Max Planck Institute Magdeburg
Preprints

Sergey Dolgov Martin Stoll

Low-rank solutions to an optimization problem constrained by the Navier-Stokes equations

MPIMD/15-15 **September 18, 2015**

Abstract

The numerical solution of PDE-constrained optimization problems subject to the non-stationary Navier-Stokes equation is a challenging task. While space-time approaches often show favorable convergence properties they often suffer from storage problems. We here propose to approximate the solution to the optimization problem in a low-rank from, which is similar to the Model Order Reduction (MOR) approach. However, in contrast to classical MOR schemes we do not compress the full solution at the end of the algorithm but start our algorithm with low-rank data and maintain this form throughout the iteration. Theoretical results and numerical experiments indicate that this approach reduces the computational costs by two orders of magnitude.

Imprint:

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg

Publisher:
Max Planck Institute for
Dynamics of Complex Technical Systems

Address:
Max Planck Institute for
Dynamics of Complex Technical Systems
Sandtorstr. 1
39106 Magdeburg

http://www.mpi-magdeburg.mpg.de/preprints/

1 Introduction

Optimization subject to constraints given by partial differential equations (PDEs) is an active field of research [54, 27]. Much progress has been made over the last years concerning the analysis of problems and the development of efficient numerical schemes to solve the linear and nonlinear optimization problems. In almost all scenarios one arrives at the solution of a large scale linear system that either represents the first order (KKT) conditions [41] or is part of some nonlinear scheme such as an SQP or interior point approach [24, 55].

The development of iterative solvers and especially preconditioners for these linear systems, which often have a saddle point form, has been a key research area in numerical linear algebra [14, 39, 21]. For parabolic problems space-time methods have shown great promise [48, 52] regarding robustness with respect to dependence on both mesh- and regularization parameters. Multigrid methods [9] are also used for parabolic problems and also methods using stationary iterations [28].

Our approach in this paper follows a recent work presented in [51] where the space-time system is solved using a low-rank decomposition. This schemes replaces the space-time solution, which can be written as a matrix, by an approximation that only needs little information coming from the space and time domains. We will make this more precise in Section 2.1. The goal is to reduce the storage amount to a small multiple of that of the stationary problem. The work in [51] only considered linear problems and we here present how this approach can be carried over to the case when we consider the optimization subject to the Navier-Stokes equations. The control of the Navier-Stokes equations has been an active research topic for the last years and we refer to [22, 17] and the references given therein.

Low-rank approximations have become state-of-the-art methods for data-sparse solution of high-dimensional problem [32, 16, 18]. This approach is based on the idea of separation of variables, and approximates a large multidimensional array (tensor) by a polylinear combination of smaller tensors. Some of the most powerful of such combinations are the Tensor Train (TT) [44] and Hierarchical Tucker (HT) [20] formats. A tensor can be associated with a discretized operator or solution of a high-dimensional PDE. To solve an equation, one needs an iterative algorithm, since elements of an initial (large) tensor are never accessed directly. One can adapt classical algorithms such as GMRES [3, 26], or develop tailored methods for low-rank formats, such as the Alternating Least Squares [25].

TT and HT decompositions are based on a recurrent application of the matrix low-rank factorization, for example, the Singular Value Decomposition. Therefore, their storage efficiency depends on the ranks of the corresponding matricizations of a tensor. If all ranks are bounded by a moderate value for the given data, the storage needed for a low-rank format is logarithmic compared to the cardinality of the initial tensor. In an ultimate scenario, one can reshape any data to a tensor, the dimensions of which are prescribed by the smallest (prime) factors of the number of elements. The TT decomposition applied to such tensor was called the Quantized Tensor Train (QTT) format [31, 43], and it has demonstrated impressive compression properties for smooth solutions of PDEs, discretized on tensor product grids.

However, if a PDE is posed on a complicated spatial domain, an ultimate tensorisation is inapplicable. Nonetheless, there can still be several independent variables, such as an aggregated spatial coordinate, time and auxiliary parameters. In this paper we consider separation of space and time variables only. In this case, the solution is reshaped to a matrix, and the classical low-rank decomposition is sought. There exist many efficient algorithms for solution of matrix equations in the low-rank form, such as the ADI and Krylov methods for Sylvester equations [56, 4] (including high-dimensional generalizations [34, 38]). However, they often require commutativity of (at least, dominant) low-rank factors of the operator. Alternating tensor algorithms can be more efficient for problems such as the Navier-Stokes equations, where the operator has a complicated structure.

Development of low-rank methods for nonlinear problems was performed in different communities in different ways. On one hand, the Proper Orthogonalized Decomposition (POD) is a mature technique in Model Order Reduction, and it was used intensively for reducing the Navier-Stokes equations, see e.g. [10, 2, 40]. However, the POD requires to solve the full problem first, which might be extremely computationally demanding. On the other hand, tensor methods compute directly the low-rank factors, but they were applied to only a few nonlinear problems. One can mention the Hartree-Fock equation [30, 49] and some plasma models [12, 33]. More developed are methods for Ricatti equations [6], but they rely on the form of the operator explicitly.

In this paper we generalize the Alternating Least Squares algorithm to the saddle-point structure of the optimality system, arising from the Lagrangian optimization, and adapt them particularly to the Navier-Stokes equations in constraints. We compare them with the traditional space-time optimization with the state-of-the-art preconditioners [48] and show that the new algorithm provides a significant reduction of computational time and storage.

Our paper is structured as follows. We first introduce the problem formulation for both the Navier-Stokes forward problem and the corresponding optimization problem. In Section 2.1 we discuss the forward formulation and introduce an appropriate low-rank formulation. This is followed by Section 2.2 where a low-rank formulation for the optimization problem is developed. We also show that this can be used with various time-discretization schemes. In Section 3.1 we propose an alternating linear scheme (ALS) for the forward simulation which in Section 3.2 is followed by a detailed discussion of such an ALS method for the optimality system that sits at the heart of the outer nonlinear Picard iteration. The particular case of carefully handling the pressure degrees of freedom needed for the ALS method is discussed in Section 3.3. We propose efficient solvers for the linear systems in saddle point form in Section 3.4. A discussion about the existence of solutions for the optimization problem is added in Section 4. Our numerical experiments shown in Section 5 illustrate that our method performs very robustly. In particular we show that the storage amount needed for the low-rank scheme is typically a fraction of the storage requirement for the full problem. This is combined with a thourough parameter study where all system parameters are varied over orders of magnitudes with only observing very benign rank growth.

2 Problem formulation

We start our discussion by introducing the formulation of the Navier-Stokes equations that we are going to use throughout this paper:

$$\mathbf{y}_t - \nu\Delta\mathbf{y} + (\mathbf{y}\cdot\nabla)\,\mathbf{y} + \nabla p = \mathbf{u}, \tag{1}$$

$$\nabla\cdot\mathbf{y} = 0, \tag{2}$$

posed on domain $\Omega \in \mathbb{R}^{2,3}$ with appropriate boundary and initial conditions (see [14] and the references mentioned therein for more details). Often one is also interested in solving optimization problems where the Navier-Stokes equations appear as a constraint [7, 8, 22]. For this we consider the following objective function

$$J(\mathbf{y},\mathbf{u}) = \frac{1}{2}\left\|\mathbf{y}-\mathbf{y}_d\right\|_{Q_0}^2 + \frac{\beta}{2}\left\|\mathbf{u}\right\|_{Q_c}^2 \tag{3}$$

where $Q_o = \Omega_o \times [0,T]$ and $Q_c = \Omega_c \times [0,T]$ are space-time cylinders. Here $\Omega_o \subseteq \Omega$ is the observation domain and $\Omega_c \subseteq \Omega$ the control domain. We for now assume that both are equal to Ω. The function $\mathbf{y}_d$ is the desired state. For this case the right hand side of equation (1) represents the control $\mathbf{u}$, which is computed in such a way that the solution of the Navier-Stokes equation is close to the desired state.

Additionally, we also consider an objective function including a vorticity term [22]

$$J_2(\mathbf{y},\mathbf{u}) = \frac{\alpha_1}{2}\left\|\mathbf{y}-\mathbf{y}_d\right\|_{Q_0}^2 + \frac{\alpha_2}{2}\left\|\mathrm{curl}(\mathbf{y})\right\|_{Q_c}^2 + \frac{\beta}{2}\left\|\mathbf{u}\right\|_{Q_c}^2 \tag{4}$$

Many people have studied the numerical solution of Navier-Stokes equation and also the optimization problem with them as a constraint. Our goal here is to discuss the possibility of extending the framework recently introduced for PDE-optimization problems [51] with linear constraints. This framework utilizes a low-rank structure of the solution and hence enables efficient solvers for the optimization problem.

Before discussing the Navier-Stokes case we briefly want to introduce the idea using the Stokes equations as an example. For this we consider the Stokes equations

$$\mathbf{y}_t - \nu\Delta\mathbf{y} + \bigtriangledown p = \mathbf{u} \tag{5}$$

$$\bigtriangledown\cdot\mathbf{y} = 0 \tag{6}$$

equipped with appropriate initial and boundary conditions. Employing a finite element discretization in space, an implicit Euler discretization for the temporal discretization of the PDE, and a trapezoidal rule for numerical integration of the objective function leads to a discretized optimization problem [52, 23]. The first order conditions using a Lagrangian with Lagrange multiplier $\boldsymbol{\lambda}$ then lead to the following system

$$\begin{bmatrix} \mathcal{M}_1 & 0 & \mathcal{K}^T \\ 0 & \mathbf{M}_2 & -\mathcal{M}_3^T \\ \mathcal{K} & -\mathcal{M}_3 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_h \\ p_h \\ \mathbf{u}_h \\ \boldsymbol{\lambda}_h \\ \xi_h \end{bmatrix} = \begin{bmatrix} \mathcal{M}_1\mathbf{y}_d \\ 0 \\ d \end{bmatrix}, \tag{7}$$

which we want to write in Kronecker notation utilizing the following that the Stokes equations are discretized as

$$\mathcal{K}\begin{bmatrix} \mathbf{y}_h \\ p_h \end{bmatrix} - \mathcal{M}_3 \mathbf{u}_h = d, \tag{8}$$

where

$$\mathcal{K} = \left(I_n \otimes \mathcal{L} + C \otimes \mathcal{M}\right), \quad \mathcal{M}_3 = I_n \otimes \begin{bmatrix} M \\ 0 \end{bmatrix}, \quad d = e_1 \otimes \begin{bmatrix} \mathbf{M}\mathbf{y}_0 \\ 0 \end{bmatrix} + f,$$

where $e_1 \in \mathbb{R}^n$ is the first unit vector (this term accounts for the initial state), and f agglomerates the boundary conditions (it will be written in detail later). The number n denotes the number of time-steps and $C \in \mathbb{R}^{n,n}$ is given by

$$C = \frac{1}{\tau}\begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix},$$

where τ is the time step. As for the spatial matrices,

$$\mathcal{L} = \begin{bmatrix} \mathbf{L} & B^T \\ B & 0 \end{bmatrix}$$

represents an instance of a time-dependent Stokes problem with B the discrete divergence, $\mathbf{L}$ is the Laplacian (including viscosity ν), and $\mathbf{M}$ is the mass matrix, associated with the velocity space, $\mathcal{M} = \begin{bmatrix} \mathbf{M} & 0 \\ 0 & 0 \end{bmatrix}$ is the mass matrix for the velocity-pressure space. The matrices $\mathcal{M}_1 = \Theta \otimes \mathcal{M}$ and $\mathbf{M}_2 = \beta\Theta \otimes \mathbf{M}$ with $\Theta = \tau \cdot \mathrm{diag}(\frac{1}{2}, 1, \ldots, 1, \frac{1}{2})$ denote the mass matrices coming from the discretization of the functional (3).

The goal then is to use the fact that the right hand side of the optimality system can be written in low-rank form and this can be carried trough an iterative solver like MINRES [46] without a substantial increase in the rank of the solution [51].

2.1 Low-rank approximation of the Navier-Stokes forward problem

The situation for the Navier-Stokes equations is more complex as the nonlinear convection term does not allow for such an easy description of the problem. Typically the Navier-Stokes equations are discretized in space followed by a discretization in time. One then has to solve a nonlinear problem at every time-step for which both Newton as well as Picard iterations have shown to be good candidates [14]. We here focus on the Picard iteration and follow the description in [14, Chapter 8.2] which establishes

$$\int \mathbf{y}_t \mathbf{v} - \int \nu \nabla \mathbf{y} : \nabla \mathbf{v} + c(\bar{\mathbf{y}}, \mathbf{y}, \mathbf{v}) - \int p(\nabla \cdot \mathbf{v}) = \int \mathbf{u}\mathbf{v} \qquad \forall \mathbf{v} \in H^1(\Omega) \tag{9}$$

$$\int q(\nabla \cdot \mathbf{y}) = 0 \qquad \forall q \in L_2(\Omega) \tag{10}$$

with the trilinear form

$$c(\bar{\mathbf{y}}, \mathbf{y}, \mathbf{v}) := \int (\bar{\mathbf{y}} \cdot \nabla \mathbf{y}) \cdot \mathbf{v},$$

where $\bar{\mathbf{y}}$ denotes the previous iterate of the nonlinear Picard solver. Note that this formulation is typically known as the Oseen equations and will be at the heart of this paper.

The basis for the low-rank solution in the Stokes case is based on the fact that the right hand side of the linear system is of low-rank or can be well approximated by a low-rank function. We start by considering the forward problem with the right-hand side $\mathbf{u}$. We now assume that $\mathbf{u}$ is either given or approximated by

$$\mathbf{u} = \sum_{i=1}^{r_{\mathbf{u}}} v_{i,\mathbf{u}}(t) w_{i,\mathbf{u}}(x)$$

which is in discretized form written as

$$\mathbf{u}_h = \sum_{i=1}^{r_{\mathbf{u}_h}} v_{i,\mathbf{u}_h} \otimes w_{i,\mathbf{u}_h}.$$

Using this for the first step of the Picard iteration, an implicit Euler discretization in time and finite elements in space, we obtain the following discretized system,

$$(C \otimes \mathcal{M} + I_n \otimes \mathcal{L})\, \mathbf{y}_h = \sum_{i=1}^{r_{\mathbf{u}_h}} v_{i,\mathbf{u}_h} \otimes w_{i,\mathbf{u}_h} + d. \tag{11}$$

Note that this represents the Stokes system as no other initial guess was used. We now assume that at a step ℓ of our Picard iteration, the previous solution is given by

$$\bar{\mathbf{y}}_h = \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} v_{i,\bar{\mathbf{y}}_h} \otimes w_{i,\bar{\mathbf{y}}_h}, \tag{12}$$

and we want to compute the next ($\ell+1$-th) Picard iteration. Notice that the trilinear form in (9) is linear in $\bar{\mathbf{y}}$, and hence preserves the low-rank form of $\bar{\mathbf{y}}_h$. Let us assume that finite elements $\{\boldsymbol{\phi}_1(x), \ldots, \boldsymbol{\phi}_m(x)\}$ are used for the discretization of the velocity in space. Then $\bar{\mathbf{y}}(t_l, x)$ is constructed from $\bar{\mathbf{y}}_h$ by interpolation

$$\bar{\mathbf{y}}(t_l, x) = \sum_{k=1}^{m} \bar{\mathbf{y}}_{h,k}(t_l) \boldsymbol{\phi}_k(x) = \sum_{k=1}^{m} \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} v_{i,\bar{\mathbf{y}}_h,l} \otimes w_{i,\bar{\mathbf{y}}_h,k} \boldsymbol{\phi}_k(x),$$

where $l = 1, \ldots, n$ is the time step. Plugging this into $c(\bar{\mathbf{y}}, \mathbf{y}, \mathbf{v})$, we obtain

$$c(\bar{\mathbf{y}}(t_l), \boldsymbol{\phi}_{j'}, \boldsymbol{\phi}_j) = \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} v_{i,\bar{\mathbf{y}}_h,l} \otimes (\mathbf{N}_i)_{j,j'}, \quad j, j' = 1, \ldots, m, \tag{13}$$

where $\mathbf{N}_i \equiv \mathbf{N}(w_{i,\bar{\mathbf{y}}_h}) \in \mathbb{R}^{m\times m}$ is defined by its elements

$$(\mathbf{N}_i)_{j,j'} = \int \boldsymbol{\phi}_j \left(\sum_{k=1}^{m} w_{i,\bar{\mathbf{y}}_h,k} \boldsymbol{\phi}_k \right) \nabla \boldsymbol{\phi}_{j'}. \tag{14}$$

Since $\boldsymbol{\phi}_k(x)$ are finitely supported, most of the triple products of $\boldsymbol{\phi}$ above are zeros, and $\mathbf{N}_i$ can be assembled in $\mathcal{O}(m)$ operations.

Now, for the fully discretized problem, we have

$$\left(C \otimes \mathcal{M} + I_n \otimes \mathcal{L} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathcal{N}_i \right) \mathbf{y}_h = \sum_{i=1}^{r_{\mathbf{u}_h}} v_{i,\mathbf{u}_h} \otimes w_{i,\mathbf{u}_h} + \bar{d}, \tag{15}$$

where $\mathcal{N}_i = \mathrm{bkldiag}(\mathbf{N}_i, 0)$ and $D_i = \mathrm{diag}(v_{i,\bar{\mathbf{y}}_h})$. Note that $\bar{d}$ consists of the contributions coming from the boundary conditions at the previous step due to the changing matrix $\mathbf{N}_i$. The Picard iteration is now continued until convergence. The main advantage of the nonlinear solver, i.e., Picard iteration in this case, as the outer iteration is that we can reduce the storage amount for the inner space-time problem. This is true if the ranks $r_{\mathbf{y}_h}$ are kept small and hence the amount of storage is kept small and only a few matrices $\mathbf{N}_i$ have to be assembled.

Note that our method can also be used when different temporal discretizations [14] are used

$$\frac{1}{\tau}\left(\mathbf{y}^{(l+1)} - \mathbf{y}^{(l)}\right) + (\bar{\mathbf{y}}^* \cdot \nabla)\,\mathbf{y}^{(l+\frac{1}{2})} - \nu\Delta \mathbf{y}^{(l+\frac{1}{2})} + \nabla p^{(l+\frac{1}{2})} = \mathbf{u}^{(l+\frac{1}{2})} \tag{16}$$

$$\nabla \cdot \mathbf{y}^{(l+\frac{1}{2})} = 0 \tag{17}$$

where $\mathbf{y}^{(l+\frac{1}{2})} := \frac{1}{2}(\mathbf{y}^{(l+1)} + \mathbf{y}^{(l)})$ and similar for $\mathbf{u}$ and p. The choice $\bar{\mathbf{y}}^* = \mathbf{y}^{(l+\frac{1}{2})}$ represents the Crank-Nicolson scheme and $\bar{\mathbf{y}}^* = \frac{3}{2}\mathbf{y}^{(l)} - \frac{1}{2}\mathbf{y}^{(l-1)}$ the Simo-Amero scheme (cf. [14] for more details and further references). Note that as we approximate the space-time solution in a low-rank form we do not proceed sequentially in time and these schemes need to be rewritten for our purposes. Hence we consider an all-at-once semi-discretized system for which the state can then be written as

$$\mathbf{y} := \begin{bmatrix} \mathbf{y}^{(1)} \\ \mathbf{y}^{(2)} \\ \vdots \\ \mathbf{y}^{(n)} \end{bmatrix}. \tag{18}$$

For this we need the two matrices

$$C = \frac{1}{\tau}\begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} \text{ and } \tilde{C} = \frac{1}{2}\begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & \ddots & \ddots & \\ & & 1 & 1 \end{bmatrix} \tag{19}$$

$$C\mathbf{y} - \nu\Delta\tilde{C}\mathbf{y} + \mathbf{N}(\bar{\mathbf{y}})\tilde{C}\mathbf{y} + \nabla\tilde{C}p = \tilde{C}\mathbf{u} \tag{20}$$

$$\nabla \cdot \tilde{C}\mathbf{y} = 0 \tag{21}$$

where for the Crank-Nicolson scheme the semi-discretized part is given by

$$\mathbf{N}(\bar{\mathbf{y}})\tilde{C}\mathbf{y} = \left(\hat{C}\bar{\mathbf{y}}\right)^T \text{blkdiag}(\cdot\nabla, \ldots, \cdot\nabla)\tilde{C}\mathbf{y}, \tag{22}$$

where $\hat{C} = \tilde{C}$ represents a Crank-Nicolson scheme or the Simo-Amero scheme via

$$\hat{C} = \frac{1}{2}\begin{bmatrix} 1 & & & & \\ 0 & 1 & & & \\ -1 & 3 & 0 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 3 & 0 \end{bmatrix}.$$

This is then followed by a spatial discretization where the discretization of most terms in (20) is straightforward and we focus on the term $\mathbf{N}(\bar{\mathbf{y}})\tilde{C}\mathbf{y}$ which using (14) is discretized as

$$\sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} \text{diag}(\hat{C}v_{i,\bar{\mathbf{y}}_h})\tilde{C} \otimes \mathbf{N}(w_{i,\bar{\mathbf{y}}_h}) = \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} \tilde{D}_i \otimes \mathbf{N}(w_{i,\bar{\mathbf{y}}_h}),$$

where $\tilde{D}_i = \text{diag}(\hat{C}v_{i,\bar{\mathbf{y}}_h})\tilde{C}$. This leaves us with the overall space-time discretization

$$\left(C \otimes \mathcal{M} + \tilde{C} \otimes \mathcal{L} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} \tilde{D}_i \otimes \mathcal{N}_i\right)\mathbf{y}_h = \sum_{i=1}^{r_{\mathbf{u}_h}} v_{i,\mathbf{u}_h} \otimes w_{i,\mathbf{u}_h} + \bar{d}. \tag{23}$$

2.2 Low-rank approximation of the optimization problem

We now consider the case when the Navier-Stokes equations represent a constraint for a misfit functional such as the one given in (3). There are now two approaches that one can take to solve the optimization. The first one discretizes the objective function first and then optimizes the discrete problem while the second approach first derives the optimality system and then discretizes the resulting system. As we are only concerned with the efficient solution of a discretized problem we believe that our approach can be used in both cases but focus on the optimize then discretize case. This means that we first build an infinite dimensional Lagrangian and then consider its variation with respect to state, pressure, control, and two Lagrange multipliers that can be identified as the adjoint state and adjoint pressure [54, 27, 22]. We here simply

state the optimality system as derived in [22, Example 3.1],

$$\begin{aligned} \mathbf{y}_t - \Delta\mathbf{y} + (\mathbf{y}\cdot\nabla)\,\mathbf{y} + \bigtriangledown p &= \mathbf{u} && \text{on } [0,T]\times\Omega \\ \nabla\cdot\mathbf{y} &= 0 && \text{on } [0,T]\times\Omega \\ \mathbf{y}(0,.) &= \mathbf{y}_0 && \text{on } \Omega \\ \mathbf{y} &= \mathbf{y}_\Gamma && \text{on } \Gamma \end{aligned} \tag{24}$$

$$\begin{aligned} -\boldsymbol{\lambda}_t - \Delta\boldsymbol{\lambda} - (\mathbf{y}\cdot\nabla)\,\boldsymbol{\lambda} + (\nabla\mathbf{y})^T\,\boldsymbol{\lambda} + \bigtriangledown\xi &= -(\mathbf{y}-\mathbf{y}_d) && \text{on } [0,T]\times\Omega \\ \nabla\cdot\boldsymbol{\lambda} &= 0 && \text{on } [0,T]\times\Omega \\ \boldsymbol{\lambda}(T,.) &= -(\mathbf{y}(T)-\mathbf{y}_d(T)) && \text{on } \Omega \\ \boldsymbol{\lambda} &= 0 && \text{on } \Gamma \end{aligned} \tag{25}$$

$$\beta\mathbf{u} + \boldsymbol{\lambda} = 0 \qquad \text{on } [0,T]\times\Omega. \tag{26}$$

Now it is easily seen that this is a nonlinear problem due to the nonlinearity coming from the Navier-Stokes equation. Additional, nonlinearities could come into this equation if more complicated objective functions are considered. Note that the questions of existence and uniqueness are answered in [22]. Once again this equation has to be treated using a nonlinear solver and we again propose the use of a Oseen (Picard-type) iteration [47] to give

$$\begin{aligned} \mathbf{y}_t - \Delta\mathbf{y} + (\bar{\mathbf{y}}\cdot\nabla)\,\mathbf{y} + \bigtriangledown p &= \mathbf{u} \\ \nabla\cdot\mathbf{y} &= 0 \end{aligned}$$

$$\begin{aligned} -\boldsymbol{\lambda}_t - \Delta\boldsymbol{\lambda} - (\bar{\mathbf{y}}\cdot\nabla)\,\boldsymbol{\lambda} + (\nabla\bar{\mathbf{y}})^T\,\boldsymbol{\lambda} + \bigtriangledown\xi &= -(\mathbf{y}-\mathbf{y}_d) \\ \nabla\cdot\boldsymbol{\lambda} &= 0 \end{aligned}$$

$$\beta\mathbf{u} + \boldsymbol{\lambda} = 0.$$

where for brevity we omitted initial/final and boundary conditions. After that, we update $\bar{\mathbf{y}} = \mathbf{y}$ and proceed with the next iteration. We are now proposing the same solution as in the forward simulation. Additionally, we assume that all quantities are discretized in time and space. This means at each step of the algorithm we assume

the states, control, and adjoint states are given by

$$\mathbf{y}_h = \sum_{i=1}^{r_{\mathbf{y}_h}} v_{i,\mathbf{y}_h} \otimes w_{i,\mathbf{y}_h} \quad p_h = \sum_{i=1}^{r_{\boldsymbol{\lambda}_h}} v_{i,p_h} \otimes w_{i,p_h} \tag{27}$$

$$\boldsymbol{\lambda}_h = \sum_{i=1}^{r_{\boldsymbol{\lambda}_h}} v_{i,\boldsymbol{\lambda}_h} \otimes w_{i,\boldsymbol{\lambda}_h} \quad \xi_h = \sum_{i=1}^{r_{\xi_h}} v_{i,\xi_h} \otimes w_{i,\xi_h} \tag{28}$$

$$\mathbf{u}_h = \sum_{i=1}^{r_{\mathbf{u}_h}} v_{i,\mathbf{u}_h} \otimes w_{i,\mathbf{u}_h}. \tag{29}$$

The Oseen equation that we have to solve is then of the following form

$$\begin{bmatrix} \Theta \otimes \mathcal{M} & 0 & \mathcal{K}^* \\ 0 & \Theta \otimes \beta\mathbf{M} & -\mathcal{M}_3^\top \\ \mathcal{K} & -\mathcal{M}_3 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_h \\ p_h \\ \mathbf{u}_h \\ \boldsymbol{\lambda}_h \\ \xi_h \end{bmatrix} = b \tag{30}$$

where b represents the right-hand side and $\mathcal{K}$ describes the forward operator for the space-time Navier-Stokes equations

$$\mathcal{K} = C \otimes \mathcal{M} + I_n \otimes \mathcal{L} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathcal{N}_i. \tag{31}$$

The adjoint PDE represented by $\mathcal{K}^*$ contains more terms than the forward equation due to the terms $(\mathbf{y} \cdot \nabla)\boldsymbol{\lambda} + (\nabla \mathbf{y})^T \boldsymbol{\lambda}$. As in the forward problem, we assume (12), so that $(\bar{\mathbf{y}} \cdot \nabla)\boldsymbol{\lambda}$ is discretized as $\sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathcal{N}_i$. The term $(\nabla \bar{y})^T \boldsymbol{\lambda}$ now becomes $\sum_{i=1}^{r_{\bar{y}}} D_i \otimes \mathbf{H}_i(\bar{\mathbf{y}}_h)$ with $\mathbf{H}_i(\bar{\mathbf{y}}_h)$ being a matrix of entries $\int \boldsymbol{\phi}_j \cdot \nabla \left(\sum_{k=1}^m w_{i,\bar{\mathbf{y}}_h,k} \boldsymbol{\phi}_k\right) \cdot \boldsymbol{\phi}_{j'}$ for $j, j' = 1, \ldots, m$. The adjoint matrix is then given by

$$\mathcal{K}^* = C^\top \otimes \mathcal{M} + I_n \otimes \mathcal{L} - \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathcal{N}_i + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathcal{H}_i,$$

where $\mathcal{H}_i = \text{blkdiag}(\mathbf{H}_i, 0)$. We have now seen that we can perform an outer Picard iteration and then proceed in a low-rank fashion with the inner Oseen problem. Algorithm 1 depicts a pseudo-code of our proposed scheme. We will in the following discuss the numerical solver for the low-rank solution of the linear system (30). We also want to discuss the case when the objective function is changed to include the vorticity term (4) following results in a different formulation of the adjoint equation [29, 36]

$$-\boldsymbol{\lambda}_t - \Delta\boldsymbol{\lambda} - (\mathbf{y} \cdot \nabla)\boldsymbol{\lambda} + (\nabla \mathbf{y})^T \boldsymbol{\lambda} + \bigtriangledown \xi = -\alpha_1(\mathbf{y} - \mathbf{y}_d) - \alpha_2 \mathbf{curl}(\mathbf{curl}(\mathbf{y})). \tag{32}$$

Algorithm 1 Picard iteration for the Navier-Stokes optimization problem

Require: Desired state $\mathbf{y}_d$, initial state $\mathbf{y}_0$, initial guess $\bar{\mathbf{y}}$.

for $\ell = 1$ to $\mathtt{L}_{\max}$ **do**
  Compute $\mathcal{D}_i, \mathcal{N}_i, \mathcal{H}_i$ for $i = 1, \ldots, r_{\bar{\mathbf{y}}_h}$
  Update the right-hand side $\bar{d}$
  Solve the system (33)
  if Error $\|\mathbf{y}_h - \bar{\mathbf{y}}_h\|$ is small **then**
    Stop
  else
    Replace $\bar{\mathbf{y}}_h = \mathbf{y}_h$ and continue
  end if
end for
return $[\mathbf{y}_h, \mathbf{u}_h, \boldsymbol{\lambda}_h]$

We now get the following system at every nonlinear step

$$\begin{bmatrix} \Theta \otimes (\alpha_1 \mathcal{M} + \alpha_2 \mathcal{L}_0) & 0 & \mathcal{K}^* \\ 0 & \Theta \otimes \beta \mathbf{M} & -\mathcal{M}_3^\top \\ \mathcal{K} & -\mathcal{M}_3 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_h \\ p_h \\ \mathbf{u}_h \\ \boldsymbol{\lambda}_h \\ \xi_h \end{bmatrix} = \begin{bmatrix} (\Theta \otimes \alpha_1 \mathbf{M}) \mathbf{y}_d \\ 0 \\ 0 \\ \bar{d} \\ 0 \end{bmatrix} = \begin{bmatrix} b_1 \\ 0 \\ b_2 \\ b_3 \\ 0 \end{bmatrix}, \tag{33}$$

where $\mathcal{L}_0 = \begin{bmatrix} \mathbf{L} & 0 \\ 0 & 0 \end{bmatrix}$ represents the discretized version of $\mathbf{curl}(\mathbf{curl}(\mathbf{y}))$, which is just the Laplacian operator, since the velocity is divergence-free.

3 Solution algorithms

We focus now on the efficient solution of the system (30) in low-rank form. Having solved the full KKT system (30), the low-rank format of the solution (27)–(28) can be computed by the Singular Value Decomposition. This is called an *offline* stage in model reduction methods [37]. Our goal is to avoid this expensive offline stage and compute the low-rank factors of the solution *directly*. One of the best tools for this task is the alternating iteration.

3.1 Alternating tensor product methods for the forward problem

First, we start from a single linear system $Ay = b$, where $A \in \mathbb{R}^{nm \times nm}$ and $b \in \mathbb{R}^{nm}$ are given, and the solution sought in the low-rank form

$$A = \sum_{i=1}^{r_A} F_{i,A} \otimes G_{i,A}, \quad b = \sum_{i=1}^{r_b} v_{i,b} \otimes w_{i,b}, \qquad y = \sum_{i=1}^{r_y} v_{i,y} \otimes w_{i,y}.$$

As a motivating example, consider the case $A = A^\top > 0$. Then the linear system can be solved as a minimization of the energy functional, $y = \arg\min_y J(y)$, with $J(y) = y^\top Ay - 2y^\top b$. Now, plug the low-rank decomposition of y into J, and optimize it sequentially (or *alternating*, hence the name) over v_y and w_y:

$$v_y = \arg\min_{v_y \in \mathbb{R}^{nr_y}} J\left(\sum_{i=1}^{r_y} v_{i,y} \otimes w_{i,y}\right), \quad w_y = \arg\min_{w_y \in \mathbb{R}^{mr_y}} J\left(\sum_{i=1}^{r_y} v_{i,y} \otimes w_{i,y}\right), \tag{34}$$

where v_y and w_y denote vertically stacked $v_{i,y}$ and $w_{i,y}$. Differentiating J w.r.t. to the elements of v and w, we can find that they are defined by smaller linear systems. Let us introduce $V_y \in \mathbb{R}^{n \times r_y}$ and $W_y \in \mathbb{R}^{m \times r_y}$, being matrices of horizontally stacked $v_{i,y}$ and $w_{i,y}$, respectively. Then (34) is satisfied by solving

$$\begin{aligned} \left[(I_n \otimes W_y)^\top A\,(I_n \otimes W_y)\right] v_y &= (I_n \otimes W_y)^\top b, \\ \left[(V_y \otimes I_m)^\top A\,(V_y \otimes I_m)\right] w_y &= (V_y \otimes I_m)^\top b, \end{aligned} \tag{35}$$

or shortly we can write $\hat{A}v_y = \hat{b}$, $\check{A}w_y = \check{b}$.

These two systems are solved one after another until convergence. Minimization of (34) is equivalent to minimization of the A-norm of the error, hence the method is called Alternating Least Squares [35]. Due to linearity of (35), it was also called Alternating Linear Scheme [25], abbreviated by *ALS* in both cases. Although it is difficult to prove theoretical convergence (it is essentially local [50]), in practice this method often converges rapidly, provided the rank r_y is high enough.

However, it is inconvenient to guess the rank a priori. There is a variety of methods which *enrich* the solution factors by some auxiliary vectors, and thus allow to increase the rank, adapting it to a desired error threshold. It is reasonable to select the enrichment related to the current residual. For example, *ADI* methods [56] solve shifted linear systems and expand e.g. V_y by $(F_A - sI)^{-1}V_R$, where V_R is a low-rank factor of the residual, *greedy* methods [53, 1, 42] compute rank-1 factors of the solutions to $Az = b - Ay$. However, an advantage of the variational formulation (34) is that the low-rank factors deliver locally minimal A-norm error in each step. Therefore, it might be more efficient to combine the enrichment with the Galerkin update (35). This is performed in the *orthogonal* greedy [42] and the *alternating minimal energy* [11] algorithms.

These methods converge relatively well if the matrix is positive definite. However, as was noticed in [5], even with orthogonal factors V and W, the Galerkin projection (35) can become degenerate if A is a saddle-point system like (33). To avoid this issue, we need to take the saddle-point structure into account explicitly.

3.2 Alternating methods for the inverse problem

Let us consider a block system

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix},$$

where each submatrix of A or subvector of b is presented in its own low-rank form, and the sizes of all blocks coincide[1]. However, we will factorize the solution components with one of the blocks *shared*: we suppose that either

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \sum_{i=1}^{r_y} \begin{bmatrix} v_{i,y,1} \\ v_{i,y,2} \\ v_{i,y,3} \end{bmatrix} \otimes \hat{w}_{i,y}, \quad \text{or} \quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \sum_{i=1}^{r_y} \check{v}_{i,y} \otimes \begin{bmatrix} w_{i,y,1} \\ w_{i,y,2} \\ w_{i,y,3} \end{bmatrix},$$

where $\hat{w}_{i,y} \in \mathbb{R}^m$, $\check{v}_{i,y} \in \mathbb{R}^n$, and agglomerated matrices are $\hat{W}_y \in \mathbb{R}^{m \times r_y}$ and $\check{V}_y \in \mathbb{R}^{n \times r_y}$. Now, we can write two ALS steps in the *block* form

$$\begin{aligned} \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} & \hat{A}_{13} \\ \hat{A}_{21} & \hat{A}_{22} & \hat{A}_{23} \\ \hat{A}_{31} & \hat{A}_{32} & \hat{A}_{33} \end{bmatrix} \begin{bmatrix} v_{y,1} \\ v_{y,2} \\ v_{y,3} \end{bmatrix} &= \begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix}, \qquad \begin{aligned} \hat{A}_{kl} &= (I \otimes \hat{W}_y)^\top A_{kl} (I \otimes \hat{W}_y), \\ \hat{b}_k &= (I \otimes \hat{W}_y)^\top b_k, \end{aligned} \\ \begin{bmatrix} \check{A}_{11} & \check{A}_{12} & \check{A}_{13} \\ \check{A}_{21} & \check{A}_{22} & \check{A}_{23} \\ \check{A}_{31} & \check{A}_{32} & \check{A}_{33} \end{bmatrix} \begin{bmatrix} w_{y,1} \\ w_{y,2} \\ w_{y,3} \end{bmatrix} &= \begin{bmatrix} \check{b}_1 \\ \check{b}_2 \\ \check{b}_3 \end{bmatrix}, \qquad \begin{aligned} \check{A}_{kl} &= (\check{V}_y \otimes I)^\top A_{kl} (\check{V}_y \otimes I), \\ \check{b}_k &= (\check{V}_y \otimes I)^\top b_k, \end{aligned} \end{aligned} \tag{36}$$

where $k, l = 1, 2, 3$. Note that the blocks $\hat{W}_y$ and $\check{V}_y$ do not contain the enumerator k, i.e. they serve as common bases for the components y_k. To compute this common basis (e.g. $\check{V}_y$), we can use the truncated SVD. Similarly to a single V_y in the previous section, we consider each component as a matrix $V_{y,k} \in \mathbb{R}^{n \times r_y}$. Having computed $V_{y,k}$ in the first step of (36), we factorize via SVD

$$\begin{bmatrix} V_{y,1} & V_{y,1} & V_{y,3} \end{bmatrix} = \check{V}_y S P^\top + \mathcal{E}, \quad \text{s.t.} \quad \|\mathcal{E}\|_F \le \varepsilon \|S\|_F,$$

where $\check{V}_y^\top \check{V}_y = I_{\hat{r}_y}$, S is a diagonal matrix of $\hat{r}_y$ dominant singular values, and $P \in \mathbb{R}^{3r_y \times \hat{r}_y}$ is a matrix of right singular vectors. Left singular vectors $\check{V}_y \in \mathbb{R}^{n \times \hat{r}_y}$ give the sought common basis. In the same way we derive $\hat{W}_y$ from $W_{y,k}$ after the second step of (36).

Notice that the new rank $\hat{r}_y$ can be chosen from the range $1, \dots, 3r_y$. That is, the blocked storage allows to increase the rank without any enrichment.[2] This is similar to the *Density Matrix Renormalization Group* (DMRG) method [57], developed in quantum physics to solve high-dimensional eigenvalue problems in low-rank formats. However, the DMRG method applied in our two-dimensional case would require to solve the whole problem without any reduction, whereas the block ALS formulation (36) allows to have both the rank adaptivity and moderate complexity.

The block ALS method requires only submatrices A_{kl} to be positive (semi-)definite, the whole matrix A needs only to be invertible. A drawback, however, is that the

[1] In practical computations, y_1, y_2, y_3 have the meanings of $\mathbf{y}_h$, $\mathbf{u}_h$ and $\boldsymbol{\lambda}_h$, but here we present the scheme in an abstract form, therefore we use more abstract notation than in Sec. 2. Later we will also show that the requirement to have all block sizes equal is not restrictive.

[2] Nonetheless, a residual-based enrichment is usually recommended, as it improves the convergence and accuracy. Even a very rough and fast low-rank approximation to the residual is usually a sufficiently good enrichment [11].

submatrices should be square. Moreover, with the Navier-Stokes constraints, $A_{31} = \mathcal{K}$ (31) and $A_{13} = \mathcal{K}^*$ are themselves saddle-point matrices. To avoid this issue, we will compute the pressures separately.

3.3 Alternating methods for the (Navier-)Stokes equation

Let us start from the forward Navier-Stokes equation, which reads

$$\begin{bmatrix} \mathbf{K} & I_n \otimes B^\top \\ I_n \otimes B & \end{bmatrix} \begin{bmatrix} \mathbf{y}_h \\ p_h \end{bmatrix} = \begin{bmatrix} d \\ 0 \end{bmatrix}, \quad \mathbf{K} = C \otimes \mathbf{M} + I_n \otimes \mathbf{L} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathbf{N}_i.$$

Suppose $\mathbf{y}_h$ is presented in the low-rank form (27). If we are performing the second ALS step, the matrix remains invertible: the saddle-point structure is introduced only in the spatial variable, and the ALS projection of the temporal variable does not touch it. Then from the second row we have $BW_{\mathbf{y}_h} = 0$. For the moment, we leave the representation of p_h aside. Returning to the first ALS step, we project the first row by $I_n \otimes W_{\mathbf{y}_h}^\top$, so we have

$$(I_n \otimes W_{\mathbf{y}_h})^\top \mathbf{K} (I_n \otimes W_{\mathbf{y}_h}) v_{\mathbf{y}_h} + (I_n \otimes W_{\mathbf{y}_h}^\top B^\top) p_h = (I_n \otimes W_{\mathbf{y}_h})^\top d.$$

However, the second term is zero irrespectively of the pressure: $W_{\mathbf{y}_h}^\top B^\top = (BW_{\mathbf{y}_h})^\top = 0$. Therefore, the first ALS step is also well-posed in this scheme.

The problem is that the formulation above is valid only with zero boundary conditions. With nonzero Dirichlet conditions enforced, we have either a different matrix instead of $B^\top$, or a nonzero second component of the right-hand side. We need to shift the velocity by some function, such that the sought solution is zero at the boundary.

At this point, it is reasonable to assume that the boundary values, as any other input data, are given in the low-rank form,

$$\mathbf{y}_h|_\Gamma = \sum_{i=1}^{r_\Gamma} v_{\Gamma,i} \otimes w_{\Gamma,i}, \tag{37}$$

where Γ denotes boundary degrees of freedom. We look for the solution in the form $\mathbf{y} = \mathbf{q} + \boldsymbol{\mu}$, where $\mathbf{q}|_\Gamma = 0$ and $\boldsymbol{\mu}|_\Gamma = \mathbf{y}|_\Gamma$. We could reformulate the equation for $\mathbf{q}$, if we find a convenient closed-form (and low-rank) expression for $\boldsymbol{\mu}$.

This can be done by solving a few stationary Stokes equations. Let us partition the spatial degrees of freedom, and hence, the matrix elements, as follows,

$$\mathbf{L} = \begin{pmatrix} \mathbf{L}_{\Omega\Omega} & \mathbf{L}_{\Omega\Gamma} \\ \mathbf{L}_{\Gamma\Omega} & \mathbf{L}_{\Gamma\Gamma} \end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} \mathbf{M}_{\Omega\Omega} & \mathbf{M}_{\Omega\Gamma} \\ \mathbf{M}_{\Gamma\Omega} & \mathbf{M}_{\Gamma\Gamma} \end{pmatrix}, \quad B = \begin{pmatrix} B_\Omega & B_\Gamma \end{pmatrix},$$

where "Ω" corresponds to the inner points, and "Γ" denotes the boundary points. The Stokes equation with nonzero boundary conditions can be written as follows,

$$\begin{bmatrix} \begin{pmatrix} \mathbf{L}_{\Omega\Omega} & \\ & I \end{pmatrix} & \begin{pmatrix} B_\Omega^\top \\ 0 \end{pmatrix} \\ \begin{pmatrix} B_\Omega & B_\Gamma \end{pmatrix} & \end{bmatrix} \begin{bmatrix} w_{\boldsymbol{\mu}_h,i} \\ p_{h,i} \end{bmatrix} = \begin{bmatrix} \begin{pmatrix} -\mathbf{L}_{\Omega\Gamma} w_{\Gamma,i} \\ w_{\Gamma,i} \end{pmatrix} \\ 0 \end{bmatrix}. \tag{38}$$

Since the Stokes equation is linear, it admits a superposition: solving (38) for all $i = 1, \ldots, r_\Gamma$, we obtain exactly the low-rank factors for $\boldsymbol{\mu}$, and summing them up in

$$\boldsymbol{\mu}_h = \sum_{i=1}^{r_\Gamma} v_{\Gamma,i} \otimes w_{\boldsymbol{\mu}_h,i},$$

we get the desired correction function.

The Navier-Stokes equation can now be rewritten for $\mathbf{q}_h$ without the boundary,

$$\begin{bmatrix} \mathbf{K}_{\Omega\Omega} & I \otimes B_\Omega^\top \\ I \otimes B_\Omega & \end{bmatrix} \begin{bmatrix} \mathbf{q}_h \\ p_h \end{bmatrix} = \begin{bmatrix} b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h \\ 0 \end{bmatrix}, \quad \text{where} \tag{39}$$

$$\begin{aligned} \mathbf{K}_{\Omega\Omega} &= C \otimes \mathbf{M}_{\Omega\Omega} + I \otimes \mathbf{L}_{\Omega\Omega} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \mathbf{N}_{i,\Omega\Omega}, \\ \mathbf{K}_{\Omega:} &= C \otimes \begin{bmatrix} \mathbf{M}_{\Omega\Omega} & \mathbf{M}_{\Omega\Gamma} \end{bmatrix} + I \otimes \begin{bmatrix} \mathbf{L}_{\Omega\Omega} & \mathbf{L}_{\Omega\Gamma} \end{bmatrix} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \begin{bmatrix} \mathbf{N}_{i,\Omega\Omega} & \mathbf{N}_{i,\Omega\Gamma} \end{bmatrix}, \end{aligned} \tag{40}$$

and $\mathbf{N} = \mathbf{N}(\bar{\mathbf{y}}_h) = \mathbf{N}(\bar{\mathbf{q}}_h + \boldsymbol{\mu}_h)$ is computed as previously from the last iterate $\bar{\mathbf{y}} = \bar{\mathbf{q}} + \boldsymbol{\mu}$, which *includes* the correction $\boldsymbol{\mu}$ together with the boundary nodes. Here, the fixed right-hand side b carries only the initial state, $b = e_1 \otimes \mathbf{M}\mathbf{y}_0$.

The presented ALS scheme computes only the velocity. To restore the pressure, suppose that the velocity is known. Then the first row in (39) gives an equation, which can be resolved by least squares:

$$\left(I_n \otimes B_\Omega B_\Omega^\top\right) p_h = \left(I_n \otimes B_\Omega\right)\left(b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h - \mathbf{K}_{\Omega\Omega}\mathbf{q}_h\right).$$

The right-hand side is low-rank, since so are $\mathbf{K}, \boldsymbol{\mu}_h$ and $\mathbf{q}_h$, and the matrix in the left-hand side is a direct product, which can be inverted without changing the rank.

In practice a single step of such method may give only an approximate velocity, so we conduct several Chorin-type iterations,

$$\begin{aligned} \begin{bmatrix} \hat{\mathbf{K}}_{\Omega\Omega} & I_{r_{\mathbf{q}_h}} \otimes B_\Omega^\top \\ I_{r_{\mathbf{q}_h}} \otimes B_\Omega & \end{bmatrix} \begin{bmatrix} w_{\mathbf{q}_h} \\ dp_h \end{bmatrix} &= \begin{bmatrix} \left(V_{\mathbf{q}_h} \otimes I_m\right)^\top \left(b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h\right) - \left(V_{\mathbf{q}_h}^\top \otimes B_\Omega^\top\right) p_h \\ 0 \end{bmatrix}, \\ \check{\mathbf{K}}_{\Omega\Omega} v_{\mathbf{q}_h} &= \left(I_n \otimes W_{\mathbf{q}_h}\right)^\top \left(b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h\right), \\ \left(I_n \otimes B_\Omega B_\Omega^\top\right) p_h &= \left(I_n \otimes B_\Omega\right)\left(b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h - \mathbf{K}_{\Omega\Omega}\mathbf{q}_h\right), \end{aligned} \tag{41}$$

and so on from the first equation. Here, $\hat{\mathbf{K}}_{\Omega\Omega} = \left(V_{\mathbf{q}_h} \otimes I_m\right)^\top \mathbf{K}_{\Omega\Omega} \left(V_{\mathbf{q}_h} \otimes I_m\right)$, and $\check{\mathbf{K}}_{\Omega\Omega} = \left(I_n \otimes W_{\mathbf{q}_h}\right)^\top \mathbf{K}_{\Omega\Omega} \left(I_n \otimes W_{\mathbf{q}_h}\right)$. Remember that $V_{\mathbf{q}_h}$ and $W_{\mathbf{q}_h}$ are orthogonalized before using them as projectors. A dummy variable dp_h in (41) is only needed to impose the divergence-free equation, and can be discarded after the calculation. Ideally, we should perform this velocity-pressure process inside each Picard iteration.

In practice, we have found that one iteration is sufficient: we update $\mathbf{N}$ before starting the computation of $w_{\mathbf{q}_h}$, using the last $\mathbf{q}_h$ from the previous step.

The inverse problem is solved in a similar way, the difference is only that there are two "pressure-like" variables. Since we neither control nor observe the boundary, the Lagrange multiplier can have zero boundary condition. So, the final ALS iteration for the inverse Navier-Stokes equation is written as follows.

1. In the first step, we compute all spatial blocks,

$$\begin{bmatrix} \hat{\mathbf{M}}_1 & 0 & 0 & \hat{\mathbf{K}}^*_{\Omega\Omega} & I_r \otimes B_\Omega^\top \\ 0 & 0 & 0 & I_r \otimes B_\Omega & 0 \\ 0 & 0 & \hat{\mathbf{M}}_2 & -\hat{\mathbf{M}}_3^\top & 0 \\ \hat{\mathbf{K}}_{\Omega\Omega} & I_r \otimes B_\Omega^\top & -\hat{\mathbf{M}}_3 & 0 & 0 \\ I_r \otimes B_\Omega & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_{\mathbf{q}_h} \\ dp_h \\ w_{\mathbf{u}_h} \\ w_{\boldsymbol{\lambda}_h} \\ d\xi_h \end{bmatrix} = \begin{bmatrix} \hat{b}_1 - (\check{V}^\top \otimes B_\Omega^\top)\xi_h \\ 0 \\ 0 \\ \hat{b}_3 - (\check{V}^\top \otimes B_\Omega^\top)p_h \\ 0 \end{bmatrix}, \tag{42}$$

where $\hat{b}_1 = (\check{V}^\top \Theta \otimes \alpha_1 \mathbf{M}_{\Omega:})\mathbf{y}_d$, $\hat{b}_3 = (\check{V} \otimes I_m)^\top (b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h)$,

$$\begin{aligned} \hat{\mathbf{M}}_1 &= (\check{V}^\top \Theta \check{V}) \otimes (\alpha_1 \mathbf{M}_{\Omega\Omega} + \alpha_2 \mathbf{L}_{\Omega\Omega}), \\ \hat{\mathbf{M}}_2 &= (\check{V}^\top \Theta \check{V}) \otimes \beta \mathbf{M}_{\Omega\Omega}, \\ \hat{\mathbf{M}}_3 &= I_r \otimes \mathbf{M}_{\Omega\Omega}, \\ \hat{\mathbf{K}}_{\Omega\Omega} &= \check{V}^\top C \check{V} \otimes \mathbf{M}_{\Omega\Omega} + I_r \otimes \mathbf{L}_{\Omega\Omega} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} \check{V}^\top D_i \check{V} \otimes \mathbf{N}_{i,\Omega\Omega}, \\ \hat{\mathbf{K}}^*_{\Omega\Omega} &= \hat{\mathbf{K}}_{\Omega\Omega} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} \check{V}^\top D_i \check{V} \otimes (\mathbf{H}_{i,\Omega\Omega} - 2\mathbf{N}_{i,\Omega\Omega}). \end{aligned} \tag{43}$$

Compute the SVD $\begin{bmatrix} W_{\mathbf{q}_h} & W_{\mathbf{u}_h} & W_{\boldsymbol{\lambda}_h} \end{bmatrix} \approx \hat{W} S P^\top$ to derive the common basis.

2. In the second step, we compute the temporal blocks of the velocities,

$$\begin{bmatrix} \check{\mathbf{M}}_1 & 0 & \check{\mathbf{K}}^*_{\Omega\Omega} \\ 0 & \check{\mathbf{M}}_2 & -\check{\mathbf{M}}_3^\top \\ \check{\mathbf{K}}_{\Omega\Omega} & -\check{\mathbf{M}}_3 & 0 \end{bmatrix} \begin{bmatrix} v_{\mathbf{q}_h} \\ v_{\mathbf{u}_h} \\ v_{\boldsymbol{\lambda}_h} \end{bmatrix} = \begin{bmatrix} \check{b}_1 \\ 0 \\ \check{b}_3 \end{bmatrix}, \tag{44}$$

where $\check{b}_1 = (\Theta \otimes \alpha_1 \hat{W}^\top \mathbf{M}_{\Omega:})\mathbf{y}_d$, $\check{b}_3 = (I_n \otimes \hat{W})^\top (b_\Omega - \mathbf{K}_{\Omega:}\boldsymbol{\mu}_h)$,

$$\begin{aligned} \check{\mathbf{M}}_1 &= \Theta \otimes \hat{W}^\top (\alpha_1 \mathbf{M}_{\Omega\Omega} + \alpha_2 \mathbf{L}_{\Omega\Omega}) \hat{W}, \\ \check{\mathbf{M}}_2 &= \Theta \otimes \beta \hat{W}^\top \mathbf{M}_{\Omega\Omega} \hat{W}, \\ \check{\mathbf{M}}_3 &= I_n \otimes \hat{W}^\top \mathbf{M}_{\Omega\Omega} \hat{W}, \\ \check{\mathbf{K}}_{\Omega\Omega} &= C \otimes \hat{W}^\top \mathbf{M}_{\Omega\Omega} \hat{W} + I_n \otimes \hat{W}^\top \mathbf{L}_{\Omega\Omega} \hat{W} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \hat{W}^\top \mathbf{N}_{i,\Omega\Omega} \hat{W}, \\ \check{\mathbf{K}}^*_{\Omega\Omega} &= \check{\mathbf{K}}_{\Omega\Omega} + \sum_{i=1}^{r_{\bar{\mathbf{y}}_h}} D_i \otimes \hat{W}^\top (\mathbf{H}_{i,\Omega\Omega} - 2\mathbf{N}_{i,\Omega\Omega}) \hat{W}. \end{aligned} \tag{45}$$

Compute the SVD $\begin{bmatrix} V_{\mathbf{q}_h} & V_{\mathbf{u}_h} & V_{\boldsymbol{\lambda}_h} \end{bmatrix} \approx \check{V} S P^\top$ to derive the common basis.

3. In the third step, we update the pressures via standard low-rank algebra,

$$\begin{aligned} \left(I_n \otimes B_\Omega B_\Omega^\top\right) p_h &= \left(I_n \otimes B_\Omega\right) \left(b_\Omega - \mathbf{K}_{\Omega:} \boldsymbol{\mu}_h - \mathbf{K}_{\Omega\Omega} \mathbf{q}_h + \mathbf{M}_3 \mathbf{u}_h\right), \\ \left(I_n \otimes B_\Omega B_\Omega^\top\right) \xi_h &= \left(I_n \otimes B_\Omega\right) \left(\Theta \otimes \alpha_1 \mathbf{M} \mathbf{y}_d - \mathbf{K}^*_{\Omega\Omega} \boldsymbol{\lambda}_h - \mathbf{M}_1 \mathbf{q}_h\right). \end{aligned} \tag{46}$$

3.4 Preconditioning for the spatial system

The systems on the temporal factor (44),(45) and pressure (46) have moderate sizes and are essentially sparse. Our concern is now the spatial system (42), which can be too large for a direct solver even for a rank-1 solution. We solve this system by a preconditioned GMRES method. First, we use a block Jacobi approximation with respect to the rank dimension: the preconditioner reads $\tilde{A} = \mathrm{blkdiag}(\tilde{A}_1, \ldots, \tilde{A}_r)$, where

$$\tilde{A}_i = \begin{bmatrix} \tilde{\mathbf{M}}_{1,i} & 0 & 0 & \tilde{\mathbf{K}}^*_{i,\Omega\Omega} & B_\Omega^\top \\ 0 & 0 & 0 & B_\Omega & 0 \\ 0 & 0 & \tilde{\mathbf{M}}_{2,i} & -\mathbf{M}^\top & 0 \\ \tilde{\mathbf{K}}_{i,\Omega\Omega} & B_\Omega^\top & -\mathbf{M} & 0 & 0 \\ B_\Omega & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \begin{aligned} \tilde{\mathbf{M}}_{1,i} &= (\check{v}_i \otimes I_m)^\top \mathbf{M}_1 (\check{v}_i \otimes I_m), \\ \tilde{\mathbf{M}}_{2,i} &= (\check{v}_i \otimes I_m)^\top \mathbf{M}_2 (\check{v}_i \otimes I_m), \\ \tilde{\mathbf{K}}_{i,\Omega\Omega} &= (\check{v}_i \otimes I_m)^\top \mathbf{K}_{\Omega\Omega} (\check{v}_i \otimes I_m), \\ \tilde{\mathbf{K}}^*_{i,\Omega\Omega} &= (\check{v}_i \otimes I_m)^\top \mathbf{K}^*_{\Omega\Omega} (\check{v}_i \otimes I_m), \end{aligned}$$

for $i = 1, \ldots, r$. Solution of each of these systems is of the same complexity as the solution of the stationary problem. We precondition it by a block-triangular factor of the LU decomposition, where the Schur complement is approximated in the factored form using the matching argument [52]. Given the right-hand side $f = \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}^\top$, the inverse triangular factor can be applied as follows,

1. $y_1 = \tilde{\mathcal{M}}_{1,i}^{-1} f_1$,
2. $y_2 = \tilde{\mathbf{M}}_{2,i}^{-1} f_2$,
3. $y_3 = \mathcal{S}^{-1} \left(\tilde{\mathcal{K}}_{i,\Omega\Omega} y_1 - \begin{pmatrix} \mathbf{M} \\ 0 \end{pmatrix} y_2 - f_3 \right)$,

followed by the assembly $y = \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix}^\top$, where the first matrix is augmented as $\tilde{\mathcal{M}}_{1,i} = \begin{bmatrix} \tilde{\mathbf{M}}_{1,i} & 0 \\ 0 & \tau h^2 \beta I \end{bmatrix}$, and $\tilde{\mathcal{K}}_{i,\Omega\Omega} = \begin{bmatrix} \tilde{\mathbf{K}}_{i,\Omega\Omega} & B_\Omega^\top \\ B_\Omega & 0 \end{bmatrix}$. Now, instead of the exact Schur complement, we use the factored approximation

$$\mathcal{S}^{-1} = \begin{bmatrix} \tilde{\mathbf{K}}_{i,\Omega\Omega} + \frac{1}{\sqrt{\beta}} \mathbf{M} & B_\Omega^\top \\ B_\Omega & 0 \end{bmatrix}^{-1} \tilde{\mathcal{M}}_{1,i} \begin{bmatrix} \tilde{\mathbf{K}}^*_{i,\Omega\Omega} + \frac{1}{\sqrt{\beta}} \mathbf{M} & B_\Omega^\top \\ B_\Omega & 0 \end{bmatrix}^{-1}.$$

The matrices in the last equation are of the form of the forward stationary Stokes equation, and, for moderate grids, can be treated by the direct linear solver. Otherwise, many iterative solvers can be used [52, 39].

4 Existence of low-rank solutions

The efficiency of tensor product methods depends heavily on the particular values of the tensor ranks. However, it is difficult to estimate the ranks of the solution of the forward Navier-Stokes equation; moreover, in a highly turbulent regime they may actually reach the maximal values. We begin the analysis with the Stokes equation, which is to be followed by the analysis of the inverse Navier-Stokes equation. The ranks in the inverse problem can actually be smaller than the ranks of the forward problem, if the desired state is taken low-rank, for example, as the Stokes solution.

Lemma 1. *Given the Stokes system (5)–(6) after the discretization,*

$$\begin{bmatrix} C \otimes \mathbf{M} + I \otimes \mathbf{L} & I \otimes B^\top \\ I \otimes B & \end{bmatrix} \begin{bmatrix} \mathbf{y}_h \\ p_h \end{bmatrix} = \begin{bmatrix} \mathbf{u}_h \\ 0 \end{bmatrix}, \tag{47}$$

where C is the time difference matrix (19) of size n, $\mathbf{L}$ and $\mathbf{M}$ are the finite element discretization of Laplace and mass operators, respectively. Suppose the right-hand side is given in the low-rank form (29), the boundary condition is given in the low-rank form (37), and the solution is approximated in the form

$$\begin{bmatrix} \mathbf{y}_h \\ p_h \end{bmatrix} \approx \sum_{i=1}^{r} \check{v}_i \otimes \begin{bmatrix} w_{i,\mathbf{y}} \\ w_{i,p} \end{bmatrix}$$

up to an accuracy ε.

Then the rank of the solution is bounded by

$$r = \mathcal{O}\left(\left(\log \varepsilon^{-1} + \log h^{-1} + \log \tau^{-1} \right)^2 \left(r_{\mathbf{u}_h} + r_\Gamma \right) \right).$$

Proof. First, we exclude the boundary condition as described in the previous section, and arrive at the Stokes system on $\mathbf{q}_h$,

$$\begin{bmatrix} C \otimes \mathbf{M}_{\Omega\Omega} + I \otimes \mathbf{L}_{\Omega\Omega} & I \otimes B_\Omega^\top \\ I \otimes B_\Omega & \end{bmatrix} \begin{bmatrix} \mathbf{q}_h \\ p_h \end{bmatrix} = \begin{bmatrix} f_\Omega \\ 0 \end{bmatrix},$$

with $f_\Omega = \mathbf{u}_\Omega - C \otimes \begin{pmatrix} \mathbf{M}_{\Omega\Omega} & \mathbf{M}_{\Omega\Gamma} \end{pmatrix} \boldsymbol{\mu}_h - I \otimes \begin{pmatrix} \mathbf{L}_{\Omega\Omega} & \mathbf{L}_{\Omega\Gamma} \end{pmatrix} \boldsymbol{\mu}_h$.

Denoting by Φ the orthonormal basis of the kernel of B_Ω, we conclude that $\mathbf{q}_h \in \operatorname{span}(I \otimes \Phi)$, or $\mathbf{q}_h = (I \otimes \Phi)\tilde{\mathbf{q}}_h$. The coefficients $\tilde{\mathbf{q}}_h$ can be found by projecting the velocity equation onto $I \otimes \Phi^\top$. Since $\Phi^\top B_I^\top = (B_I \Phi)^\top = 0$, we have

$$\left(C \otimes \tilde{\mathbf{M}} + I \otimes \tilde{\mathbf{L}} \right) \tilde{\mathbf{q}}_h = \left(I \otimes \Phi^\top \right) f_\Omega, \tag{48}$$

where

$$\tilde{\mathbf{M}} = \Phi^\top \mathbf{M}_{\Omega\Omega} \Phi, \quad \tilde{\mathbf{L}} = \Phi^\top \mathbf{L}_{\Omega\Omega} \Phi.$$

Since both $\mathbf{M}_{\Omega\Omega}$ and $\mathbf{L}_{\Omega\Omega}$ are symmetric positive definite and Φ is orthogonal, it holds that $\tilde{\mathbf{M}}$ and $\tilde{\mathbf{L}}$ are symmetric positive definite. We can premultiply (48) by $(I \otimes \tilde{\mathbf{M}})^{-1}$, to then get

$$\tilde{\mathbf{q}}_h = \left(C \otimes I_m + I_n \otimes \tilde{\mathbf{M}}^{-1} \tilde{\mathbf{L}} \right)^{-1} \left(\left(I \otimes \tilde{\mathbf{M}}^{-1} \Phi^\top \right) f_\Omega \right).$$

The terms in the first brackets commute, and each of them is positive definite. The inverse can be approximated in the low-rank form by the exponential quadrature [15, 19]: for given R and k we introduce $t_k = \exp(k\pi/\sqrt{R})$, $c_k = t_k\pi/\sqrt{R}$, then

$$\tilde{\mathbf{K}}^{-1} \equiv \left(C \otimes I_m + I_n \otimes \tilde{\mathbf{M}}^{-1}\tilde{\mathbf{L}}\right)^{-1} \approx \sum_{k=-R}^{R} c_k \exp(-t_k C) \otimes \exp(-t_k \tilde{\mathbf{M}}^{-1}\tilde{\mathbf{L}}),$$

where the accuracy is estimated by $\mathcal{O}\left(\|\tilde{\mathbf{K}}\|_2 \mathrm{e}^{-\pi\sqrt{2R}}\right)$, provided that $\|\tilde{\mathbf{K}}^{-1}\| = \mathcal{O}(1)$. In other terms, the tensor rank of $\tilde{\mathbf{K}}^{-1}$ estimates as $\mathcal{O}((\log \varepsilon^{-1} + \log \operatorname{cond} \tilde{\mathbf{K}})^2)$.

Remember that the rank of $\mathbf{u}_h$, and, therefore, of $\mathbf{u}_\Omega$, is bounded by $r_{\mathbf{u}_h}$. The solution $\mathbf{q}_h$ in the initial Finite Element basis is restored without changing the rank, by multiplying $I \otimes \Phi$ by $\tilde{\mathbf{q}}_h$. Moreover, $\operatorname{cond} \tilde{\mathbf{K}} = \mathcal{O}(h^{-2} + \tau^{-1})$, where h is the spatial, and τ is the time grid steps. Finally, $\operatorname{rank}(\mathbf{q}_h) \le (2R+1)(r_{\mathbf{u}_h} + 2r_\Gamma)$, and the rank of the velocity $\mathbf{y}_h$ is estimated immediately, since the rank of $\boldsymbol{\mu}_h$ is r_Γ.

From (41) we see that the pressure rank is $r_{\mathbf{u}_h} + \operatorname{rank}(\mathbf{K}_{\Omega:}\mathbf{y})$, where now $\mathbf{K}_{\Omega:}$ has rank 2, which concludes the lemma.

□

Remark 1. *We can use the solution of the Stokes equation as a desired state in the misfit optimization (3), constrained by the Navier-Stokes equation (with smaller viscosity). This is reasonable if we want to get rid of turbulences. If the control is defined on the whole domain, it is known that the misfit decays with the regularization parameter, $\|\mathbf{y}_h - \mathbf{y}_d\| = \mathcal{O}(\sqrt{\gamma})$. Therefore, if we select $\varepsilon \sim \sqrt{\gamma}$, Lemma 1 is also valid for the solution of the inverse Navier-Stokes problem.*

If the state can be approximated by a low-rank decomposition, the low-rankness of the control comes straightforwardly, taking into account the Kronecker form of the forward operator (31). It is enough to multiply (31) by a rank-$r_{\mathbf{y}_h}$ representation.

Corollary 1. *Let the state be decomposed in a low-rank form (27) with the rank $r_{\mathbf{y}_h}$. Then the control admits a low-rank decomposition (29) with the rank $r_{\mathbf{u}_h} \le r_{\mathbf{y}_h}^2 + 2r_{\mathbf{y}_h}$.*

5 Numerical Results

The inflow condition $\mathbf{y}_1|_{x_1=0} = x_2(1-x_2)(1-\mathrm{e}^{-0.1t})$ is imposed at the left boundary, Neumann boundary condition at the right boundary $x_1 = L$ and zero condition at other walls. Other default parameters are given in Table 1. In the experiments below, we will vary each of them separately.

Our implementation is based on the IFISS package [13] and the tensor train toolbox [45] both of which are Matlab® based packages. Nevertheless, the methods presented here are usable in any other computational evironment.

We consider two types of the objective functional (4). First, we minimize only the distance to the desired state (i.e. $\alpha_1 = 1$, $\alpha_2 = 0$), where the desired state is the solution of the Stokes equation. Second, we minimize only the vorticity without any

Table 1: Default simulation parameters

n	h	T	ε	ν	β	α_1	α_2
2^{11}	$1/32$	200	10^{-4}	10^{-3}	10^{-3}	1	0

desired state by setting $\alpha_1 = 0$ and $\alpha_2 = 1$. Both observation and control domains coincide with the whole domain, $Q_o = Q_c = Q$. The domain is the so-called "backward step" with length $L = 5$ (see Fig. 16).

There are three ways to estimate the error in the solution. First, we compute the relative residual of the KKT system (33),

$$\begin{aligned}\text{residual} = &\left\| \mathcal{M}_1 \begin{bmatrix} \mathbf{y}_h \\ p_h \end{bmatrix} + \mathcal{K}^* \begin{bmatrix} \boldsymbol{\lambda}_h \\ \xi_h \end{bmatrix} - b_1 \right\|_F / \|b_1\|_F \\ &+ \left\| \mathcal{M}_2 \mathbf{u}_h - \mathcal{M}_3^\top \begin{bmatrix} \boldsymbol{\lambda}_h \\ \xi \end{bmatrix} \right\|_F / \|\mathcal{M}_2 \mathbf{u}_h\|_F \\ &+ \left\| \mathcal{K} \begin{bmatrix} \mathbf{y}_h \\ p_h \end{bmatrix} - \mathcal{M}_3 \mathbf{u} - b_3 \right\|_F / \|b_3\|_F.\end{aligned}$$

Second, we can solve the problem with two thresholds, e.g. ε and 0.1ε. Denoting e.g. the state velocity of the former one as $\mathbf{y}$, and of the latter one as $y_\star$, we can compute

$$\mathcal{E}(\mathbf{y}, \mathbf{y}_\star) = \|\mathbf{y} - \mathbf{y}_\star\|_F / \|\mathbf{y}_\star\|_F,$$

and similarly for the control $\mathbf{u}$ and other quantities. Let us assume that the true error $\|\mathbf{y} - \mathbf{y}_{ex}\|$ depends almost linearly on ε, $\|\mathbf{y} - \mathbf{y}_{ex}\| = C\varepsilon + o(\varepsilon)$. Justification of this linear dependence is given by Fig 9, 15. Then

$$\|\mathbf{y} - \mathbf{y}_{ex}\| = C\varepsilon + o(\varepsilon) \le \|\mathbf{y} - \mathbf{y}_\star\| + \|\mathbf{y}_\star - \mathbf{y}_{ex}\| = \|\mathbf{y} - \mathbf{y}_\star\| + C \cdot 0.1\varepsilon + o(\varepsilon),$$

and so $\|\mathbf{y} - \mathbf{y}_{ex}\| \le \frac{1}{0.9}\|\mathbf{y} - \mathbf{y}_\star\| + o(\varepsilon)$.

Third, we can measure the distance between two Picard iterations, $\mathcal{E}(\mathbf{y}, \bar{\mathbf{y}})$.

5.1 Convergence of the Picard iteration

In the first test, we check the convergence of the residual with the Picard iteration in the low-rank scheme. We test both distance and vorticity minimization, and two accuracy thresholds, $\varepsilon = 10^{-4}$ and $\varepsilon = 10^{-7}$. The results are presented in Fig. 1.

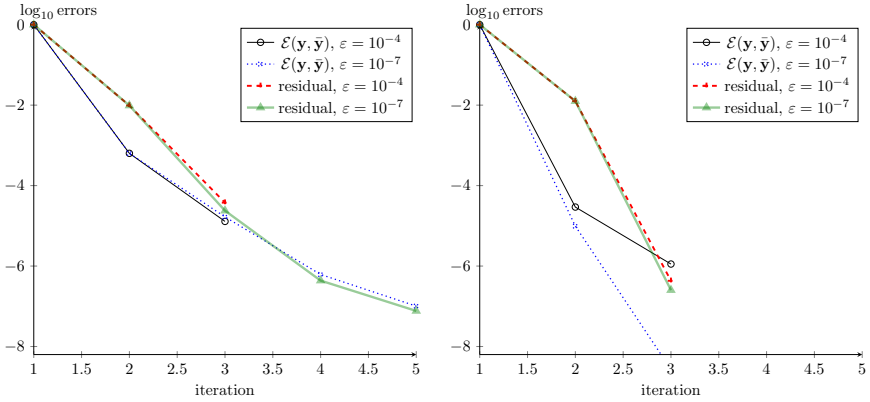
We see that the convergence is very fast and attained in 3 iterations in both lower-accuracy tests. The vorticity minimization converges faster than the misfit minimization, since the Stokes solution might actually be more turbulent than the one with the minimal vorticity.

5.2 Optimization of a tracking type functional

5.2.1 Comparison with the full scheme

An important justification for a new approach is a comparison with an established method. In our case, we compare the low-rank scheme (LR) with the classical precon-

Figure 1: Tracking functional, nonlinear convergence. Left: misfit minimization ($\alpha_1 = 1, \alpha_2 = 0$). Right: vorticity minimization ($\alpha_1 = 0, \alpha_2 = 1$).



ditioned GMRES for the full KKT system without low-rank approximations. Since the full data do not fit into the memory on fine grids, we perform two tests with $h = 1/8$ and $h = 1/16$. The results are reported in Fig. 2 and 3, respectively. In the horizontal axes, we vary the number of time steps n.

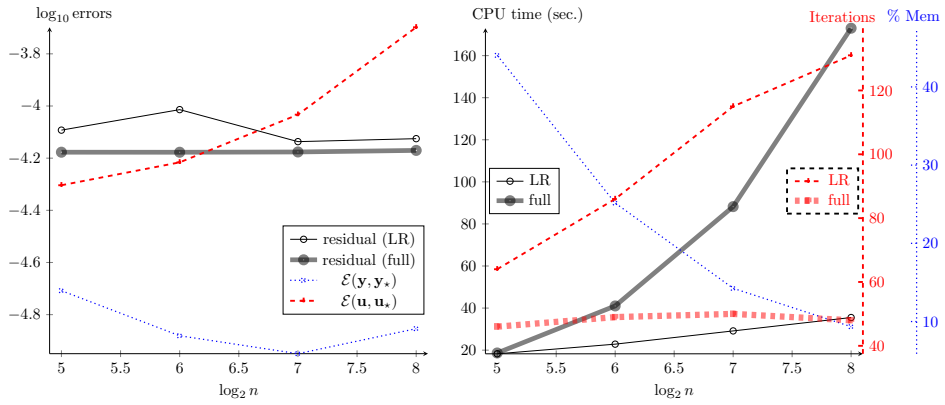
We see that the residuals are almost the same in both schemes, although the control error grows with the grid refinement. Another interesting quantity is the number of GMRES iterations. Both schemes perform the block Jacobi preconditioning with respect to time. However, the low-rank method invokes the GMRES (for the spatial factor) twice in each Picard iteration due to the ALS procedure. Moreover, it is being perturbed by the SVD truncation. This results in a higher number of iterations. However, the CPU time of the full scheme is at least equal to the time of the low-rank scheme for the coarsest grid, and becomes larger for finer grids. The ratio of memory costs, needed for the low-rank and full solutions, decreases with the system size and falls below 10% when the number of time steps exceeds a couple of hundreds.

5.2.2 Experiment with n

Here, we perform an extended test of the low-rank scheme with respect to the number of time steps, see Fig. 4. The method is robust even with very fine time grids up to $n = 65536$. The rank grows logarithmically with the number of time steps, and so do the CPU time and number of iterations.

In this and remaining figures, we report the rank of the solution, instead of the memory ratio as in Fig. 2, 3. The memory ratio reads % Mem $= \dfrac{n+m}{nm} r$. For example, for $h = 1/32$ it holds $m = 5890$ (for the velocity), and for $n \gtrsim m$, the memory consumption is reduced by factors of 100 to 300.

Figure 2: Tracking functional, LR vs full, $h = 1/8$. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



5.2.3 Experiment with T

We can also vary the time interval (Fig. 5). From the physical point of view, solving an optimal control problem on a smaller time interval means that we must exert a larger force. Interestingly, this is also true in the sense of computational complexity. While the rank is almost independent of T, the matrix become more ill-conditioned for smaller T, and hence the number of iterations grows, followed by the CPU time. The state error is also stable w.r.t. T, while the control error is higher for smaller T.

5.2.4 Experiment with β

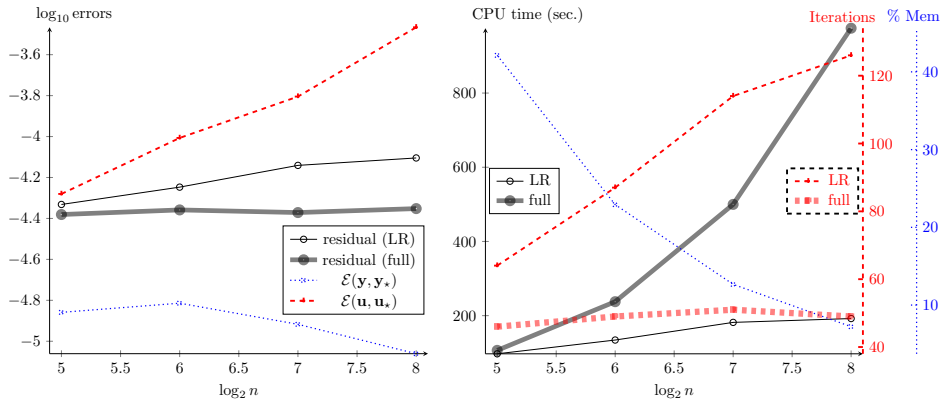
In this test, we vary the control regularization parameter. The results are shown in Fig 6. As an additional quantity, we report the distance to the desired state, $\mathcal{E}(\mathbf{y}, \mathbf{y}_d)$. This distance decays proportionally to $\sqrt{\beta}$, until being contaminated by the SVD error of level ε.

The scheme is quite robust until β becomes too large. In the right panel of Fig. 6, we show also the number of Picard iterations until convergence. We see that for small β the scheme needs 3 Picard iterations (in agreement with Fig. 1), but for $\beta \geq 0.1$ the convergence becomes slower. In particular, for $\beta \geq 1$, we were not able to compute a solution out of 15 Picard iterations. In future it might be more appropriate to use Newton methods for strongly nonlinear systems. In general smaller values for β are more important as they allow the state to better approximate the desired state.

5.2.5 Experiment with h

Now we vary the spatial mesh size. The results are shown in Fig 7. Here, the CPU time grows nearly quadratically with the reciprocal h, as expected for a two-dimensional problem. Other quantities grow as well, but in a much milder way.

Figure 3: Tracking functional, LR vs full, $h = 1/16$. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



5.2.6 Experiment with ν

Another parameter to vary is the viscosity. The results are shown in Fig 8. It is natural that all performance indicators improve with larger viscosity, as the system becomes closer to the Stokes regime. Nonetheless, even a highly convection dominated simulation with $\nu = 1/5000$ is tractable.

5.2.7 Experiment with ε

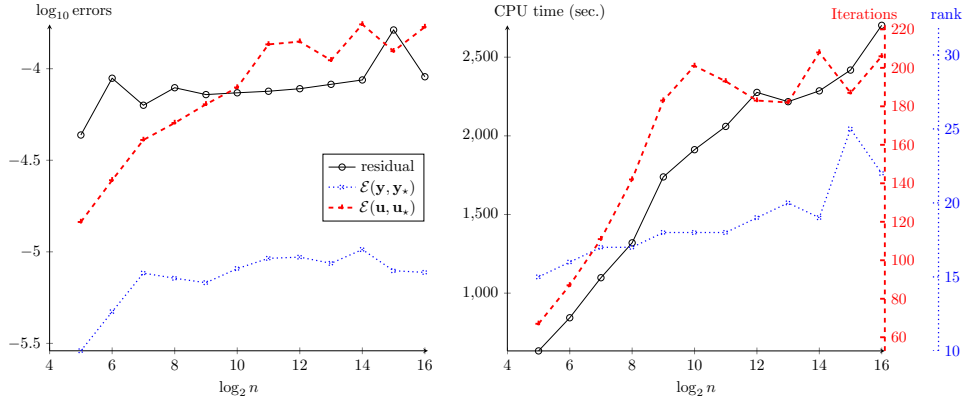
In the last test with the distance functional, we vary the low-rank approximation threshold, see Fig. 9. We confirm that all errors decay linearly with ε. Besides, we notice that all complexity indicators grow as $\log \varepsilon^{-1}$.

5.3 Optimization of a vorticity functional

In this section, we consider the case of the vorticity minimization. The default parameters are the same as in the previous section. The results are presented in the same layout. Fig. 10 shows the performance of the scheme with respect to the number of time steps n, Fig. 11 corresponds to the time interval T, in Fig. 12 we vary the control regularization β, Fig. 13 considers different spatial grid steps h, Fig. 14 shows the role of the viscosity ν, and Fig. 15 concerns variation of the truncation threshold ε.

The behavior of the method for the vorticity minimization is highly similar to the case of the misfit minimization. Here we outline the main differences. First, the solution with minimal vorticity exhibits smaller ranks than the Stokes solution: 10–15 versus 20–30 in the previous section. This leads to smaller computational times. The velocity error is also smaller: here it remains on the level 10^{-6}, compared to 10^{-5} for the misfit functional. The control error and the residual follow the same

Figure 4: Tracking functional, n varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



trends as in the previous case. The Laplace operator in the observation matrix $\mathcal{M}_1$ leads to higher condition numbers of the KKT matrix. For example, in Fig. 13, the number of iterations begins to grow for spatial grids finer than $h = 1/32$, while the misfit optimization demonstrated the opposite situation. The same holds true for the viscosity Fig. 14: with $\nu \sim 1$, the spectra of matrices $\mathcal{M}_1$ and $\mathcal{K}$ come closer to each other, and the efficiency of the preconditioner deteriorates. Variation of ε also reveals that the error estimates decay linearly with ε, which confirms their consistency.

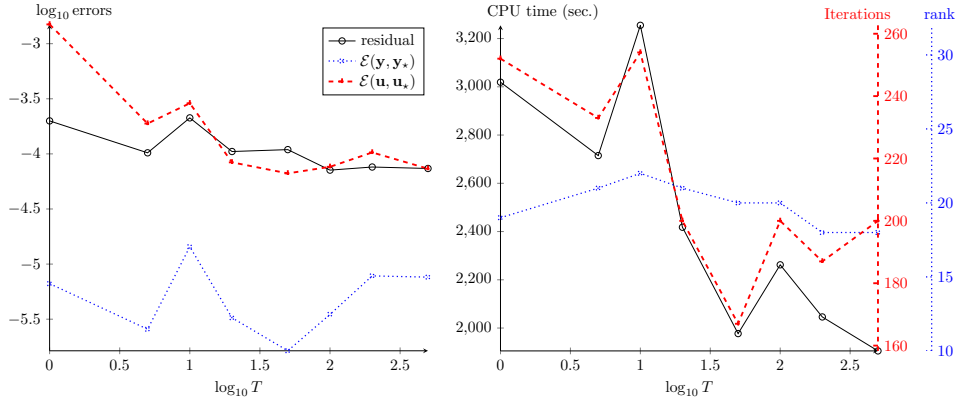
To see how the vorticity minimization influences the behavior of the fluid, we show the vorticity plots in Fig. 16. In the left plane we show the snapshots of the solution of the forward Navier-Stokes equations with the default parameters, taken at $t = 12$ and $t = 200$, in the right plane we show the solution of the optimal control problem at the same time steps. We see that the flow becomes much less turbulent when the control is employed.

From Fig. 12 we can also note that the norm of the curl is almost independent on β in the considered range. It might become larger with larger β, but we were unable to compute such tests due to the convergence issue mentioned in Sec. 5.2.4.

6 Conclusions

We have shown in this paper that a low-rank approach for solving the optimal control problem subject to the Navier-Stokes equations is possible. In order to achieve this we have established the low-rank formulation for two different objective functions and then introduced a schemes that utilizes the low-rank nature of the desired state to carry this low-rank through an alternating iteration procedure. For this we had to rely on efficient tensor techniques in combination with sophisticated spatial preconditioners for Navier-Stokes systems. We further establish existence results for the low-rank solutions to the Stokes equations. In our numerical results we have performed a parameter study

Figure 5: Tracking functional, T varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



with respect to the convergence of our proposed scheme. We showed that our method is robust with respect to parameter changes while maintaining a consistent low rank for even large-scale setups.

References

[1] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings, *A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids*, Journal of Non-Newtonian Fluid Mechanics, 139 (2006), pp. 153 – 176.

[2] 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*, Journal of Fluid Mechanics, 729 (2013), pp. 285–308.

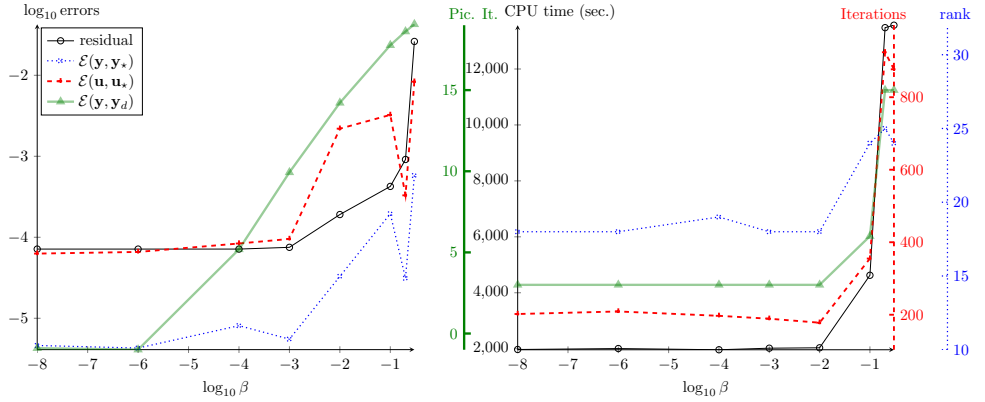
[3] J. Ballani and L. Grasedyck, *A projection method to solve linear systems in tensor format*, Numerical Linear Algebra with Applications, 20 (2013), pp. 27–43.

[4] P. Benner and T. Breiten, *Low rank methods for a class of generalized lyapunov equations and related issues*, Numerische Mathematik, 124 (2013), pp. 441–470.

[5] P. Benner, S. Dolgov, A. Onwunta, and M. Stoll, *Low-rank solvers for unsteady stokes-brinkman optimal control problem with random data*, MPI Magdeburg Preprint 15-10, 2015.

[6] P. Benner, J.-R. Li, and T. Penzl, *Numerical solution of large-scale lyapunov equations, riccati equations, and linear-quadratic optimal control problems*, Numerical Linear Algebra with Applications, 15 (2008), pp. 755–777.

Figure 6: Tracking functional, β varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



[7] L. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, *Large-Scale PDE-constrained Optimization: An Introduction*, Large-scale PDE-constrained optimization, (2003), p. 3.

[8] A. Borzì and R. Griesse, *Experiences with a space-time multigrid method for the optimal control of a chemical turbulence model*, Int. J. Numer. Methods Fluids, 47 (2005), pp. 879–885.

[9] A. Borzì and V. Schulz, *Multigrid methods for PDE optimization.*, SIAM Rev., 51 (2009), pp. 361–395.

[10] J. Burkardt, M. Gunzburger, and H.-C. Lee, *POD and cvt-based reduced-order modeling of Navier-Stokes flows*, Computer Methods in Applied Mechanics and Engineering, 196 (2006), pp. 337 – 355.

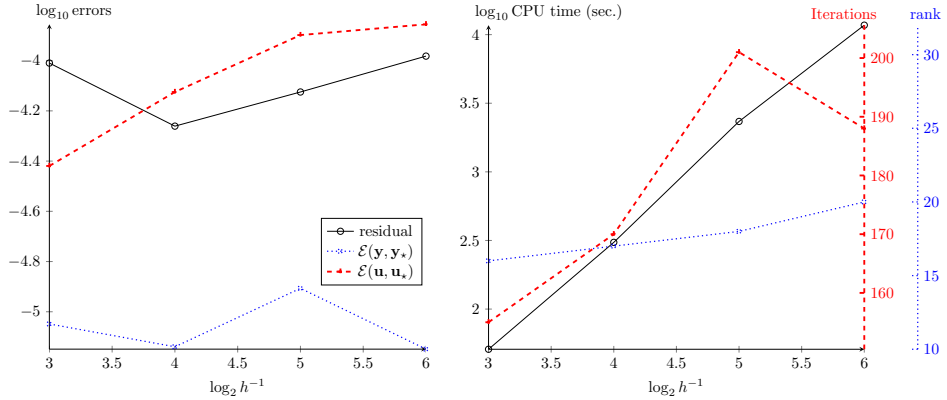
[11] S. V. Dolgov and D. V. Savostyanov, *Alternating minimal energy methods for linear systems in higher dimensions*, SIAM J Sci. Comput., 36 (2014), pp. A2248–A2271.

[12] S. V. Dolgov, A. P. Smirnov, and E. E. Tyrtyshnikov, *Low-rank approximation in the numerical modeling of the Farley-Buneman instability in ionospheric plasma*, J. Comp. Phys., 263 (2014), pp. 268–282.

[13] H. C. Elman, A. Ramage, and D. J. Silvester, *Algorithm 886: IFISS, a Matlab toolbox for modelling incompressible flow*, ACM Trans. Math. Software, 33 (2007), pp. Art. 14, 18.

[14] H. C. Elman, D. Silvester, and A. Wathen, *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*, Oxford University Press, 2014.

Figure 7: Tracking functional, h varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



[15] L. Grasedyck, *Existence and computation of low Kronecker-rank approximations for large systems in tensor product structure*, Computing, 72 (2004), pp. 247–265.

[16] L. Grasedyck, D. Kressner, and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitt., 36 (2013), pp. 53–78.

[17] M. D. Gunzburger and S. Manservisi, *Analysis and approximation of the velocity tracking problem for navier–stokes flows with distributed control*, SIAM Journal on Numerical Analysis, 37 (2000), pp. 1481–1512.

[18] W. Hackbusch, *Tensor Spaces And Numerical Tensor Calculus*, Springer–Verlag, Berlin, 2012.

[19] W. Hackbusch and B. N. Khoromskij, *Low-rank Kronecker-product approximation to multi-dimensional nonlocal operators. II. HKT representation of certain operators*, Computing, 76 (2006), pp. 203–225.

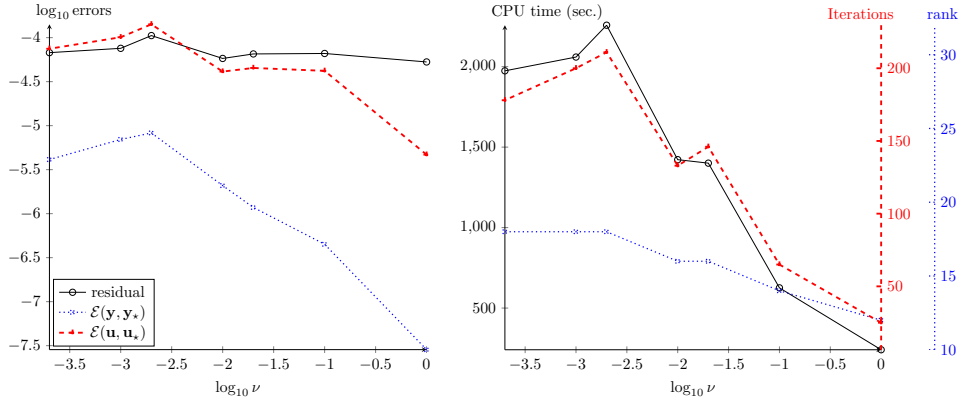
[20] W. Hackbusch and S. Kühn, *A new scheme for the tensor representation*, J. Fourier Anal. Appl., 15 (2009), pp. 706–722.

[21] R. Herzog and E. W. Sachs, *Preconditioned conjugate gradient method for optimal control problems with control and state constraints*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2291–2317.

[22] M. Hinze, *Optimal and instantaneous control of the instationary Navier-Stokes equations*, Habilitation, TU Berlin, 2000.

[23] M. Hinze, M. Köster, and S. Turek, *A Hierarchical Space-Time Solver for Distributed Control of the Stokes Equation*, tech. rep., SPP1253-16-01, 2008.

Figure 8: Tracking functional, ν varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



[24] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE Constraints*, Mathematical Modelling: Theory and Applications, Springer-Verlag, New York, 2009.

[25] S. Holtz, T. Rohwedder, and R. Schneider, *The alternating linear scheme for tensor optimization in the tensor train format*, SIAM J. Sci. Comput., 34 (2012), pp. A683–A713.

[26] T. Huckle and K. Waldherr, *Subspace iteration method in terms of matrix product states*, Proc. Appl. Math. Mech., 12 (2012), pp. 641–642.

[27] K. Ito and K. Kunisch, *Lagrange multiplier approach to variational problems and applications*, vol. 15 of Advances in Design and Control, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.

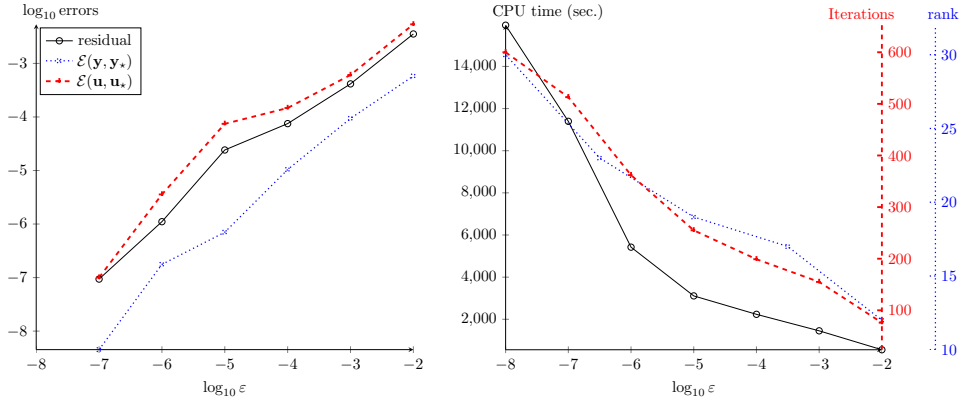
[28] K. Ito, K. Kunisch, V. Schulz, and I. Gherman, *Approximate nullspace iterations for kkt systems*, SIAM Journal on Matrix Analysis and Applications, 31 (2010), pp. 1835–1847.

[29] H. Kasumba and K. Kunisch, *Vortex control in channel flows using translational invariant cost functionals*, Computational Optimization and Applications, 52 (2012), pp. 691–717.

[30] V. Khoromskaia, *Black-box Hartree–Fock solver by tensor numerical methods*, Computational Methods in Applied Mathematics, 14 (2014), pp. 89–111.

[31] B. N. Khoromskij, *$\mathcal{O}(d \log n)$–Quantics approximation of N–d tensors in high-dimensional numerical modeling*, Constr. Approx., 34 (2011), pp. 257–280.

Figure 9: Tracking functional, ε varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



[32] B. N. Khoromskij, *Tensor numerical methods for high-dimensional PDEs: basic theory and initial applications*, arXiv preprint 1409.7970, 2014. to appear in ESAIM: Proceedings.

[33] K. Kormann, *A semi-lagrangian vlasov solver in tensor train format*, arXiv preprint 1408.7006, 2014.

[34] D. Kressner and C. Tobler, *Krylov subspace methods for linear systems with tensor product structure*, SIAM J. Matrix Anal. Appl, 31 (2010), pp. 1688–1714.

[35] P. Kroonenberg and J. de Leeuw, *Principal component analysis of three-mode data by means of alternating least squares algorithms*, Psychometrika, 45 (1980), pp. 69–97.

[36] K. Kunisch and B. Vexler, *Optimal vortex reduction for instationary flows based on translation invariant cost functionals*, SIAM Journal on Control and Optimization, 46 (2007), pp. 1368–1397.

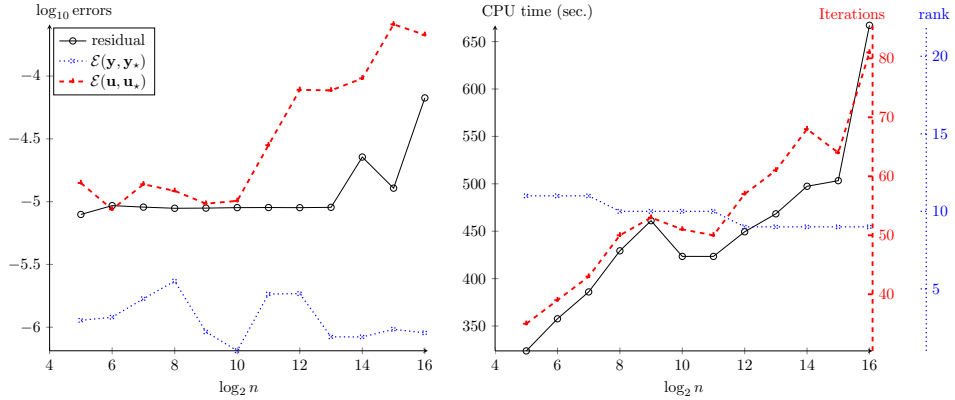
[37] K. Kunisch and S. Volkwein, *Galerkin POD methods for parabolic problems*, Numerische Mathematik, 90 (2001), pp. 117–148.

[38] T. Mach and J. Saak, *Towards an ADI iteration for tensor structured equations*, MPI Magdeburg Preprint 2011-12, 2011.

[39] K. Mardal and R. Winther, *Preconditioning discretizations of systems of partial differential equations*, Numerical Linear Algebra with Applications, 18 (2011), pp. 1–40.

[40] B. R. Noack, P. Papas, and P. A. Monkewitz, *The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows*, Journal of Fluid Mechanics, 523 (2005), pp. 339–365.

Figure 10: Vorticity functional, n varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



[41] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer Series in Operations Research, Springer-Verlag, New York, 1999.

[42] A. Nouy, *Proper generalized decompositions and separated representations for the numerical solution of high dimensional stochastic problems*, Archives of Computational Methods in Engineering, 17 (2010), pp. 403–434.

[43] I. V. Oseledets, *Approximation of $2^d \times 2^d$ matrices using tensor decomposition*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2130–2145.

[44] ——, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317.

[45] I. V. Oseledets, S. Dolgov, V. Kazeev, D. Savostyanov, O. Lebedeva, P. Zhlobich, T. Mach, and L. Song, *TT-Toolbox*. https://github.com/oseledets/TT-Toolbox.

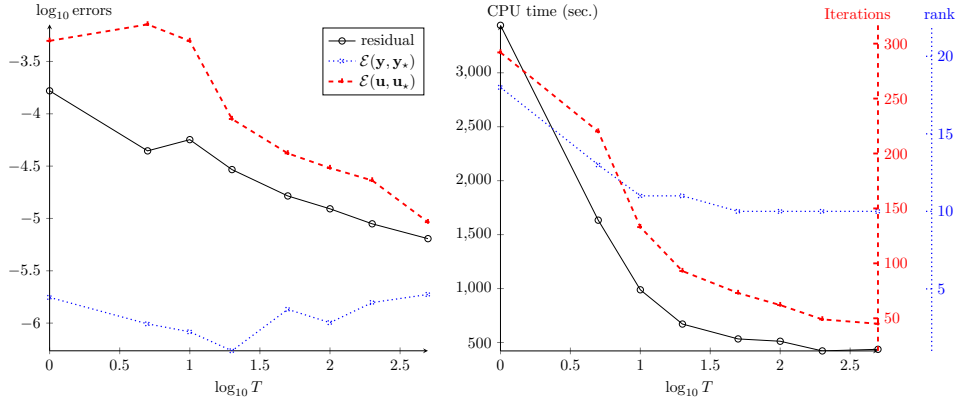
[46] C. C. Paige and M. A. Saunders, *Solutions of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal, 12 (1975), pp. 617–629.

[47] J. W. Pearson, *Preconditioned iterative methods for navier-stokes control problems*, (2013).

[48] J. W. Pearson, M. Stoll, and A. J. Wathen, *Regularization-robust preconditioners for time-dependent PDE-constrained optimization problems*, SIAM J. Matrix Anal.Appl, 33 (2012), pp. 1126–1152.

[49] M. V. Rakhuba and I. V. Oseledets, *Grid-based electronic structure calculations: the tensor decomposition approach*, arXiv preprint 1508.07632, 2015.

Figure 11: Vorticity functional, T varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.



[50] T. Rohwedder and A. Uschmajew, *On local convergence of alternating schemes for optimization of convex problems in the tensor train format*, SIAM J. Num. Anal., 51 (2013), pp. 1134–1162.

[51] M. Stoll and T. Breiten, *A low-rank in time approach to PDE-constrained optimization*, SIAM Journal on Scientific Computing, 37 (2015), pp. B1–B29.

[52] M. Stoll and A. Wathen, *All-at-once solution of time-dependent Stokes control*, Journal of Computational Physics, 232 (2013), pp. 498–515.

[53] V. Temlyakov, *Greedy Approximation*, Cambridge University Press, 2011.

[54] F. Tröltzsch, *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*, Amer Mathematical Society, 2010.

[55] M. Ulbrich, *Semismooth Newton Methods for Variational Inequalities and Constrained Optimization Problems*, SIAM Philadelphia, 2011.

[56] E. L. Wachspress, *The ADI Model Problem*, Springer, New York, 2013.

[57] S. R. White, *Density-matrix algorithms for quantum renormalization groups*, Phys. Rev. B, 48 (1993), pp. 10345–10356.

Figure 12: Vorticity functional, β varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.

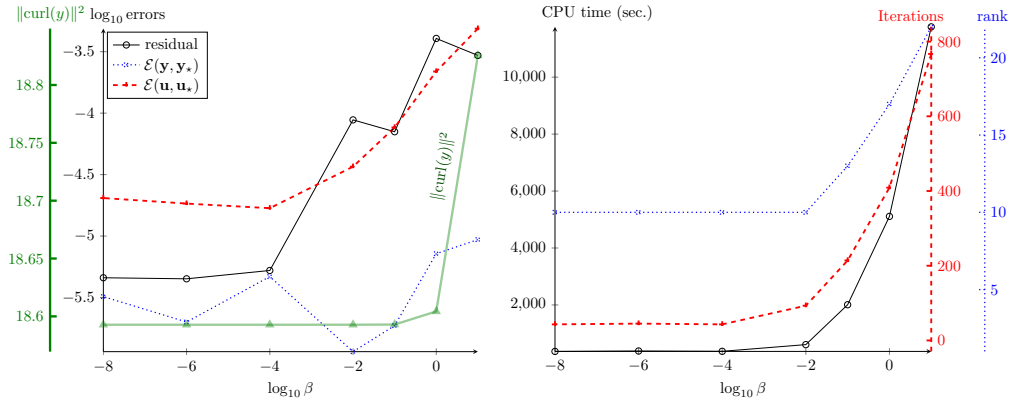


Figure 13: Vorticity functional, h varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.

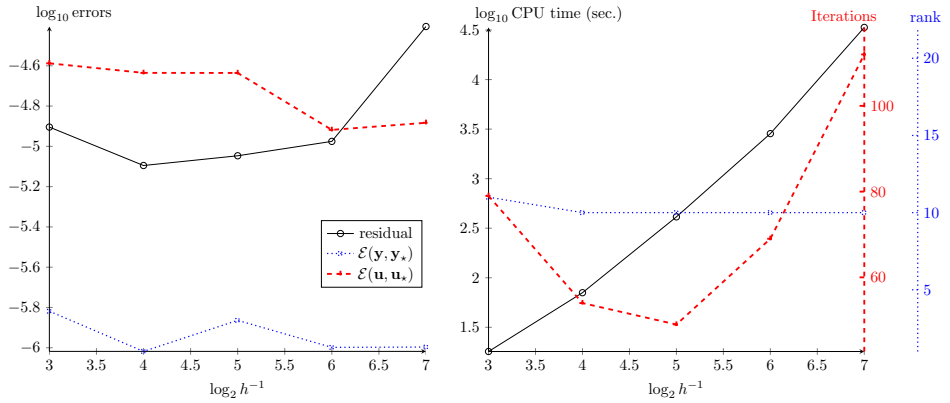


Figure 14: Vorticity functional, ν varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.

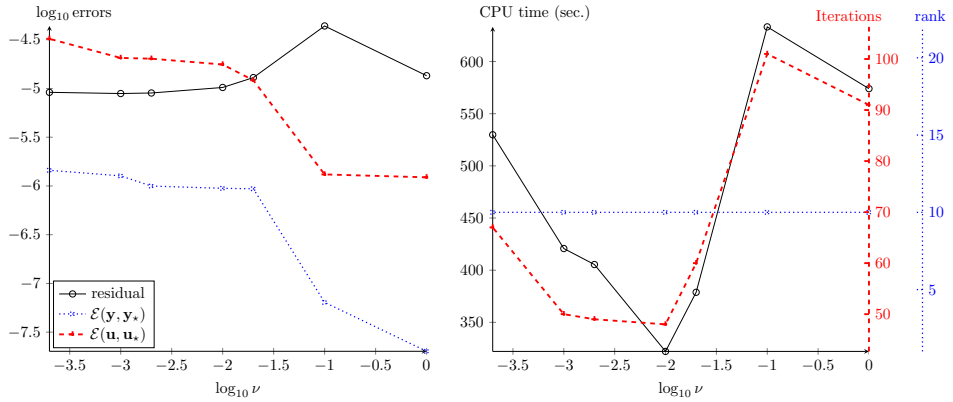


Figure 15: Vorticity functional, ε varies. Left: Residual and errors w.r.t. the reference solutions. Right: CPU time, total number of local iterations, rank.

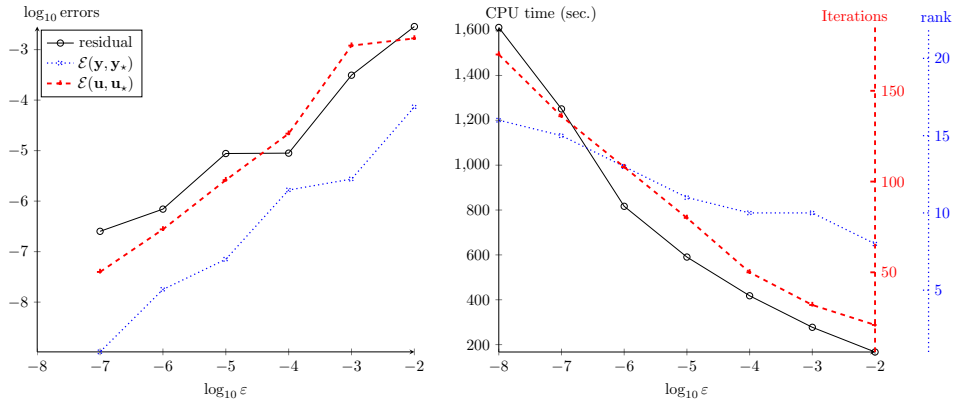


Figure 16: Vorticity plots at $t = 12$ (top) and $t = 200$ (bottom) for the forward (uncontrolled) Navier-Stokes system (left), and the controlled system with the minimized vorticity (right).