rational krylov algorithms for model reduction, In Proceedings of the European Control Conference, (2007) 4179–4186.

[14] E. Grimme, Krylov projection methods for model reduction, Ph.D. thesis, Coordinated Science Laboratory, University of Illinois at Urbana Champaign, (1997).

[15] M. A. Hamadi, K. Jbilou, A. Ratnani, Model reduction method in large scale dynamical systems using an extended-rational block arnoldi method, J. Appl. Math. Comput., (2021).

[16] M. Heyouni, K. Jbilou, A. Messaoudi, T. Tabaa, Model reduction in large scale mimo dynamical systems via the block lanczos method, Comput. Appl. Math., 27(2) (2008) 211–236.

[17] K. Jbilou, A survey of krylov-based methods for model reduction in large-scale mimo dynamical systems, Appl. Comput. Math., 15(2) (2016) 117–147.

[18] L. Knizhnerman, D. Druskin, M. Zaslavsky, On optimal convergence rate of the rational krylov subspace reduction for electromagnetic problems in unbounded domains, SIAM J. Numer. Anal., 47(2) (2009) 953–971.

[19] S. Gugercin, A. C. Antoulas, A survey of model reduction by balanced truncation and some new results, Internat. J. Control, 77(8) (2003) 748–766.

[20] V. Mehrmann, T. Stykel, Balanced truncation model reduction for large-scale systems in descriptor form, Lecture Notes in Computational Science and Engineering, 45 (2005) 83–115.

[21] B. C. Moore, Principal component analysis in linear systems: controllability, observability and model reduction, IEEE Trans. Auto. Cont., AC-26 (1981) 17–32.

[22] P. Benner, P. Goyal, J. Heiland, I. P. Duff, Operator inference and physics-informed learning of low-dimensional models for incompressible flows, arXiv:2010.06701v1, (2020).

[23] M. Heinkenschloss, D. C. Sorensen, K. Sun, Balanced truncation model reduction for a class of descriptor systems with application to the oseen equations, SIAM J. Sci. Comput., 30(2) (2008) 1038–1063.

[24] P. Benner, Z. Bujanović, P. Kürschner, J. Saak, Radi: a low-rank adi-type algorithm for large scale algebraic riccati equations, Numer. Math. 138 (2018) 301–330.

[25] M. Heyouni, K. Jbilou, An extended block arnoldi method for large matrix riccati equations, Elect. Trans. Numer. Anal., 33 (2009) 53–62.

[26] V. Simoncini, D. B. Szyld, M. Marlliny, On two numerical methods for the solution of large-scale algebraic riccati equations, IMA Journal of Numerical Analysis 34 (2014) 904–920.

[27] S. Gugercin, T. Stykel, S. Wyatt, Model reduction of descriptor systems by interpolatory projection methods, SIAM J. Sci. Comput., 35(5) (2013) B1010–B1033.

[28] A. Chkifa, M. A. Hamadi, K. Jbilou, A. Ratnani, A computational method for model reduction in index-2 dynamical systems for stokes equations, Comput. Math. with Appl., 99 (2021) 171–181.

[29] K. A. Cliffe, T. J. Garratt, A. Spence, Eigenvalues of block matrices arising from problems in fluid mechanics, SIAM J. Matrix Appl., 15 (1994) 1310–1318.

[30] U. M. Ascher, L. R. Petzold, Computer methods for ordinary differential equations and differential-algebraic equations, SIAM, (1998).

[31] E. Eich-Soellner, C. Fuhrer, Numerical methods in multibody dynamics, European Consortium for Mathematics in Industry, B. G. Teubner GmbH, (1998).

[32] V. Simoncini, A new iterative method for solving large-scale lyapunov matrix equations, SIAM J. Sci. Comput., 29(3) (2007) 1268–1288.

[33] V. Mehrmann, T. Penzl, Benchmark collections in slicot, Technical Report SLWN1998-5, SLICOT Working Note, ESAT, KU Leuven, K. Mercierlaan 94, Leuven-Heverlee 3100, Belgium, (1998).

[34] J. Saak, M. Kohler, P. Benner, M-m.e.s.s. -1.0.1- the matrix equations sparse solvers library (2016). doi:10.5281/zenodo.50575.

[35] G. H. Golub, C. F. van Loan, Matrix computations, Johns Hopkins University Press, Baltimore, (1996).