SIAM J. SCI. COMPUT.
Vol. 37, No. 1, pp. B1–B29

A LOW-RANK IN TIME APPROACH TO PDE-CONSTRAINED OPTIMIZATION[*]

MARTIN STOLL[†] AND TOBIAS BREITEN[‡]

Abstract. The solution of time-dependent PDE-constrained optimization problems is a challenging task in numerical analysis and applied mathematics. All-at-once discretizations and corresponding solvers provide efficient methods to robustly solve the arising discretized equations. One of the drawbacks of this approach is the high storage demand for the vectors representing the discrete space-time cylinder. Here we introduce a low-rank in time technique that exploits the low-rank nature of the solution. The theoretical foundations for this approach originate in the numerical treatment of matrix equations and can be carried over to PDE-constrained optimization. We illustrate how three different problems can be rewritten and used within a low-rank Krylov subspace solver with appropriate preconditioning.

Key words. PDE-constrained optimization, low-rank methods, space-time methods, preconditioning, Schur-complement, matrix equations

AMS subject classifications. 65F08, 65F10, 65F50, 92E20, 93C20

DOI. 10.1137/130926365

1. Introduction.

Many complex phenomena in the natural, engineering, and life sciences are modeled using partial differential equations (PDEs). To obtain optimal configurations of these equations one typically formulates this as a PDE-constrained optimization problem of the form

$$\min \mathcal{J}(y,u)$$

subject to

$$\mathcal{L}(y,u)=0$$

with $\mathcal{J}(y,u)$ the functional of interest and $\mathcal{L}(y,u)$ representing the differential operator. Problems of this type have been carefully analyzed in the past (see [49, 84] and the references therein).

Recently with the advancement of algorithms and technology, research has focused on the efficient numerical solution of these problems. In this paper we focus on the efficient solution of the discretized first order conditions in a space-time framework. The KKT conditions when considered in an all-at-once approach, i.e., simultaneous discretization in space and time, are typically of vast dimensionality. Matrix-free approaches have recently been developed to guarantee the (nearly) optimal convergence of iterative Krylov subspace solvers. The focus for both steady [68, 73] and transient problems [61, 81] has been on the development of efficient preconditioning

[*]Submitted to the journal's Computational Methods in Science and Engineering section June 26, 2013; accepted for publication (in revised form) October 16, 2014; published electronically January 8, 2015.
http://www.siam.org/journals/sisc/37-1/92636.html

[†]Numerical Linear Algebra for Dynamical Systems, Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstr. 1, 39106 Magdeburg, Germany (stollm@mpi-magdeburg.mpg.de).
[‡]Institute for Mathematics and Scientific Computing, Heinrichstr. 36/III, University of Graz, Austria (tobias.breiten@uni-graz.at). Most of this work was completed while this author was with the Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg.

strategies for the linear system that typically are of structured form (see [16, 27] for introductions to the numerical solution of saddle point systems).

One of the obstacles using a space-time discretization is the storage requirement for the large vectors needed to represent the solution at all times. Approaches such as checkpointing [38] or multiple shooting [42] are possible methods to solve these problems. Here we want to introduce an alternative to these schemes that can for certain problems provide an efficient representation with a minimal amount of storage. We are basing our methodology on recent developments within the solution of large and sparse matrix equations; see, e.g., [4, 13, 25, 28, 36, 50, 51, 54, 70, 74, 77, 85] and references therein. One classical representative in this category is the Lyapunov equation

$$AX + XA^T = -\tilde{C}\tilde{C}^T,$$

where we are interested in approximating the matrix-valued unknown X. Solving this system is equivalent to solving the linear system

$$(I \otimes A + A \otimes I)x = \tilde{c},$$

where x and $\tilde{c}$ are related to X and $\tilde{C}\tilde{C}^T$, respectively. For details on the relevance of this equation within control theory, see [3, 44, 52]. In [15, 56, 64, 65, 70] the authors have introduced low-rank iterative schemes that approximate intermediate iterates X_k in a low-rank fashion that is maintained until convergence. We can exploit these technologies for problems coming from PDE-constrained optimization. It is not expected that these techniques outperform optimal solvers with only a few time-steps. The more crucial component is that they enable computations with many time-steps that would otherwise not be possible.

The paper is structured as follows. In section 2 we introduce the heat equation as our model problem and discuss its discretization. Section 3 illustrates how this problem can be reformulated using Kronecker technology and how we need to adapt a standard Krylov-subspace solver to be able to solve this problem efficiently. As we need a preconditioner for fast convergence we next discuss possible preconditioners in section 3. We provide some theoretical results in section 4. Section 5 is devoted to illustrating that our methodology can be carried over to other state equations such as Stokes equations and the convection-diffusion equation. Finally, in section 6 we illustrate the competitiveness of our approach.

2. A PDE-constrained optimization model problem.

We start the derivation of the low-rank in time method by considering an often used model problem in PDE-constrained optimization (see [47, 49, 84]) that nevertheless reflects the crucial structure exhibited by many problems of similar type. Our goal is the minimization of a misfit functional that aims at bringing the state y as close as possible to a desired or observed state y_{obs} while using a control u, i.e.,

$$\min_{y,u} \frac{1}{2}\|y - y_{obs}\|^2_{L_2(\Omega_1)} + \frac{\beta}{2}\|u\|^2_{L_2(\Omega_2)}, \tag{2.1}$$

subject to a partial differential equation that connects both state and control, referred to as the state equation. We start by considering the heat equation with a distributed control term,

$$\begin{aligned} y_t - \nabla^2 y &= u \quad \text{in } \Omega, \\ y &= f \quad \text{on } \partial\Omega, \end{aligned} \tag{2.2}$$

or equipped with Neumann-boundary control,

$$\begin{aligned} y_t - \nabla^2 y &= f \quad \text{in } \Omega, \\ \frac{\partial y}{\partial \mathrm{n}} &= u \quad \text{on } \partial\Omega. \end{aligned} \tag{2.3}$$

For a more detailed discussion on the well-posedness, existence of solutions, etc., we refer the interested reader to [47, 49, 84]. Classically these problems are solved using a Lagrangian to incorporate the constraints and then consider the first order optimality conditions or KKT conditions [49, 58, 84]. This can be done either by forming a discrete Lagrangian and then performing the optimization procedure or by first considering an infinite-dimensional Lagrangian for whose first order conditions we employ a suitable discretization. Here we perform the first approach, although much of what we state in this paper is valid for both cases. Our goal is to build a discrete Lagrangian using an all-at-once approach [61, 81, 40] using a discrete problem within the space-time cylinder $\Omega \times [0, T]$. Using the trapezoidal rule in time and finite elements in space leads to the discrete objective function

$$J(y,u) = \frac{\tau}{2}\left(y - y_{obs}\right)^T \mathcal{M}_1 \left(y - y_{obs}\right) + \frac{\tau\beta}{2} u^T \mathcal{M}_2 u \tag{2.4}$$

with $\mathcal{M}_1 = \text{blkdiag}(\frac{1}{2}M_1, M_1, \ldots, M_1, \frac{1}{2}M_1)$, $\mathcal{M}_2 = \text{blkdiag}(\frac{1}{2}M_2, M_2, \ldots, M_2, \frac{1}{2}M_2)$ being space-time matrices where M_1 is the mass matrix associated with the domain Ω_1 and M_2 is the corresponding mass matrix for Ω_2. The vectors $y = [y_1^T \ldots y_{n_t}^T]^T$ and $u = [u_1^T \ldots u_{n_t}^T]^T$ are of vast dimensionality and represent a collection of spatial vectors for all time-steps collected into one single vector.

The all-at-once discretization of the state equation using finite elements in space and an implicit Euler scheme in time is given by

$$\mathcal{K}y - \tau\mathcal{N}u = d, \tag{2.5}$$

where

$$\mathcal{K} = \begin{bmatrix} L & & & \\ -M & L & & \\ & \ddots & \ddots & \\ & & -M & L \end{bmatrix}, \quad \mathcal{N} = \begin{bmatrix} N & & & \\ & N & & \\ & & \ddots & \\ & & & N \end{bmatrix}, \quad d = \begin{bmatrix} M_1 y_0 + f \\ f \\ \vdots \\ f \end{bmatrix}.$$

Here, M is the mass matrix for the domain Ω, the matrix L is defined as $L = M + \tau K$, the matrix N represents the control term either via a distributed control (square matrix) or via the contributions of a boundary control problem (rectangular matrix), and the right-hand-side d consists of a contribution from the initial condition y_0 and a vector f representing forcing terms and contributions of boundary conditions. The first order conditions using a Lagrangian formulation with Lagrange multiplier p leads to the following system:

$$\underbrace{\begin{bmatrix} \tau\mathcal{M}_1 & 0 & -\mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_2 & \tau\mathcal{N}^T \\ -\mathcal{K} & \tau\mathcal{N} & 0 \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} y \\ u \\ p \end{bmatrix} = \begin{bmatrix} \tau\mathcal{M}_1 y_{obs} \\ 0 \\ d \end{bmatrix}. \tag{2.6}$$

Systems of this form have previously been studied in [81, 61, 82, 57]. As these systems are of vast dimensionality it is crucial to find appropriate preconditioners together with Krylov subspace solvers to efficiently obtain an approximation to the solution. The vast dimensionality of system matrices does not allow the use of direct solvers [26, 23] but we can employ Krylov subspace solvers in a matrix-free way by never forming the matrix $\mathcal{A}$ and only implicitly performing the matrix vector product. The main bottleneck of this approach is the storage requirement for the space-time vectors which can be reduced by working on the Schur-complement if it exists of the matrix $\mathcal{A}$ or removing the control from the system matrix [76, 45]. Other approaches that can be employed are checkpointing schemes [38] or multiple shooting approaches [42]. In the following we want to present an alternative that uses the underlying tensor structure of the first order conditions.

3. A Kronecker view.

We noticed earlier that the linear system in (2.6) is of vast dimensionality and that we need only very few matrices to efficiently perform the matrix vector multiplication with $\mathcal{A}$, and we can approach this in a matrix-free form by never forming $\mathcal{A}$. Nevertheless, the vectors y, u, and p themselves are enormous and every storage reduction would help to improve the performance of an optimization scheme. The goal now is to employ the structure of the linear system to reduce the storage requirement for the iterative method. Our approach is based on recent developments for matrix equations [10, 54, 36]. Using the definition of the Kronecker product

$$W \otimes V = \begin{bmatrix} w_{11}V & \dots & w_{1m}V \\ \vdots & \ddots & \vdots \\ w_{n1}V & \dots & w_{nm}V \end{bmatrix}$$

we note that (2.6) can also be written as

$$\underbrace{\begin{bmatrix} D_1 \otimes \tau M_1 & 0 & -\left(I_{n_t} \otimes L + C^T \otimes M\right) \\ 0 & D_2 \otimes \beta\tau M_2 & D_3 \otimes \tau N^T \\ -\left(I_{n_t} \otimes L + C \otimes M\right) & D_3 \otimes \tau N & 0 \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} y \\ u \\ p \end{bmatrix} \tag{3.1}$$

$$= \begin{bmatrix} D_1 \otimes \tau M_1 y_{obs} \\ 0 \\ d \end{bmatrix},$$

where $D_1 = D_2 = \operatorname{diag}\left(\frac{1}{2}, 1, \dots, 1, \frac{1}{2}\right)$ and $D_3 = I_{n_t}$. Additionally, the matrix $C \in \mathbb{R}^{n_t,n_t}$ is given by

$$C = \begin{bmatrix} 0 & & & \\ -1 & 0 & & \\ & \ddots & \ddots & \\ & & -1 & 0 \end{bmatrix}$$

and represents the implicit Euler scheme. It is of course possible to use a different discretization in time. So far we have simply reformulated the previously given system. But our goal was to derive a scheme that allows for a reduction in storage requirement for the vectors y, u, and p. For this we remind the reader of the definition of the vec operator via

$$\operatorname{vec}(W) = \begin{bmatrix} w_{11} \\ \vdots \\ w_{n1} \\ \vdots \\ w_{nm} \end{bmatrix}$$

as well as the relation

$$\left(W^T \otimes V\right) \operatorname{vec}(Y) = \operatorname{vec}(VYW).$$

Now employing this and using the notation

$$Y = [\ y_1, y_2, \dots, y_{n_t}\ ], \quad U = [\ u_1, u_2, \dots, u_{n_t}\ ], \quad P = [\ p_1, p_2, \dots, p_{n_t}\ ]$$

we get that

$$\begin{aligned}(3.2)\quad &\begin{bmatrix} D_1 \otimes \tau M_1 & 0 & -\left(I_{n_t} \otimes L + C^T \otimes M\right) \\ 0 & D_2 \otimes \beta\tau M_2 & D_3 \otimes \tau N^T \\ -\left(I_{n_t} \otimes L + C \otimes M\right) & D_3 \otimes \tau N & 0 \end{bmatrix} \begin{bmatrix} \operatorname{vec}(Y) \\ \operatorname{vec}(U) \\ \operatorname{vec}(P) \end{bmatrix} \\ &= \operatorname{vec}\left(\begin{bmatrix} \tau M_1 Y D_1^T - L P I_{n_t}^T - MPC \\ \tau\beta M_2 U D_2^T + \tau N^T P D_3^T \\ -L Y I_{n_t}^T - M Y C^T + \tau N U D_3^T \end{bmatrix} \right).\end{aligned}$$

So far nothing is gained from rewriting the problem in this form. As was previously done in [10] we assume for now that if Y, U, and P can be represented by a low-rank approximation, any iterative Krylov subspace solver can be implemented using a low-rank version of (3.2). We denote the low-rank representations by

$$(3.3)\quad Y = W_Y V_Y^T \text{ with } W_Y \in \mathbb{R}^{n_1, k_1}, V_Y \in \mathbb{R}^{n_t, k_1},$$

$$(3.4)\quad U = W_U V_U^T \text{ with } W_U \in \mathbb{R}^{n_2, k_2}, V_U \in \mathbb{R}^{n_t, k_2},$$

$$(3.5)\quad P = W_P V_P^T \text{ with } W_P \in \mathbb{R}^{n_1, k_3}, V_P \in \mathbb{R}^{n_t, k_3},$$

with $k_{1,2,3}$ being small in comparison to n_t, and we rewrite (3.2) accordingly to get

$$(3.6)\quad \begin{bmatrix} \tau M_1 W_Y V_Y^T D_1^T - L W_P V_P^T I_{n_t}^T - M W_P V_P^T C \\ \tau\beta M_2 W_U V_U^T D_2^T + \tau N^T W_P V_P^T D_3^T \\ -L W_Y V_Y^T I_{n_t}^T - M W_Y V_Y^T C^T + \tau N W_U V_U^T D_3^T \end{bmatrix},$$

where we skipped the vec operator and instead used matrix-valued unknowns. Note that we can write the block-rows of (3.6) as

$$\text{(first block-row)} \quad \begin{bmatrix} \tau M_1 W_Y & -L W_P & -M W_P \end{bmatrix} \begin{bmatrix} V_Y^T D_1^T \\ V_P^T I_{n_t}^T \\ V_P^T C \end{bmatrix},$$

$$(3.7)\quad \text{(second block-row)} \quad \begin{bmatrix} \tau\beta M_2 W_U & \tau N^T W_P \end{bmatrix} \begin{bmatrix} V_U^T D_2^T \\ V_P^T D_3^T \end{bmatrix},$$

$$\text{(third block-row)} \quad \begin{bmatrix} -L W_Y & -M W_Y & \tau N W_U \end{bmatrix} \begin{bmatrix} V_Y^T I_{n_t}^T \\ V_Y^T C^T \\ V_U^T D_3^T \end{bmatrix}.$$

We obtain a significant storage reduction if we can base our approximation of the solution using the low-rank factors (3.7). It is easily seen that due to the low-rank nature of the factors we have to perform fewer multiplications with the submatrices by also maintaining smaller storage requirements. As the usage of a direct solver is out of the question we here rely on a preconditioned Krylov subspace solver, namely, MINRES introduced in [59] as the underlying matrix is symmetric and indefinite. Before explaining all the intricacies of the method we state the resulting algorithm and carefully explain the necessary details afterward. Algorithm 1 shows a low-rank implementation of the classical preconditioned MINRES method as presented in [59]. Note that due to the truncation to low-rank the application of the preconditioner is not identical for every step of the iteration and the use of a flexible solver needs to be investigated in the future. Here we use a rather small truncation tolerance to try to maintain a very accurate representation of what the full-rank representation would look like.

It is hard to hide the fact that the low-rank version presented here seems much messier than its vector-based relative. This is due to the fact that we want to maintain the structure of the saddle point system, which is reflected in low-rank representations associated with the state (all matrices with indices 11 and 12), the control (all matrices with indices 21 and 22), and the Lagrange multiplier (all matrices with indices 31 and 32). Please keep in mind that

$$\operatorname{vec}\left(\begin{bmatrix} Z_{11}Z_{12}^T \\ Z_{21}Z_{22}^T \\ Z_{31}Z_{32}^T \end{bmatrix}\right) = z$$

corresponds to the associated vector z from a vector-based version of MINRES.

For Algorithm 1 to be accessible to the reader, we need to dissect its different parts. Starting with the inner products of the classical MINRES method we see that we can efficiently evaluate the inner product $\left(z^{(j)}, v^{(j)}\right)$. In more detail, we use

$$\operatorname{vec}\left(\begin{bmatrix} Z_{11}^{(j)}(Z_{12}^{(j)})^T \\ Z_{21}^{(j)}(Z_{22}^{(j)})^T \\ Z_{31}^{(j)}(Z_{32})^T \end{bmatrix}\right) = z^{(j)} \text{ and } \operatorname{vec}\left(\begin{bmatrix} V_{11}^{(j)}(V_{12}^{(j)})^T \\ V_{21}^{(j)}(V_{22}^{(j)})^T \\ V_{31}^{(j)}(V_{32})^T \end{bmatrix}\right) = v^{(j)}$$

and the relation for the trace

$$\operatorname{trace}\left(A^T B\right) = \operatorname{vec}(A)^T \operatorname{vec}(B)$$

to compute the inner product $\left(z^{(j)}, v^{(j)}\right)$ (for convenience ignoring the index j) via

$$\begin{aligned} \left(z^{(j)}, v^{(j)}\right) &= \operatorname{trace}\left(\left(Z_{11}Z_{12}^T\right)^T \left(V_{11}V_{12}^T\right)\right) \\ &\quad + \operatorname{trace}\left(\left(Z_{21}Z_{22}^T\right)^T \left(V_{21}V_{22}^T\right)\right) \\ &\quad + \operatorname{trace}\left(\left(Z_{31}Z_{32}^T\right)^T \left(V_{31}V_{32}^T\right)\right), \end{aligned} \tag{3.8}$$

where $z^{(j)}$ and $v^{(j)}$ are the vectorization of the stacked V and Z matrices. Note that so far we have rewritten the vector problem in matrix form, but the interested reader might have noted that the matrices formed as part of (3.8) are of the full dimensionality $n \times n_t$ in the case of a distributed control problem. Due to the properties of the trace operator we are in luck as

$$\operatorname{trace}\left(\left(Z_{11}Z_{12}^T\right)^T \left(V_{11}V_{12}^T\right)\right) = \operatorname{trace}\left(Z_{11}^T V_{11} V_{12}^T Z_{12}\right)$$

Algorithm 1: Low-rank MINRES.

```
Zero-Initiliazation of $V_{11}^{(0)},\ldots,W_{11}^{(0)},\ldots,$ and $W_{11}^{(1)},\ldots.$
Choose $U_{11}^{(0)},U_{12}^{(0)},U_{21}^{(0)},U_{22}^{(0)},U_{31}^{(0)},U_{32}^{(0)}$
Set $V_{11},V_{12},\ldots$ to normalized residual
while residual norm > tolerance do
    $Z_{11}^{(j)}=Z_{11}^{(j)}/\gamma_j,\ Z_{21}^{(j)}=Z_{21}^{(j)}/\gamma_j,\ Z_{31}^{(j)}=Z_{31}^{(j)}/\gamma_j,$
    $[F_{11},F_{12},F_{21},F_{22},F_{31},F_{32}]=\texttt{Amult}(Z_{11}^{(j)},Z_{12}^{(j)},Z_{21}^{(j)},Z_{22}^{(j)},Z_{31}^{(j)},Z_{32}^{(j)})$
    $\delta_j=\texttt{traceproduct}(F_{11},F_{12},F_{21},F_{22},F_{31},F_{32},Z_{11}^{(j)},Z_{12}^{(j)},Z_{21}^{(j)},Z_{22}^{(j)},Z_{31}^{(j)},Z_{32}^{(j)})$
    $V_{11}^{(j+1)}=\left\{F_{11}\quad -\frac{\delta_j}{\gamma_j}V_{11}^{(j)}\quad -\frac{\gamma_j}{\gamma_{j-1}}V_{11}^{(j-1)}\right\},\quad V_{12}^{(j+1)}=\left\{F_{12}\quad V_{12}^{(j)}\quad V_{12}^{(j-1)}\right\}$
    $V_{21}^{(j+1)}=\left\{F_{21}\quad -\frac{\delta_j}{\gamma_j}V_{21}^{(j)}\quad -\frac{\gamma_j}{\gamma_{j-1}}V_{21}^{(j-1)}\right\},\quad V_{22}^{(j+1)}=\left\{F_{22}\quad V_{22}^{(j)}\quad V_{22}^{(j-1)}\right\}$
    $V_{31}^{(j+1)}=\left\{F_{31}\quad -\frac{\delta_j}{\gamma_j}V_{31}^{(j)}\quad -\frac{\gamma_j}{\gamma_{j-1}}V_{31}^{(j-1)}\right\},\quad V_{32}^{(j+1)}=\left\{F_{32}\quad V_{32}^{(j)}\quad V_{32}^{(j-1)}\right\}$
    $\left\{Z_{11}^{(j+1)},Z_{12}^{(j+1)},Z_{21}^{(j+1)},Z_{22}^{(j+1)},Z_{31}^{(j+1)},Z_{32}^{(j+1)}\right\}=$
    $\texttt{Aprec}(V_{11}^{(j+1)},V_{12}^{(j+1)},V_{21}^{(j+1)},V_{22}^{(j+1)},V_{31}^{(j+1)},V_{32}^{(j+1)})$
    $\gamma_{j+1}=\sqrt{\texttt{tracepoduct}(Z_{11}^{(j+1)},\ldots,V_{11}^{(j+1)},\ldots)}$
    $\alpha_0=c_j\delta_j-c_{j-1}s_j\gamma_j$
    $\alpha_1=\sqrt{\alpha_0^2+\gamma_{j+1}^2}$
    $\alpha_2=s_j\delta_j+c_{j-1}c_j\gamma_j$
    $\alpha_3=s_{j-1}\gamma_j$
    $c_{j+1}=\frac{\alpha_0}{\alpha_1}$
    $s_{j+1}=\frac{\gamma_{j+1}}{\alpha_1}$
    $W_{11}^{(j+1)}=\left\{Z_{11}^{(j)}\quad -\alpha_3W_{11}^{(j-1)}\quad -\alpha_2W_{11}^{(j)}\right\},\quad W_{12}^{(j+1)}=\left\{Z_{12}^{(j)}\quad W_{12}^{(j-1)}\quad W_{12}^{(j)}\right\}$
    $W_{21}^{(j+1)}=\left\{Z_{21}^{(j)}\quad -\alpha_3W_{21}^{(j-1)}\quad -\alpha_2W_{21}^{(j)}\right\},\quad W_{22}^{(j+1)}=\left\{Z_{22}^{(j)}\quad W_{22}^{(j-1)}\quad W_{22}^{(j)}\right\}$
    $W_{31}^{(j+1)}=\left\{Z_{31}^{(j)}\quad -\alpha_3W_{31}^{(j-1)}\quad -\alpha_2W_{31}^{(j)}\right\},\quad W_{32}^{(j+1)}=\left\{Z_{32}^{(j)}\quad W_{32}^{(j-1)}\quad W_{32}^{(j)}\right\}$
    if Convergence criterion fulfilled then
        Compute approximate solution
        stop
    end if
end while
```

allows us to to compute the trace of small matrices rather than of the ones from the full temporal/spatial discretization. We denote the reformulation of the trace in Algorithm 1 by the term `tracepoduct`.

We have now defined the matrix vector multiplication denoted by `Amult` in Algorithm 1 and shown in detail in Algorithm 2 as well as the efficient computation of the inner products within the low-rank MINRES algorithm. We have not yet defined the brackets $\{\}$. The brackets $U := \{U_1 \quad V_1 \quad W_1\}$ and $\{U_2 \quad V_2 \quad W_2\}$ can be understood as a concatenation and truncation by the way of an abstract function `trunc` that takes as inputs the matrices $[U_1 \quad V_1 \quad W_1]$ and $[U_2 \quad V_2 \quad W_2]$ and gives back low-rank approximations to these matrices, i.e., $\tilde{Z}_1 \approx [U_1 \quad V_1 \quad W_1]$ and $\tilde{Z}_2 \approx [U_2 \quad V_2 \quad W_2]$. We now briefly discuss how the `trunc` function could be designed.

We want to perform the truncation of two matrices V and U that represent the low-rank representation of $Z = VU^T$. As discussed in [54] we can perform skinny QR factorizations of both matrices, i.e., $V = Q_vR_v$ and $U = Q_uR_u$. We then note that $Z = Q_vR_vR_u^TQ_u^T$. A singular value decomposition [32] of the matrix

```
Algorithm 2: Matrix multiplication: Amult.
    Input: $W_{11}, W_{12}, W_{21}, W_{22}, W_{31}, W_{32}$
    Output: $Z_{11}, Z_{12}, Z_{21}, Z_{22}, Z_{31}, Z_{32}$
    $Z_{11} = \begin{bmatrix} \tau M_1 W_{11} & -LW_{31} & -MW_{31} \end{bmatrix}$
    $Z_{21} = \begin{bmatrix} \tau\beta M_2 W_{21} & \tau N W_{31} \end{bmatrix}$
    $Z_{31} = \begin{bmatrix} -LW_{11} & -MW_{31} & \tau N W_{21} \end{bmatrix}$
    $Z_{12} = \begin{bmatrix} D_1 W_{12} & I_{n_t} W_{32} & C^T W_{32} \end{bmatrix}$
    $Z_{22} = \begin{bmatrix} D_2 W_{22} & D_3 W_{32} \end{bmatrix}$
    $Z_{32} = \begin{bmatrix} I_{n_t} W_{12} & C W_{12} & D_3 W_{22} \end{bmatrix}$
```

$R_v R_u^T = B\Sigma C^T$ provides the means to reduce the rank by dropping small (depending on some tolerance) singular values. Using MATLAB notation we get a low-rank approximation via the truncated expression $B(:,1:k)\Sigma(1:k,1:k)C(:,1:k)^T$. This leads to the overall low-rank approximation $V_{new} = Q_v B(:,1:k)$ and $U_{new} = Q_v C(:,1:k)\Sigma(1:k,1:k)$, which in turn gives $Z \approx V_{new}U_{new}^T$. We have implemented this approach in MATLAB but noted that the computation of the skinny QR factorization was rather slow. Alternatively, we exploited the MATLAB function `svds` to directly compute a truncated singular value decomposition of VU^T by passing a function handle that allowed the implicit application of the $Z = VU^T$ without ever forming this matrix. This approach proved advantageous in terms of the time needed for the truncation. Note that alternative ways to compute the truncated SVD are of course possible [48, 6, 79].

Before discussing the possible preconditioners, employed via the `Aprec` function in Algorithm 1, we state that the vector update formulas given in Algorithm 1 are straightforward versions of vector versions of MINRES.

We additionally want to briefly comment on some of the alternative approaches to the presented methodology. One can of course reduce the dimensionality by eliminating the control when possible and obtain a system that is still vast and can be cast using our low-rank methodology. The use of reduced Hessian approaches typically leads to a symmetric positive-definite system for which CG would be applicable. But these formulations usually involve the inverse of the discretized PDE and this means that in order to simply apply the system matrix to a vector one has to very accurately solve for the PDE, as otherwise the matrix vector product does not represent the original KKT system. We refer to [39] for more details. Many algorithms employ the checkpointing technique [38], where only snapshots in time are stored. This is often done when the KKT system is treated in a block-Gauss–Seidel fashion, i.e., solving adjoint PDE, gradient equation, and forward PDE in an alternating manner. For such an iteration convergence might be slow or without proper scaling the method might not converge at all. The multiple shooting approach presented in [42] is in spirit very similar to the full-rank system we introduced here. Heinkenschloss [42] introduced an augmented Lagrangian formulation that leads to a large-scale linear system than can be reordered to obtain a system similar to (2.6). Our approach implicitly picks the "correct" number of vectors needed to accurately represent the solution in time. On the other hand we are currently limited to a formulation that can be written in the tensor form shown above. This is not true for the full-space approach and techniques based on the checkpointing methodology.

While the storage requirements can be reduced dramatically we still need to precondition the linear systems as we still have to deal with possibly very large matrices from the discretization in space. We show in the next section that we can use many of the techniques from the full-order space-time system for the low-rank scheme.

Preconditioning for low-rank MINRES. The study of preconditioners for the optimal control subject to parabolic PDEs has recently seen developments that were aimed at providing robust performance with respect to the many system parameters such as mesh-size or regularization parameters (see [73, 61, 60, 53]). More results can be found in [40, 17] and for multigrid techniques we recommend [18] and the references therein. We start our derivation of suitable preconditioners based on an approach presented by Pearson and colleagues [61, 63], where we start with a block-diagonal preconditioner

$$\mathcal{P} = \begin{bmatrix} A_0 & & \\ & A_1 & \\ & & \hat{S} \end{bmatrix}. \tag{3.9}$$

Here $A_0 \approx \tau \mathcal{M}_1$ and $A_1 \approx \tau\beta\mathcal{M}_2$ are approximations to the upper left block of $\mathcal{A}$ and $\hat{S}$ is an approximation to the Schur-complement

$$S = \tau^{-1}\mathcal{K}\mathcal{M}_1^{-1}\mathcal{K}^T + \frac{\tau}{\beta}\mathcal{N}\mathcal{M}_2^{-1}\mathcal{N}^T.$$

One approximation that has proved to be very effective [61, 63] is of the form

$$\hat{S} = \tau^{-1}\left(\mathcal{K} + \hat{\mathcal{M}}\right)\mathcal{M}_1^{-1}\left(\mathcal{K} + \hat{\mathcal{M}}\right)^T,$$

where in the case of a distributed control problem the matrix $\hat{\mathcal{M}}$ is given by

$$\hat{\mathcal{M}} = \frac{\tau}{\sqrt{\beta}}\operatorname{blkdiag}(M, \ldots, M).$$

Note that for for simplicity we assumed $D_1 = D_2 = D_3 = I_{n_t}$ during the Schur-complement approximation. It is of course possible to obtain robust approximations for other choices, but they would make the presentation of the Schur-complement approximation less accessible (see [61] for details using different Ds). This approach will be the basis for the derivation of efficient preconditioners for the low-rank version of MINRES. For this we need the preconditioner $\mathcal{P}$ to maintain the low-rank structure as described in (3.7). Due to the nature of the upper left block of $\mathcal{A}$ given by

$$\begin{bmatrix} D_1 \otimes \tau M_1 & 0 \\ 0 & D_2 \otimes \beta\tau M_2 \end{bmatrix}$$

we see that an efficient preconditioner given, for example, by

$$\begin{bmatrix} D_1 \otimes \tau \hat{M}_1 & 0 \\ 0 & D_2 \otimes \beta\tau \hat{M}_2 \end{bmatrix},$$

where the mass matrices are approximated by the Chebyshev semi-iteration [86], will naturally maintain the desired structure. But what can be said about the Schur-complement S of the above system? Starting from the previously used approximation

$$\hat{\mathcal{S}} = \tau^{-1}\left(I_{n_t} \otimes \hat{L} + C \otimes M\right)\mathcal{M}_1^{-1}\left(I_{n_t} \otimes \hat{L} + C \otimes M\right)^T,$$

where $\hat{L} = ((1 + \frac{\tau}{\sqrt{\beta}})M_1 + \tau K)$, we see that there already exists an inherent tensor structure within this approximation. In [81] the authors observe that such a system can be easily solved as the matrix $(I_{n_t} \otimes \hat{L} + C \otimes M)$ is of block-triangular nature. This means one can sequentially pass through the vectors associated with each grid-point in time. For our purpose the block-triangular nature will not be sufficient to guarantee the low-rank preserving nature of our algorithm. In simple terms, a low-rank factorization in time does not allow for a temporal decoupling of the time-steps as the vectors for each time-step are not readily identified. In mathematical terms we can see that it is not possible to explicitly write down the inverse of $(I_{n_t} \otimes \hat{L} + C \otimes M)$. Our starting point is a Block–Jacobi version of the Schur-complement approximation. This procedure is motivated by the fact that we can simply write

$$\left(I_{n_t} \otimes \hat{L}\right)^{-1} = \left(I_{n_t} \otimes \hat{L}^{-1}\right).$$

The last expression assures us that this preconditioner applied to any vector $v = \operatorname{vec}\left(RS^T\right)$ can be written as

$$\operatorname{vec}\left(\hat{L}^{-1} R S^T I_{n_t}\right).$$

We can now simply use the Schur-complement approximation

$$\hat{\mathcal{S}} = \tau^{-1}\left(I_{n_t} \otimes \hat{L}\right)\mathcal{M}_1\left(I_{n_t} \otimes \hat{L}\right)^T$$

or when using $\hat{\mathcal{S}} = \tau^{-1}(I_{n_t} \otimes \hat{L} + C \otimes M)\,\mathcal{M}_1(I_{n_t} \otimes \hat{L} + C \otimes M)^T$ approximate the inverse of $(I_{n_t} \otimes \hat{L} + C \otimes M)$ by a small, fixed number of steps of a stationary iteration with the block-diagonal preconditioner $(I_{n_t} \otimes \hat{L})$.

Another possibility is to employ a matrix equation approach to approximately solve for the Schur-complement system with $\hat{\mathcal{S}}$, where we use that $(I_{nt} \otimes \hat{L} + C \otimes M)$ is the Kronecker representation of the generalized Sylvester operator

$$\mathcal{S}(X) = \hat{L}X + MXC^T.$$

As mentioned in the beginning, there exist several low-rank methods such as the ADI iteration (see, e.g., [14]) and projection-based methods (see, e.g., [74]) that allow us to approximately solve linear matrix equations of this type. For our purposes, we use the method IKPIK, which is an inexact version of the method KPIK developed in [74]. We employ this method with a small and fixed number of steps to approximately solve the two matrix equations $(I_{n_t} \otimes L + \hat{C} \otimes M)$ and its transpose found in

$$\hat{\mathcal{S}} = \tau^{-1}\left(I_{n_t} \otimes L + \hat{C} \otimes M\right)\mathcal{M}_1^{-1}\left(I_{n_t} \otimes L + \hat{C} \otimes M\right)^T.$$

Note that due to the nature of the problem we have rewritten $\hat{\mathcal{S}}$ using

$$\hat{C} = \begin{bmatrix} 1 + \frac{\tau}{\sqrt{\beta}} & & & \\ -1 & 1 + \frac{\tau}{\sqrt{\beta}} & & \\ & \ddots & \ddots & \\ & & -1 & 1 + \frac{\tau}{\sqrt{\beta}} \end{bmatrix}.$$

This was done as IKPIK needs to work with the inverse of $\hat{C}$ and in the old formulation C could not be inverted. As this method is designed for the classical Sylvester equation we transform the two systems during the preconditioning to become

$$(I_{n_t} \otimes M)^{-1} \left(I_{n_t} \otimes L + \hat{C} \otimes M \right) = \left(I_{n_t} \otimes M^{-1} L + \hat{C} \otimes I \right)$$

and similarly for the transpose equation. The inexactness in IKPIK [77] allows us to approximately solve the systems for the two matrices $M^{-1}L$ and $\hat{C}$. Note that in our case the matrix $\hat{C}$ is trivial to solve for and we employ algebraic multigrid combined with a few steps of a stationary iteration [19] for the solution with $M^{-1}L$. Note that this approach is not yet ideal as one should use methods design for generalized Sylvester equations. However, due to the limitation of the scope of this paper, here we refrain from these latter ideas and instead propose them as possible topics of future research.

These preconditioners are then embedded into Algorithm 1 via the preconditioning function outlined in Algorithm 3.

As was noted in [80], a time-periodic control problem where $y(0,.) = y(T,.)$ results in the matrix C having circulant structure and we can then make use of the Fourier transform to transform the Schur-complement system to a system with only block-diagonal matrices that are now of complex nature, which simplifies the Sylvester equation.

Similar matrix structures are obtained in [1] for the simultaneous discretization in space and time. Preconditioning results using tensor structures are found in [2, 24].

```
ALGORITHM 3: PRECONDITIONER APPLICATION: APREC.
    Input: $W_{11}, W_{12}, W_{21}, W_{22}, W_{31}, W_{32}$
    Output: $Z_{11}, Z_{12}, Z_{21}, Z_{22}, Z_{31}, Z_{32}$
    Solve $\tau M Z_{11} = W_{11}$
    Solve $D_1 Z_{12} = W12$
    Solve $\tau\beta M Z_{21} = W_{21}$
    Solve $D_2 M Z_{22} = W_{22}$
    Compute $Z_{31}$ and $Z_{32}$ as the low rank solution of $\hat{S}$ with right-hand side defined
    by $W_{31}$ and $W_{32}$.
```

We have now obtained an overall low-rank algorithm for the computation of low-rank in time solutions to the PDE-constrained optimization problems. We want to comment on some of the algorithms properties before moving on to the existence of low-rank solutions in the next section. The computational cost of the algorithm is again dominated by the matrix vector product and the application of the preconditioner. Both of these are needed in any other iterative solver and the dimensionality of the low-rank factors determines how expensive the matrix vector products are. The cost of the preconditioner is here dominated by how efficiently we can solve the Schur-complement system. The stationary iteration suffers from the fact that the matrix structure does not result in parameter independent convergence. The solver based on standard Sylvester equations IKPIK typically shows much more parameter independent convergence behavior. We refer to section 6 for the numerical results. Nevertheless, more research is needed to employ solvers for generalized Sylvester equations. Additionally, the algorithm has to compute the truncation of the resulting matrices. This step is not free and one should use the full-space method for a small set of time-steps.

Nevertheless, the computation via a truncated SVD proved to be typically quite fast as we only needed to multiply with the large and skinny low-rank factor matrices.

4. Existence result of low-rank solutions. The previously derived low-rank method of course is competitive only if the solution to the optimal control problem exhibits a fast singular value decay, allowing us to replace it by a low-rank approximation. It thus remains to show that this is indeed a reasonable assumption for problems of the form (2.6). For this reason, in this section we establish a direct connection between (2.6) and the more prominent Sylvester equation

$$AX + XB = \tilde{C}, \tag{4.1}$$

where $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{m\times m}$, and $\tilde{C} \in \mathbb{R}^{n\times m}$. For the case that $\tilde{C}$ is of low-rank, i.e., $\tilde{C} = W_{\tilde{C}} V_{\tilde{C}^T}$, $W_{\tilde{C}} \in \mathbb{R}^{n\times k}$, $V_{\tilde{C}} \in \mathbb{R}^{m\times k}$, and $k \ll n, m$, it is well-known (see, e.g., [36, 35, 55]) that there exist approximations $X_r = W_X V_X^T \approx X$ with $W_X \in \mathbb{R}^{n\times r}, V_X \in \mathbb{R}^{m\times r}$, and $r \ll n, m$. Moreover, recently there has been an increased interest in numerical methods that, rather than computing the true solution and computing an approximation afterward, solely work on low-rank factors and iteratively construct approximations X_r converging to the true solution X, making these approaches feasible for dimensions $n, m \sim 10^6$. Popular methods are projection-based methods (see [9, 30, 75]), ADI-based methods (see [8, 14, 12, 78]), and multigrid methods (see [37]).

Let us now consider the second block-row of (2.6), for which we obtain that

$$(D_2 \otimes \beta\tau M_2)\, u + \left(D_3 \otimes \tau N^T\right) p = 0.$$

Solving this equation for u and inserting the result into the third block-row of (2.6) gives

$$-\left(I_{n_t} \otimes L + C \otimes M\right) y - \frac{1}{\beta}\left(D_3 \otimes \tau N\right)\left(D_2^{-1} \otimes M_2^{-1}\right)\left(D_3 \otimes N^T\right) p = d,$$

which, due to the properties of the Kronecker product and the definition of D_3, can be simplified to

$$-\left(I_{n_t} \otimes L + C \otimes M\right) y - \frac{\tau}{\beta}\left(D_2^{-1} \otimes N M_2^{-1} N^T\right) p = d.$$

Together with the first blow-rock, we thus can reformulate (2.6) in matrix notation as

$$\begin{aligned} \tau M_1 Y D_1 - LP - MPC &= \tau M_1 Y_{obs} D_1, \\ -LY - MYC^T - \frac{\tau}{\beta} N M_2^{-1} N^T P D_2^{-1} &= D. \end{aligned}$$

So far, we have only eliminated the second block-row and rewritten the problem in its matrix form. For the connection to (4.1), we have to make some additional assumptions on our initial setup (2.1). Typically, in real-life applications we can only observe a small portion $\tilde{y}$ of the state rather than the full y. In other words, the mass matrix M_1 in this case can be replaced by a low-rank matrix $\tilde{C}_{obs}\tilde{C}_{obs}^T = M_1$, with $\tilde{C}_{obs} \in \mathbb{R}^{n_1\times \ell}$ determining the observable parts of y. Note that in the context of classical control theory, $\tilde{C}_{obs}$ simply denotes the measurable output quantity of interest within the state-space representation of a linear dynamical system; see [3, 44, 52].

Similarly, in the case of boundary control, the rectangular matrix $N \in \mathbb{R}^{n_1 \times m}$ usually contains significantly fewer columns than rows. In summary, this means that we are often interested in the solution $\begin{bmatrix} Y & P \end{bmatrix}$ of the linear matrix equation

(4.2)
$$L \begin{bmatrix} Y & P \end{bmatrix} \begin{bmatrix} 0 & -I \\ -I & 0 \end{bmatrix} + M \begin{bmatrix} Y & P \end{bmatrix} \begin{bmatrix} 0 & -C^T \\ -C & 0 \end{bmatrix} + \tilde{C}_{obs} \tilde{C}_{obs}^T \begin{bmatrix} Y & P \end{bmatrix} \begin{bmatrix} \tau D_1 & 0 \\ 0 & 0 \end{bmatrix}$$
$$+ N M_2^{-1} N^T \begin{bmatrix} Y & P \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & -\frac{\tau}{\beta} D_2^{-1} \end{bmatrix} = \begin{bmatrix} \tau M_1 Y_{obs} D_1 \\ D \end{bmatrix}.$$

Pre- and postmultiplying the previous equation by M^{-1} and $\left[\begin{smallmatrix} 0 & -I \\ -I & 0 \end{smallmatrix}\right]$ leads to a generalized Sylvester equation of the form

$$\mathcal{A}\mathcal{X} + \mathcal{X}\mathcal{B} + \mathcal{Q}_1 \mathcal{X} \mathcal{R}_1 + \mathcal{Q}_2 \mathcal{X} \mathcal{R}_2 = \mathcal{E}_1 \mathcal{F}_2^T,$$

where

$$\mathcal{A} = M^{-1} L, \ \mathcal{B} = \begin{bmatrix} C^T & 0 \\ 0 & C \end{bmatrix}, \ \mathcal{Q}_1 = M^{-1} \tilde{C}_{obs} \tilde{C}_{obs}^T,$$

$$\mathcal{R}_1 = \begin{bmatrix} 0 & -\tau D_1 \\ 0 & 0 \end{bmatrix}, \ \mathcal{Q}_2 = M^{-1} N M_2^{-1} N^T, \ \mathcal{R}_2 = \begin{bmatrix} 0 & 0 \\ -\frac{\tau}{\beta} D_2^{-1} & 0 \end{bmatrix},$$

$$\mathcal{E}_1 = \begin{bmatrix} M_1 W_{Y_{obs}} & W_D \end{bmatrix}, \ \mathcal{F}_1 = \begin{bmatrix} D_1 V_{Y_{obs}} & 0 \\ 0 & V_D \end{bmatrix}.$$

Note that we assumed $Y_{obs} = W_{Y_{obs}} V_{Y_{obs}}^T$ and $Y_D = W_D V_D^T$ to be the low-rank representations for the right-hand side.

In what follows, we proceed as in [10, 22] and use the Sherman–Morrison–Woodbury formula [32] to simplify the previous equation. Since $\mathcal{Q}_1 = \mathcal{U}_1 \mathcal{V}_1^T$ and $\mathcal{Q}_2 = \mathcal{U}_2 \mathcal{V}_2^T$, for the Kronecker structured linear system, we subsequently obtain

$$\left(\mathcal{I} \otimes \mathcal{A} + \mathcal{B}^T \otimes \mathcal{I} + \mathcal{R}_1^T \otimes \mathcal{Q}_1 + \mathcal{R}_2^T \otimes \mathcal{Q}_2\right) \operatorname{vec}(\mathcal{X}) = \operatorname{vec}\left(\mathcal{E}_1 \mathcal{F}_1^T\right),$$

which can be rewritten as

$$\left(\underbrace{\mathcal{I} \otimes \mathcal{A} + \mathcal{B}^T \otimes \mathcal{I}}_{\tilde{A}} + \underbrace{\begin{bmatrix} \mathcal{I} \otimes \mathcal{U}_1 & \mathcal{I} \otimes \mathcal{U}_2 \end{bmatrix}}_{\tilde{U}} \underbrace{\begin{bmatrix} \mathcal{R}_1^T \otimes \mathcal{V}_1^T \\ \mathcal{R}_2^T \otimes \mathcal{V}_2^T \end{bmatrix}}_{\tilde{V}^T} \right) \operatorname{vec}(\mathcal{X}) = \operatorname{vec}\left(\mathcal{E}_1 \mathcal{F}_1^T\right).$$

According to the Sherman–Morrison–Woodbury formula, we alternatively get

$$\tilde{A} \operatorname{vec}(\mathcal{X}) = \operatorname{vec}\left(\mathcal{E}_1 \mathcal{F}_1^T\right) - \tilde{U} \underbrace{(I + \tilde{V}^T \tilde{A}^{-1} \tilde{U})^{-1} \tilde{V}^T \tilde{A}^{-1} \operatorname{vec}\left(\mathcal{E}_1 \mathcal{F}_1^T\right)}_{\operatorname{vec}(\mathcal{Y})}.$$

Since we have

$$\begin{aligned} \tilde{U} \operatorname{vec}(\mathcal{Y}) &= \tilde{U} \operatorname{vec}\left(\begin{bmatrix} \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{bmatrix}\right) \\ &= \begin{bmatrix} \mathcal{I} \otimes \mathcal{U}_1 & \mathcal{I} \otimes \mathcal{U}_2 \end{bmatrix} \operatorname{vec}\left(\begin{bmatrix} \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{bmatrix}\right) \\ &= \operatorname{vec}(\mathcal{U}_1 \mathcal{Y}_1) + \operatorname{vec}(\mathcal{U}_2 \mathcal{Y}_1) \end{aligned}$$

we can conclude that

$$\tilde{U}\,\mathrm{vec}\,(\mathcal{Y}) =: \mathrm{vec}\left(\mathcal{E}_2\mathcal{F}_2^T\right)$$

with $\mathcal{E}_2 \in \mathbb{R}^{n_1\times(l+m)},\ \mathcal{F}_2 \in \mathbb{R}^{2n_t\times(l+m)}$. In particular, this implies

$$\tilde{A}\,\mathrm{vec}\,(\mathcal{X}) = \mathrm{vec}\left(\mathcal{E}_1\mathcal{F}_1^T\right) - \mathrm{vec}\left(\mathcal{E}_2\mathcal{F}_2^T\right)$$

or, in other words, $\mathcal{X}$ can also be derived as the solution to a regular Sylvester equation of the form

$$\mathcal{A}\mathcal{X} + \mathcal{X}\mathcal{B} = \begin{bmatrix}\mathcal{E}_1 & -\mathcal{E}_2\end{bmatrix}\begin{bmatrix}\mathcal{F}_1^T\\ \mathcal{F}_2^T\end{bmatrix}.$$

We have now established that the PDE-constrained optimization problem can be written in form of a classical Sylvester equation for which we can use the existence results for a low-rank solution introduced in [36]. Note that we do not claim to actually proceed this way in order to compute the solution matrix $\mathcal{X}$. Obviously, determining the intermediate solution $\mathrm{vec}\,(\mathcal{Y})$ would be a challenge on its own. The previous steps rather should be understood as a theoretical evidence for the assumption that $\mathcal{X}$ indeed exhibits a very strong singular value decay. Keep in mind that we had to assume that the desired state Y_{obs} as well as D are of low-rank and that $l, m \ll n_1$, which is a reasonable assumption for realistic control problems.

A special case. One might argue that for the distributed control case, i.e., N begin square together with an (almost) entirely observable state, i.e., $\tilde{C}_{obs}\tilde{C}_{obs}^T = M_1$, the previous low-rank assumptions no longer hold true. Consequently, applying the Sherman–Morrison–Woodbury formula will not simplify (4.2) and we thus will have to deal with a linear matrix equation of the form

$$\sum_{i=1}^{4}\mathcal{A}_i\mathcal{X}\mathcal{B}_i = \mathcal{E}_1\mathcal{F}_1^T, \tag{4.3}$$

where we cannot benefit from additional structure in $\mathcal{A}_i$ and $\mathcal{B}_i$. Still, as has already been (numerically) observed and partially discussed in [10, 11, 22] for the special Lyapunov type case, i.e., $\mathcal{B}_i = \mathcal{A}_i^T$, the solution matrix $\mathcal{X}$ still seems to exhibit similar low-rank properties.

Although the most general case certainly is an interesting topic of future research, we want to conclude by pointing out that for the special case $M_2 = M_1 = N = M$ we immediately get an analogous (in fact even stronger) low-rank existence result for (4.2). This is due to the fact that here (4.2) is equivalent to the Sylvester equation

$$L\begin{bmatrix}Y & P\end{bmatrix}\begin{bmatrix}0 & -I\\ -I & 0\end{bmatrix} + M\begin{bmatrix}Y & P\end{bmatrix}\begin{bmatrix}\tau D_1 & -C^T\\ -C & -\frac{\tau}{\beta}D_2^{-1}\end{bmatrix} = \begin{bmatrix}\tau M Y_{obs} D_1\\ D\end{bmatrix}$$

for which we again can apply the low-rank existence results from [36].

5. Other state equations.

Stokes equation. In addition to the heat equation as a test problem we here also consider the Stokes equation. The discretization of the Stokes control problem can be performed similarly to the case for the heat equation and we refer the interested reader to [27]. The Stokes equations are given by

$$y_t - \nu\triangle y + \nabla p = u \text{ in } [0,T]\times\Omega, \tag{5.1}$$

$$-\nabla\cdot y = 0 \text{ in } [0,T]\times\Omega, \tag{5.2}$$

$$y(t,.) = g(t) \text{ on } \partial\Omega, t\in[0,T], \tag{5.3}$$

$$y(0,.) = y^0 \text{ in } \Omega, \tag{5.4}$$

and the functional we are interested in is defined as

$$J(y,u) = \frac{1}{2}\int_0^T\int_{\Omega_1}(y-\bar{y})^2\,dxdt + \frac{\beta}{2}\int_0^T\int_{\Omega_2}u^2dxdt. \tag{5.5}$$

Our goal is to build a discrete Lagrangian using an all-at-once approach [82], where we set up a discrete problem within the space-time cylinder. Using the trapezoidal rule and $Q2/Q1$ finite elements in space leads to the following discrete objective function:

$$J(y,u) = \frac{\tau}{2}\left(y-y_{obs}\right)^T\mathcal{M}_1\left(y-y_{obs}\right) + \frac{\tau\beta}{2}u^T\mathcal{M}_2u \tag{5.6}$$

with

$$\mathcal{M}_1 = \text{blkdiag}\left(\frac{1}{2}M_1, 0, M_1, 0, \ldots, M_1, 0, \frac{1}{2}M_1, 0\right),$$
$$\mathcal{M}_2 = \text{blkdiag}\left(\frac{1}{2}M_2, M_2, \ldots, M_2, \frac{1}{2}M_2\right),$$

where we reuse the notation for the heat equation. Note that for the Stokes case the vectors y_i are split into a velocity v part with $d = 2,3$ components and pressure part p, i.e.,

$$y_i = \begin{bmatrix} y_i^v \\ y_i^p \end{bmatrix}.$$

The all-at-once discretization of the state equation using finite elements in space and an implicit Euler scheme in time is given by

$$\mathcal{K}y - \tau\mathcal{N}u = d, \tag{5.7}$$

where

$$\mathcal{K} = \begin{bmatrix} \mathcal{L} & & & \\ -\mathcal{M} & \mathcal{L} & & \\ & \ddots & \ddots & \\ & & -\mathcal{M} & \mathcal{L} \end{bmatrix}, \quad \mathcal{N} = \begin{bmatrix} N & 0 & & & & \\ 0 & 0 & 0 & & & \\ 0 & N & 0 & & & \\ & 0 & & & & \\ & & & \ddots & & \\ & & & & 0 & N \\ & & & & 0 & 0 \end{bmatrix}, \quad d = \begin{bmatrix} Ly_0 + f \\ 0 \\ f \\ \vdots \\ f \\ 0 \end{bmatrix}.$$

Here,

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

represents an instance of a time-dependent Stokes problem with B the discrete divergence, M is the mass matrix for the domain Ω, the matrix L is defined as $L = \tau^{-1}M + K$, the matrix N is a rectangular matrix that can be written as

$$D_3 \otimes \mathcal{N}_s \text{ with } \mathcal{N}_s = \begin{bmatrix} N \\ 0 \end{bmatrix}$$

which represents the distributed control term control term where $N = M$, the matrix

$$\mathcal{M} = \begin{bmatrix} \tau^{-1}M & 0 \\ 0 & 0 \end{bmatrix}$$

is associated with the discretization in time via the implicit Euler scheme, and the right-hand side d consists of a contribution from the initial condition y_0 and a vector f representing forcing terms and contributions of boundary conditions. Note that all matrices here correspond to the ones introduced for the heat equation but equipped with a block form corresponding to the components for the velocity y_v and pressure y_p. The first order conditions using a Lagrangian with Lagrange multiplier p lead to the system

$$\underbrace{\begin{bmatrix} \tau\mathcal{M}_1 & 0 & -\mathcal{K}^T \\ 0 & \beta\tau\mathcal{M}_2 & \mathcal{N}^T \\ -\mathcal{K} & \mathcal{N} & 0 \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} y \\ u \\ p \end{bmatrix} = \begin{bmatrix} \tau\mathcal{M}_1 y_{obs} \\ 0 \\ d \end{bmatrix}, \tag{5.8}$$

where again we can switch to a Kronecker structure defined by

$$\begin{bmatrix} D_1 \otimes \tau\mathcal{M} & 0 & -\left(I_{n_t} \otimes \mathcal{L} + C^T \otimes \mathcal{M}\right) \\ 0 & D_2 \otimes \beta\tau M_2 & D_3 \otimes \mathcal{N}_s^T \\ -\left(I_{n_t} \otimes \mathcal{L} + C \otimes \mathcal{M}\right) & D_3 \otimes \mathcal{N}_s & 0 \end{bmatrix}. \tag{5.9}$$

We can now in a similar way as earlier use the low-rank MINRES method. Again, here we apply a block-diagonal preconditioner of the form

$$P = \begin{bmatrix} D_1 \otimes \tau\hat{M}_1 & 0 & 0 \\ 0 & D_2 \otimes \beta\tau\hat{M}_2 & 0 \\ 0 & 0 & \hat{S} \end{bmatrix}. \tag{5.10}$$

Here $\hat{M} = \text{blkdiag}\,(\hat{M}_1, \gamma I)$ with $\gamma = \beta\tau h^d$ (see [82] for details). Here d is the dimension of the problem ($d = 2, 3$) and h the mesh parameter. The matrices M_1 and M_2 are approximated via a Chebyshev semi-iteration [33, 34, 86], or in the case of lumped mass matrices we trivially have $\hat{M}_{1,2} = M_{1,2}$. The approximation of the Schur-complement is much more tricky in this case as for the indefinite $\mathcal{M}_1$ the Schur-complement is not well-defined. Thus, we again use the approximation $\hat{M} = \text{blkdiag}\,(\hat{M}_1, \gamma I)$ to form an approximate Schur-complement

$$S = \tau^{-1}\mathcal{K}\hat{\mathcal{M}}_1^{-1}\mathcal{K}^T + \tau^{-1}\beta^{-1}\mathcal{N}\mathcal{M}_2^{-1}\mathcal{N}^T$$

with $\hat{\mathcal{M}}_1$ a block-diagonal involving $\hat{M}$. We in turn approximate this via

$$\hat{S} = \tau^{-1}\left(\mathcal{K} + \hat{\mathcal{M}}\right)\mathcal{M}_1^{-1}\left(\mathcal{K} + \hat{\mathcal{M}}\right)^T,$$

where $\hat{\mathcal{M}} = \text{blkdiag}(\frac{1}{\sqrt{\beta}}M_1, 0, \ldots, \frac{1}{\sqrt{\beta}}M_1, 0)$ for the distributed control case. As in section 3 we note that the matrix

$$\left(\mathcal{K} + \hat{\mathcal{M}}\right) = \left(I_{n_t} \otimes \tilde{\mathcal{L}} + C \otimes \mathcal{M}\right)$$

with $\tilde{\mathcal{L}} = \left[\begin{smallmatrix} (\tau^{-1}+\beta^{-1/2})M_1+K & B^T \\ B & 0 \end{smallmatrix}\right]$. We now proceed in the following way. A stationary iteration scheme with a fixed number of steps is used to approximately solve $(I_{n_t} \otimes \tilde{\mathcal{L}} + C \otimes M)$ with preconditioner $(I_{n_t} \otimes \tilde{\mathcal{L}})$ and within this preconditioner systems with $\tilde{\mathcal{L}}$ are approximately solved using another Uzawa scheme with a fixed but small number of iterations. For this inner Uzawa iteration the inverse of the preconditioner is given by

$$\begin{bmatrix} \left[\left(\tau^{-1} + \beta^{-1/2}\right) M_1 + K\right]_{MG}^{-1} & 0 \\ 0 & \left(\tau^{-1} + \beta^{-1/2}\right) [K_p]_{MG}^{-1} + [M_p]_{MG}^{-1} \end{bmatrix},$$

where the $[\ldots]_{MG}^{-1}$ indicates the use of a geometric [41, 87] or algebraic multigrid method [69, 29]. Preconditioners of this type are of so-called Cahouet–Chabard form [21] and the derivation can be done using a least squares commutator approach [27, 82].

Again, for more robustness, more sophisticated Sylvester solvers should be used in the future to guarantee robustness with respect to the system parameters.

Convection-diffusion equation. Before coming to the numerical results we quickly want to introduce the last state equation considered here. The PDE constraint is now given by the convection-diffusion equation

$$y_t - \varepsilon \triangle y + w \cdot \nabla y = u \text{ in } \Omega, \tag{5.11}$$

$$y(:, x) = f \text{ on } \partial\Omega, \tag{5.12}$$

$$y(0, :) = y_0 \tag{5.13}$$

as the constraint to the following objective function:

$$J(y, u) = \frac{1}{2}\int_0^T \int_{\Omega_1} (y - \bar{y})^2 \, dxdt + \frac{\beta}{2}\int_0^T \int_{\Omega_2} u^2 dxdt. \tag{5.14}$$

Note that the parameter ε is crucial to the convection-diffusion equation as a decrease in its value is adding more hyperbolicity to the PDE where the wind w is predefined. Such optimization problems have recently been discussed in [67, 43, 62] and for brevity we do not discuss the possible pitfalls regarding the discretization. Here we focus on a discretize-then-optimize scheme using a streamline upwind Galerkin (SUPG) approach introduced in [20]. Note that other schemes, such as discontinuous Galerkin methods [83] or local projection stabilization [62], are typically more suitable discretizations for the optimal control setup as they often provide the commutation between optimize first or discretize first for the first order conditions. Nevertheless, our approach will also work for these discretizations. Once again we employ a trapezoidal rule in connection with finite elements and now additionally use a SUPG stabilization. The discretized objective function and state equation are given by

$$J(y, u) = \frac{\tau}{2}\left(y - y_{obs}\right)^T \mathcal{M}_1 \left(y - y_{obs}\right) + \frac{\tau\beta}{2} u^T \mathcal{M}_2 u,$$

which is the same as for the heat equation case. For the all-at-once discretization of the convection-diffusion equation we get the same structure as before,

$$\mathcal{K}y - \tau\mathcal{N}u = d, \tag{5.15}$$

with

$$\mathcal{K} = \begin{bmatrix} L_s & & & \\ -M_s & L_s & & \\ & \ddots & \ddots & \\ & & -M_s & L_s \end{bmatrix}, \quad \mathcal{N} = \begin{bmatrix} M_s & & & \\ & M_s & & \\ & & \ddots & \\ & & & M_s \end{bmatrix}, \quad d = \begin{bmatrix} M_1 y_0 + f \\ f \\ \vdots \\ f \end{bmatrix}.$$

Note that due to the SUPG test functions used we now have M_s, which is obtained entrywise from evaluating the integrals

$$(M_s)_{ij} = \int_\Omega \phi_i \phi_j + \delta \int_\Omega \phi_i \, (w \cdot \nabla \phi_j),$$

where ϕ are the finite element test functions and δ is a parameter coming from the use of SUPG [20, 27]. We then have $L_s = M_s + \tau K_s$, where K_s is the standard nonsymmetric matrix representing the SUPG discretization of the convection-diffusion equation. We can now see that this again is of the desired Kronecker form

$$\begin{bmatrix} D_1 \otimes \tau M_1 & 0 & -\left(I_{n_t} \otimes L_s + C \otimes M_s\right)^T \\ 0 & D_2 \otimes \beta \tau M_2 & D_3 \otimes \tau M_s^T \\ -\left(I_{n_t} \otimes L_s + C \otimes M_s\right) & D_3 \otimes \tau M_s & 0 \end{bmatrix} \begin{bmatrix} y \\ u \\ p \end{bmatrix} = \begin{bmatrix} D_1 \otimes \tau M_1 y_{obs} \\ 0 \\ d \end{bmatrix}. \tag{5.16}$$

Again, we employ the low-rank version of MINRES to solve this system. Note that for nonsymmetric formulations such as the one obtained from an optimize-then-discretize strategy we can also use low-rank versions of nonsymmetric Krylov solvers such as GMRES [72] or BICG [31]. A preconditioner is of the form

$$P = \begin{bmatrix} D_1 \otimes \tau \hat{M}_1 & 0 & 0 \\ 0 & D_2 \otimes \beta \tau \hat{M}_2 & 0 \\ 0 & 0 & \hat{S} \end{bmatrix}, \tag{5.17}$$

where the two blocks involving mass matrices are as before and the Schur-complement of $\mathcal{A}$

$$\begin{aligned} S = {} & \left(I_{n_t} \otimes L_s + C \otimes M_s\right) \left(D_1^{-1} \otimes \tau^{-1} M_1^{-1}\right) \left(I_{n_t} \otimes L_s + C \otimes M_s\right)^T \\ & + \left(D_3 \otimes \tau M_s\right) \left(D_2^{-1} \otimes \beta^{-1} \tau^{-1} M_2^{-1}\right) \left(D_3 \otimes \tau M_s^T\right). \end{aligned} \tag{5.18}$$

As before with the heat and Stokes problem the aim for an efficient approximation of S is by not dropping terms but rather to create an approximation that matches both terms in S. In a similar way to the technique introduced in [63] for the steady case can now be extended by introducing a term $\hat{D} \otimes \hat{M}$ so that

$$\begin{aligned} & \left(\hat{D} \otimes \hat{M}\right) \left(D_1^{-1} \otimes \tau^{-1} M_1^{-1}\right) \left(\hat{D} \otimes \hat{M}\right)^T \\ & \quad \approx \left(D_3 \otimes \tau M_s\right) \left(D_2^{-1} \otimes \beta^{-1} \tau^{-1} M_2^{-1}\right) \left(D_3 \otimes \tau M_s^T\right). \end{aligned}$$

If we now assume $D_1 = D_2 = D_3 = I_{n_t}$ one can obtain

$$\hat{S} = \left(I_{n_t} \otimes \hat{L}_s + C \otimes M_s\right) \left(D_1^{-1} \otimes \tau^{-1} M_1^{-1}\right) \left(I_{n_t} \otimes \hat{L}_s + C \otimes M_s\right)^T, \tag{5.19}$$

where

$$\hat{L}_s = \left(1 + \frac{1}{\sqrt{\beta}}\right) M_s + \tau K_s.$$

Note that it is of course possible to construct such an approximation without assuming $D_1 = D_2 = D_3 = I_{n_t}$. We can now proceed with a stationary iteration for the two Sylvester equations in $\hat{S}$, or if we want to employ IKPIK again we rewrite the approximation as

$$\hat{S} = \left(I_{n_t} \otimes L_s + \hat{C} \otimes M_s\right) (D_1^{-1} \otimes \tau^{-1} M_1^{-1}) \left(I_{n_t} \otimes L_s + \hat{C} \otimes M_s\right)^T \tag{5.20}$$

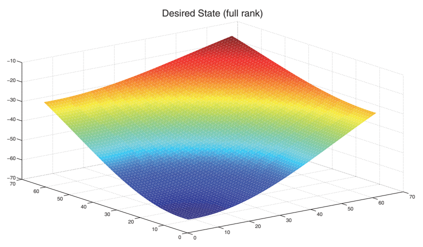
with

$$\hat{C} = \begin{bmatrix} 1+\frac{\tau}{\sqrt{\beta}} & & & \\ -1 & 1+\frac{\tau}{\sqrt{\beta}} & & \\ & \ddots & \ddots & \\ & & -1 & 1+\frac{\tau}{\sqrt{\beta}} \end{bmatrix}.$$

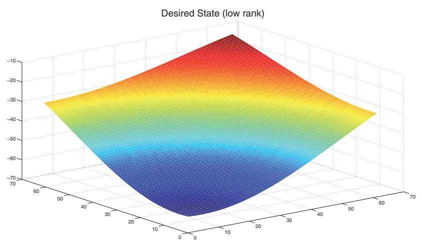
Now when solving with $\hat{S}$ we can approximately solve both $(I_{n_t} \otimes L_s + \hat{C} \otimes M_s)$ and its transpose using IKPIK, where we use an algebraic multigrid for the approximate solution with the stiffness matrix L_s. These preconditioners need to be adapted when other discretizations are used, but we expect our results to carry over to these cases as well.

6. Numerical results.

We are now going to present results for the low-rank in time solution of certain PDE-constrained optimization problems. The results presented in this section are based on an implementation of the above described algorithms within MATLAB, whereas we perform the discretization of the PDE-operators within the deal.II [7] framework using $Q1$ finite elements for the heat equation and the convection-diffusion equation. The Stokes equation is discretized with $Q2$ elements for the velocity and $Q1$ elements for the pressure component. For the algebraic multigrid approximation, we used HSL MI_20 [19]. For some preconditioner we used backslash within MATLAB for the innermost solution within the preconditioner. Our implementation of MINRES is based on a vector version presented in [27] and was stopped with a tolerance of 10^{-4} or 10^{-6} for the relative pseudoresidual. Please note that we ideally should couple the stopping criterion for the iterative solver to the discretization error. We also give results for the stationary iteration and IKPIK used for the approximation within the preconditioner. Our experiments are performed for a final time $T = 1$ with a varying number of time-steps. As the domain Ω we consider the unit cube, but other domains are of course possible. We specify the boundary conditions for each problem separately. Throughout the results section we fixed the truncation at 10^{-8}, for which we observed good results. Additionally, we also performed not-listed experiments with a tolerance of 10^{-10} for which we also observed fast convergence. Larger tolerances should be combined with a deeper analysis of the algorithms and a combination with flexible outer solvers. All results are performed on a standard Ubuntu desktop Linux machine with Intel Xeon CPU W3503 @ 2.40 GHz and 6 GB of RAM.



(a) Desired state $\bar{\mathbf{y}}_{10}$—full rank



(b) Desired state $\bar{\mathbf{y}}_{10}$—low rank

Fig. 1. *Desired state in full-rank and low-rank form.*

Table 1

Results for full-rank MINRES versus low-rank (LR) MINRES with stationary iteration preconditioner for 20 *or* 100 *time-steps and a variety of different meshes. Both iteration numbers and computing times in seconds are listed. DoF = degrees of freedom. OoM indicates out of memory in MATLAB. Results are shown for* $\beta = 10^{-4}$.

DoF	FR (20) # it(t)	LR (20) # it(t)	FR (100) # it(t)	LR (100) # it(t)
289	17(0.2)	15(5.6)	31(1.1)	19(7.6)
1089	19(0.7)	17(8.9)	33(5.56)	21(11.9)
4225	19(3.5)	17(19.7)	35(26.6)	23(26.3)
16641	21(17.5)	19(72.1)	35(125.8)	23(97.6)
66049	23(81.8)	19(324.8)	OoM	25(427.4)

6.1. The heat equation.

Distributed control. As the first example shown in this section we use the heat equation with a distributed control term. We choose the boundary conditions for this problem to be of zero Dirichlet type. We first show how well the desired state

$$y_{obs} = -64 \exp\left(-\left((x_0 - 0.5t)^2 + (x_1 - 0.5t)^2\right)\right)$$

is approximated in low-rank form. Figure 1 illustrates this for grid point 10 in time where the right-hand side $\text{vec}^{-1}(\tau \mathcal{M}_1 y_{obs}) = B_{11} B_{12}^T$ is approximated by low-rank factors of rank 2.

Table 1 shows first results for the comparison of the full-rank MINRES versus the low-rank version. We want to point out that here we use the backslash operator in MATLAB to evaluate the matrix L within the preconditioner, but this can easily be replaced by a multigrid approximation and in fact is done later. For the full-rank version, we only used a block-diagonal approximation for the matrix $\mathcal{K}$ and hence the robustness with respect to changes in the number of time-steps is not given. This would typically be the case and our results using deal.II and C++ in [81, 61] show robustness with respect to the number of time-steps. Nevertheless, every increase in the number of time-steps also results in an increase in the matrix size and so one would expect when the number of time-steps is increased fivefold that the same happens for the time needed to solve the linear system. Going back to the results in Table 1, where both the timings and iteration numbers are shown for a variety of mesh-sizes and two different orders of grid points in time, both methods perform mesh-independent and we can see that the low-rank method shows almost no increase when the number of

Table 2

Results for full-rank MINRES versus low-rank MINRES with both the stationary iteration (SI) and IKPIK *for a fixed mesh with* $16,641$ *unknowns in space. We show varying time-steps and additionally the rank of the state/control/adjoint state. Both iteration numbers and computing times in seconds are listed. Results are shown for* $\beta = 10^{-4}$.

DoF 16641	20 # it(t)	100 # it(t)	200 # it(t)	400 # it(t)	600 # it(t)
LR(SI)	19(108.2)	21(307.8)	25(432.7)	43(671.9)	61(937.3)
LR(IKPIK)	19(115.1)	19(288.9)	19(296.8)	21(335.3)	21(357.1)
Rank (SI)	8/10/10	10/11/11	12/13/13	11/14/14	14/15/15
FR	21(18.3)	35(124.0)	63(434.3)	OoM	OoM

time-steps is drastically increased. Note also the degrees of freedom given here are only for the spatial discretization. The overall dimension of the linear system is then given by $3nn_t$, where n represents the spatial degrees of freedom. We see that the iteration times for the full rank solver go up, and using the nonoptimal preconditioners we additionally see that the times increase more than just by a factor of five. We also see that due to the cost of performing a low-rank truncation the full-rank method always outperforms the low-rank scheme for a small number of time-steps. Nevertheless, the low-rank method can easily solve problems that are no longer tractable for full-rank methods.

Next we compare how both the full-rank and the low-rank method perform when the number of time-steps is further increased. We therefore consider a fixed mesh for a varying time-discretization. Table 2 shows the results for both the full-rank and the low-rank method. We additionally show the rank of the three components of the state, control, and adjoint state. We started computing the truncation process using a maximum size of the truncated SVD of 20, which was sufficient for all discretizations in time using a truncation tolerance of 10^{-8}. In order to keep the iteration numbers from growing too much with an increase in the number of time-steps we increased the number of stationary iterations for the preconditioner from two to three for the last two columns in Table 2. Additionally, we show the results for the Schur-complement approximation when IKPIK is employed. For this we employ IKPIK with a fixed number of steps. We used four steps for the results shown in Table 2. For the evaluation of the inverse of the stiffness matrix within IKPIK we used a five steps of a stationary iteration with one cycle of an algebraic multigrid as a preconditioner. The algebraic multigrid is also used within the stationary iteration approximation to the Sylvester equation. We see again that the full-rank method exceeds the memory limit in MATLAB. It can also be seen that the increase in rank and computing time is typically moderate. Note that the system dimension considering a full-rank solution is ranging from $998,460$ to $29,953,800$ unknowns.

In order to illustrate the distribution of the singular values we show in Figure 2 how the relative value of the singular values behaves throughout the iteration. Shown are the scaled singular values (σ_j/σ_1) of the approximation to the state for the problem with 4225 unknowns and 100 grid points in time. In Table 3 we illustrate the performance of our scheme when different values for the regularization parameter are considered. So far the preconditioners introduced based on the stationary iteration have used a direct solver for the solution of the systems with

$$\hat{L} = \left(\left(1 + \frac{\tau}{\sqrt{\beta}} \right) M_1 + \tau K \right)$$

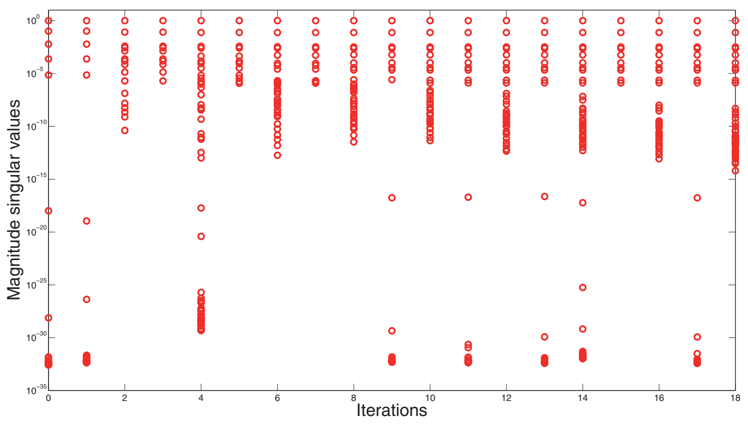


Fig. 2. *Singular values of the approximate solution during the iteration before truncation.*

Table 3

Results for low-rank MINRES with 100 *time-steps and a varying regularization parameter on three different meshes. Both iteration numbers and computing times in seconds are listed. The results shown use the* IKPIK *approximation of the Sylvester-type operators.*

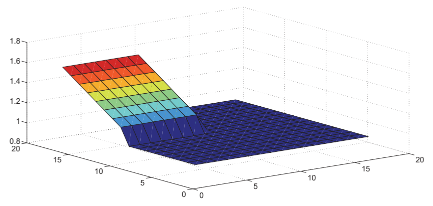
DoF β	1089 (100) # it(t)	4225 (100) # it(t)	16641 (100) # it(t)	16641 (100) $\|\mathbf{y}-\bar{\mathbf{y}}\| / \|\bar{\mathbf{y}}\|$
10^{-4}	17(25.8)	19(80.1)	19(311.9)	0.3979
10^{-6}	17(23.8)	17(64.7)	19(267.5)	0.2019
10^{-8}	15(20.3)	17(56.7)	19(227.2)	0.0809

both in the full-rank method and the low-rank one. We now illustrate that we can easily approximate this matrix using an algebraic multigrid technique by also showing that our preconditioner performs robustly with respect to the regularization parameter β. We here compute the truncated singular value decomposition up to order 20 and then cut off corresponding to the truncation tolerance. We additionally increased the number of stationary iteration steps for the matrix $(I_{n_t} \otimes \hat{L} + C \otimes M)$ with preconditioner $(I_{n_t} \otimes \hat{L})$ to 4.

Boundary control. In the following we demonstrate that our approach also works for the case of a boundary control problem. The desired state is shown in Figure 3(a) and the computed state needed to approximate this in Figure 3(b). In Table 4 we show results for the low-rank MINRES approximation for a variety of mesh-parameters and regularization parameters. Details on the preconditioners used can be found in [61]. As in the last example for the distributed control case we choose four Uzawa iterations and a tolerance of 10^{-4} for the iterative solver. Here we evaluate $\hat{L}$ again using the backslash operator in MATLAB but the use of AMG is straightforward. We do not employ IKPIK as its use for this setup needs to be further investigated.

6.2. Stokes equation.

The configuration for the Stokes equation is taken from [82] and originally appeared in [46]. The spatial domain is the unit cube $\Omega = [0,1]^d$ with a time domain $[0,1]$. The target flow is the solution for an unsteady Stokes


(a) Desired state at grid point 65 in time.

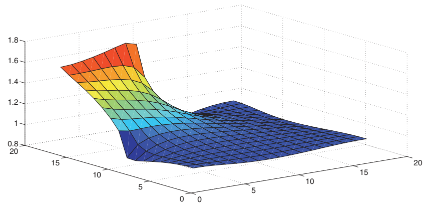

(b) Computed state at grid point 65 in time for $\beta = 10^{-6}$.

FIG. 3. *Desired state and computed state for a boundary control problem.*

TABLE 4
Results for low-rank MINRES with 100 *time-steps and a varying regularization parameter on three different meshes for a boundary control example. Both iteration numbers and computing times in seconds are listed.*

DoF β	289 (100) # it(t)	4225 (100) # it(t)	16641 (100) # it(t)
10^{-2}	49(137.3)	61(236.18)	79(802.7)
10^{-4}	67(179.8)	99(406.6)	151(1510.6)
10^{-6}	63(169.2)	95(380.4)	147(1448.6)

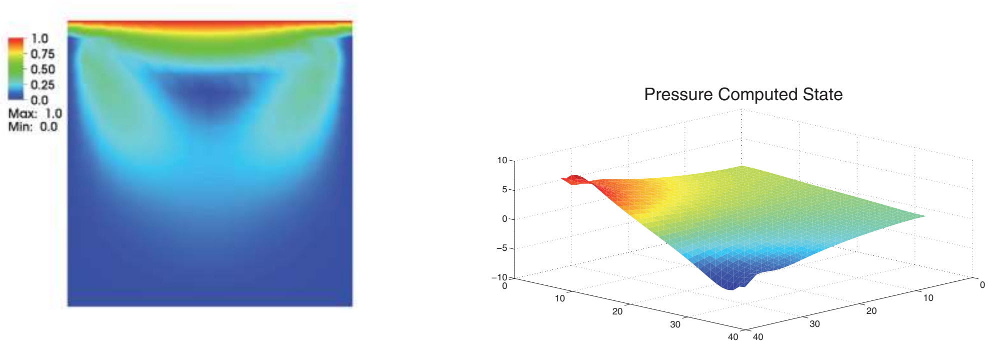
TABLE 5
Results for low-rank MINRES with 100 *time-steps and a varying regularization parameter on three different meshes for a Stokes control example. Both iteration numbers and computing times in seconds are listed.*

DoF β	578+81 (100) # it(t)	2178+289 (100) # it(t)	8450+1089 (100) # it(t)
10^{-1}	11(224.4)	12(624.8)	14(3601.9)
10^{-5}	15(290.2)	15(737.6)	17(4091.5)

equation with Dirichlet boundary conditions, i.e., $y = (1, 0)$ when the second spatial component $x_2 = 1$ and $y = (0, 0)$ on the remaining boundary for the two-dimensional case. For the control problem we now consider the following time-dependent boundary conditions. For the top-boundary where $x_2 = 1$ we get $y = (1 + \frac{1}{2}\cos(4\pi t - \pi), 0)$ and zero elsewhere in two space dimensions and we set viscosity to $1/100$. Figure 4(a) depicts the desired state and the corresponding computed pressure is shown in Figure 4(b). For the results shown in Table 5 we note that we needed to set the number of stationary iteration steps for the outer iteration to 30 and for the inner one for the small saddle point system to 5. We believe that the outer iteration can be replaced by a robust Sylvester solver.

Apart from the approximation of the Neumann–Laplacian on the pressure space whose inverse was evaluated using an algebraic multigrid scheme, we simply used the backslash operator to evaluate the remaining components. A further increase in computational efficiency can be gained when these are replaced by multigrid approximations.

6.3. Convection-diffusion equation. The configuration for the convection-diffusion equation is taken from [27] and is typically referred to as the double glazing problem. The spatial domain is the unit cube $\Omega = [-1, 1]^2$ with a time domain $[0, 1]$.



(a) Desired state at grid point 10 in time.

(b) Pressure of the computed state for $\beta = 1e-1$.

FIG. 4. *Desired state and computed pressure for the Stokes flow problem.*

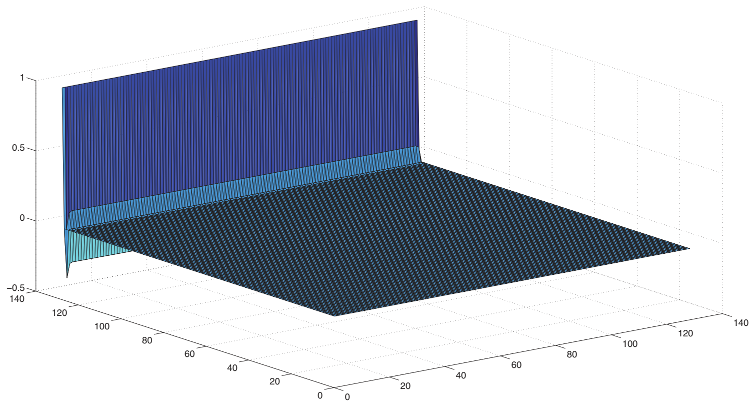


FIG. 5. *Computed state for $\beta = 10^{-6}$ at grid point 10 in time.*

The wind w is given by

$$w = \left(2y(1-x^2), -2x(1-y^2)\right).$$

Here we set the parameter to ε to $1/200$ and the boundary condition is a Dirichlet zero condition with the exception of $y = 1$ when $x_2 = 1$. The desired state is set to zero throughout the domain [62]. In Figure 5 we show the computed state for grid point 10 in time. Due to the nonsymmetric nature of the PDE operator we have not employed the recommended multigrid technique [66] and simply used the backslash operator here. The results shown in Table 6 indicate a robust performance of the low-rank MINRES method. We here set the number of stationary iterations to 15. We additionally show iteration numbers for the use of IKPIK with the use of a direct solver and a fixed number of IKPIK steps. The tolerance for MINRES is set to 10^{-6}.

Additionally, we show in Table 7 the numerical ranks that we obtain for the computed state. We hereby change the parameter ε in order to make the problem more hyperbolic. The tolerance for the low-rank truncation is chosen to be 10^{-8}.

Table 6

Results for low-rank MINRES with 100 *or* 200 *time-steps and a varying regularization parameter on a variety of meshes for a convection-diffusion control example. Both iteration numbers and computing times in seconds are listed. The tolerance for convergence is* 1e − 6.

	DoF	1089 (100)	4225 (100)	16641 (100)	4225 (200)	16641 (200)
	β	# it(t)	# it(t)	# it(t)	# it(t)	# it(t)
SI	10^{-8}	4(17.36)	6(66.1)	6(291.1)	6(68.8)	6(277.6)
IKPIK	10^{-8}	4(8.5)	6(35.6)	6(151.0)	6(36.7)	6(161.4)
SI	10^{-6}	10(40.4)	10(109.7)	14(639.5)	12(128.1)	14(646.3)
IKPIK	10^{-6}	10(22.3)	10(72.3)	14(520.3)	12(98.7)	14(568.6)

Table 7

Results for low-rank MINRES with 100 *time-steps and a fixed mesh with* 4225 *DoFs. We here show the ranks of the low-rank factors with respect to a varying* ε *within the convection-diffusion equation. The tolerance for convergence is* 1e − 6.

DoF	1/50	1/500	1/5000	1/50000
$\bar{\mathbf{y}}_1$ (trunctol $= 1e-8$)	9/7/7	7/6/6	7/5/5	7/5/5
$\bar{\mathbf{y}}_2$ (trunctol $= 1e-8$)	30/35/35	29/49/49	29/50/50	29/50/50
$\bar{\mathbf{y}}_2$ (trunctol $= 1e-5$)	4/3/3	3/4/4	3/4/4	3/4/4

We show ranks for both the zero desired state $\bar{\mathbf{y}}_1$ and a different desired state with a higher frequency $\bar{\mathbf{y}}_2$. We note that the control and adjoint states both need more terms for the low-rank representation when the desired state is nontrivial. The maximum number of vectors stored was limited to 50, which was not sufficient for small values of ε. This does not indicate that the method fails in this case but rather that it is crucial to investigate the relation between the discretization error, the algebraic error, and the truncation error. The truncation tolerance of 10^{-8} could have simply been too tight for the level of discretization. Hence, we additionally show the results for the truncation level 10^{-5}, which in this case is smaller than h^2.

7. Outlook. We believe that the research presented here opens some interesting angles that should be studied in the future. The incorporation of additional constraints such as control and state constraints is typically very important for real-world scenarios. We plan to investigate a technique introduced in [45] where the state and adjoint state are computed first and hence is amenable to low-rank techniques, and then the constrained control is computed. It is further desired to investigate more complicated discretizations in time. Of particular interest, we want to study backward differentiation formulas [5] as these can be easily incorporated simply modifying the C matrix in (3.1). We further plan to incorporate more sophisticated generalized Sylvester equation solvers for $(I_{n_t} \otimes L + C \otimes M)$, which we believe allows for more robustness with respect to the system parameters and should be combined with a flexible outer method [71]. It is further crucial to investigate how the low-rank techniques can be extended to incorporate nonlinearities of both the objective function and the PDE constraint such as [17].

8. Conclusions. In this paper we proposed the use of a low-rank methodology for the solution to PDE-constrained optimization problems. In particular we introduced a low-rank in time approach that allows us to significantly reduce the storage requirements in time for a one-shot solution of the optimal control problem. We were also able to rewrite the problem in such a way that we can obtain low-rank existence results from classical Sylvester equation theory. We additionally discussed a stationary iteration as a preconditioner for the Schur-complement approximation within the overall block-diagonal preconditioner. We further illustrated that this technique can be used for many well-known PDEs. Our numerical results illustrated that even with the rather crude Schur-complement approximation a rather robust performance could be obtained. The low-rank method presented enabled computations that are no longer possible to perform with the full-rank approach, which we see as a crucial feature of our methodology.

REFERENCES

[1] R. Andreev, *Space-Time Discretization of the Heat Equation. A Concise MATLAB Implementation*, preprint, arXiv:1212.6037, 2012.

[2] R. Andreev and C. Tobler, *Multilevel Preconditioning and Low Rank Tensor Iteration for Space-Time Simultaneous Discretizations of Parabolic PDES*, Technical report 2012–16, Seminar for Applied Mathematics, ETH Zürich, 2012.

[3] A. Antoulas, *Approximation of Large-Scale Dynamical Systems*, SIAM, Philadelphia, 2005.

[4] A. Antoulas, D. Sorensen, and Y. Zhou, *On the decay rate of Hankel singular values and related issues*, Systems Control Lett., 46 (2002), pp. 323–342.

[5] U. M. Ascher, R. M. Mattheij, and R. D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Classics in Appl. Math. 13, SIAM, Philadelphia, 1955.

[6] J. Baglama and L. Reichel, *Augmented implicitly restarted Lanczos bidiagonalization methods*, SIAM J. Sci. Comput, 27 (2005), pp. 19–42.

[7] W. Bangerth, R. Hartmann, and G. Kanschat, *deal.*II—*a general-purpose object-oriented finite element library*, ACM Trans. Math. Software, 33 (2007), pp. Art. 24, 27.

[8] U. Baur and P. Benner, *Factorized solution of Lyapunov equations based on hierarchical matrix arithmetic*, Computing, 78 (2006), pp. 211–234.

[9] P. Benner and T. Breiten, *On Optimality of Interpolation-Based Low-Rank Approximations of Large-Scale Matrix Equations*, MPI Magdeburg Preprint MPIMD/11-10, 2011.

[10] P. Benner and T. Breiten, *Low rank methods for a class of generalized Lyapunov equations and related issues*, Numer. Math., 124 (2013), pp. 441–470.

[11] P. Benner and T. Damm, *Lyapunov equations, energy functionals, and model order reduction of bilinear and stochastic systems*, SIAM J. Control Optim., 49 (2011), pp. 686–711.

[12] P. Benner and P. Kürschner, *Computing Real Low-Rank Solutions of Sylvester Equations by the Factored ADI Method*, MPI Magdeburg Preprint MPIMD/13-05, May 2013.

[13] P. Benner, J.-R. Li, and T. Penzl, *Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems*, Numer. Linear Algebra Appl., 15 (2008), pp. 755–777.

[14] P. Benner, R. Li, and N. Truhar, *On the ADI method for Sylvester equations*, J. Comput. Appl. Math., 233 (2009), pp. 1035–1045.

[15] P. Benner and E. Quintana-Ortí, *Solving stable generalized Lyapunov equations with the matrix sign function*, Numer. Algorithms, 20 (1999), pp. 75–100.

[16] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[17] M. Benzi, E. Haber, and L. Taralli, *A preconditioning technique for a class of PDE-constrained optimization problems*, Adv. Comput. Math., 35 (2011), pp. 149–173.

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

[19] J. Boyle, M. D. Mihajlovic, and J. A. Scott, *HSL_MI*20*: An Efficient AMG Preconditioner*, Tech. report RAL-TR-2007-021, Rutherford Appleton Laboratory (CCLRC), 2007.

[20] A. N. Brooks and T. J. Hughes, *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg., 32 (1982), pp. 199–259.

[21] J. Cahouet and J. Chabard, *Some fast* 3*D finite element solvers for the generalized Stokes problem*, Internat. J. Numer. Methods Fluids, 8 (1988), pp. 869–895.

[22] T. Damm, *Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations*, Numer. Linear Algebra Appl., 15 (2008), pp. 853–871.

[23] T. Davis, *UMFPACK Version* 4.4 *User Guide*, Technical report, Department of Computer and Information Science and Engineering, University of Florida, Gainesville, 2005.

[24] S. Dolgov, B. N. Khoromskij, I. Oseledets, and E. Tyrtyshnikov, *A reciprocal preconditioner for structured matrices arising from elliptic problems with jumping coefficients*, Linear Algebra Appl., 436 (2012), pp. 2980–3007.

[25] V. Druskin, L. Knizhnerman, and V. Simoncini, *Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation*, SIAM J. Numer. Anal., 49 (2011), pp. 1875–1898.

[26] I. S. Duff, A. M. Erisman, and J. K. Reid, *Direct Methods for Sparse Matrices*, Monogr. Numer. Anal., Oxford University Press, New York, 1989.

[27] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*, Numer. Math. Sci. Comput., Oxford University Press, New York, 2005.

[28] A. Eppler and M. Bollhöfer, *A structure preserving FGMRES method for solving large Lyapunov equations*, in Progress in Industrial Mathematics at ECMI 2010, M. Günther, A. Bartel, M. Brunk, S. Schöps, and M. Striebel, eds., Math. Ind., Springer-Verlag, Berlin, 2012, pp. 131–136.

[29] R. Falgout, *An introduction to algebraic multigrid*, Comput. Sci. Eng., 8 (2006), pp. 24–33.

[30] G. Flagg and S. Gugercin, *On the ADI method for the Sylvester equation and the optimal-$\mathcal{H}_2$ points*, Appl. Numer. Math., 64 (2013), pp. 50–58.

[31] R. Fletcher, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis, Lecture Notes in Math. 506, Springer, Berlin, 1976, pp. 73–89.

[32] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins Stud. Math. Sci., Johns Hopkins University Press, Baltimore, MD, 1996.

[33] G. H. Golub and R. S. Varga, *Chebyshev semi-iterative methods, successive over-relaxation iterative methods, and second order Richardson iterative methods.* I, Numer. Math., 3 (1961), pp. 147–156.

[34] G. H. Golub and R. S. Varga, *Chebyshev semi-iterative methods, successive over-relaxation iterative methods, and second order Richardson iterative methods.* II, Numer. Math., 3 (1961), pp. 157–168.

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

[36] L. Grasedyck, *Existence of a low rank or -matrix approximant to the solution of a Sylvester equation*, Numer. Linear Algebra Appl., 11 (2004), pp. 371–389.

[37] L. Grasedyck and W. Hackbusch, *A multigrid method to solve large scale Sylvester equations*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 870–894.

[38] A. Griewank and A. Walther, *Algorithm* 799: *Revolve: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation*, ACM Trans. Math. Software, 26 (2000), pp. 19–45.

[39] E. Haber and U. M. Ascher, *Preconditioned all-at-once methods for large, sparse parameter estimation problems*, Inverse Problems, 17 (2001), pp. 1847–1864.

[40] E. Haber, U. M. Ascher, and D. W. Oldenburg, *Inversion of* 3d *electromagnetic data in frequency and time domain using an inexact all-at-once approach*, Geophysics, 69 (2004), pp. 1216–1228.

[41] W. Hackbusch, *Multigrid Methods and Applications*, Springer Ser. Comput. Math. 4, Springer-Verlag, Berlin, 1985.

[42] M. Heinkenschloss, *A time-domain decomposition iterative method for the solution of distributed linear quadratic optimal control problems*, J. Comput. Appl. Math., 173 (2005), pp. 169–198.

[43] M. Heinkenschloss and D. Leykekhman, *Local error estimates for SUPG solutions of advection-dominated elliptic linear-quadratic optimal control problems*, SIAM J. Numer. Anal., 47 (2010), pp. 4607–4638.

[44] D. Hinrichsen and A. Pritchard, *Mathematical Systems Theory* I: *Modelling, State Space Analysis, Stability and Robustness*, Vol. 1, Springer-Verlag, Berlin, 2005.

[45] M. Hinze, *A variational discretization concept in control constrained optimization: The linear-quadratic case*, Comput. Optim. Appl., 30 (2005), pp. 45–61.

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

[47] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE Constraints*, Math. Model. Theory Appl., Springer-Verlag, New York, 2009.

[48] M. E. Hochstenbach, *A Jacobi-Davidson type SVD method*, SIAM J. Sci. Comput, 23 (2001), pp. 606–628.

[49] K. Ito and K. Kunisch, *Lagrange Multiplier Approach to Variational Problems and Applications*, Adv. Design Control 15, SIAM, Philadelphia, 2008.

[50] I. Jaimoukha and E. Kasenally, *Krylov subspace methods for solving large Lyapunov equations*, SIAM J. Numer. Anal., 31 (1994), pp. 227–251.

[51] K. Jbilou and A. J. Riquet, *Projection methods for large Lyapunov matrix equations*, Linear Algebra Appl., 415 (2006), pp. 344–358.

[52] H. Knobloch and H. Kwakernaak, *Lineare Kontrolltheorie*, Springer-Verlag, Berlin, 1985.

[53] M. Kollmann and M. Kolmbauer, *A preconditioned MinRes solver for time-periodic parabolic optimal control problems*, Numer. Linear Algebra Appl., 20 (2013), pp. 761–784.

[54] 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.

[55] D. Kressner and C. Tobler, *Low-rank tensor Krylov subspace methods for parametrized linear systems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1288–1316.

[56] J.-R. Li and J. White, *Low rank solution of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 260–280.

[57] T. P. Mathew, M. Sarkis, and C. E. Schaerer, *Analysis of block parareal preconditioners for parabolic optimal control problems*, SIAM J. Sci. Comput., 32 (2010), pp. 1180–1200.

[58] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer Ser. Oper. Res. Financ. Eng., Springer, New York, 2006.

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

[60] J. W. Pearson, M. Stoll, and A. Wathen, *Preconditioners for state constrained optimal control problems with Moreau-Yosida penalty function*, Numer. Linear Algebra Appl., 21 (2014), pp. 81–97.

[61] 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.

[62] J. W. Pearson and A. J. Wathen, *Fast Iterative Solvers for Convection-Diffusion Control Problems*, Electron. Trans. Numer. Anal., 40 (2013), pp. 294–310.

[63] J. W. Pearson and A. J. Wathen, *A new approximation of the Schur complement in preconditioners for PDE-constrained optimization*, Numer. Linear Algebra Appl., 19 (2012), pp. 816–829.

[64] T. Penzl, *A Cyclic Low Rank Smith Method for Large, Sparse Lyapunov Equations with Applications in Model Reduction and Optimal Control*, Technical report SFB393/98-6, Fakultät für Mathematik, TU Chemnitz, Germany, 1998.

[65] T. Penzl, *Eigenvalue decay bounds for solutions of Lyapunov equations: The symmetric case*, Systems Control Lett., 40 (2000), pp. 139–144.

[66] A. Ramage, *A multigrid preconditioner for stabilised discretisations of advection-diffusion problems*, J. Comput. Appl. Math., 110 (1999), pp. 187–203.

[67] T. Rees, *Preconditioning Iterative Methods for PDE Constrained Optimization*, Ph.D. thesis, University of Oxford, UK, 2010.

[68] T. Rees, H. S. Dollar, and A. J. Wathen, *Optimal solvers for PDE-constrained optimization*, SIAM J. Sci. Comput., 32 (2010), pp. 271–298.

[69] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid Methods, Frontiers in Appl. Math. 3, SIAM, Philadelphia, 1987, pp. 73–130.

[70] Y. Saad, *Numerical solution of large Lyapunov equation*, in Signal Processing, Scattering, Operator Theory and Numerical Methods, M. A. Kaashoek, J. H. van Schuppen, and A. C. M. Ran, eds., Birkhauser, Basel, 1990, pp. 503–511.

[71] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–461.

[72] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[73] J. Schöberl and W. Zulehner, *Symmetric indefinite preconditioners for saddle point problems with applications to PDE-constrained optimization problems*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 752–773.

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

[75] V. Simoncini, *The extended Krylov subspace for parameter dependent systems*, Appl. Numer. Math., 60 (2010), pp. 550–560.

[76] V. Simoncini, *Reduced order solution of structured linear systems arising in certain PDE-constrained optimization problems*, Comput. Optim. Appl., to appear.

[77] V. Simoncini, *Computational Methods for Linear Matrix Equations*, Technical report, Università di Bologna, 2013.

[78] D. Sorensen and A. Antoulas, *The Sylvester equation and approximate balanced reduction*, Linear Algebra Appl., 351–352 (2002), pp. 671–700.

[79] M. Stoll, *A Krylov-Schur approach to the truncated SVD*, Linear Algebra Appl., 436 (2012), pp. 2795–2806.

[80] M. Stoll, *All-at-once solution of a time-dependent time-periodic PDE-constrained optimization problems*, IMA J. Numer. Anal., 34 (2014), pp. 1554–1577.

[81] M. Stoll and A. Wathen, *All-at-Once Solution of Time-Dependent PDE-Constrained Optimization Problems*, Technical report, University of Oxford, UK, 2010.

[82] M. Stoll and A. Wathen, *All-at-once solution of time-dependent Stokes control*, J. Comput. Phys., 232 (2013), pp. 498–515.

[83] T. Sun, *Discontinuous Galerkin finite element method with interior penalties for convection diffusion optimal control problem*, Int. J. Numer. Anal. Model, 7 (2010), pp. 87–107.

[84] F. Tröltzsch, *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*, AMS, Providence, RI, 2010.

[85] B. Vandereycken and S. Vandewalle, *A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2553–2579.

[86] A. J. Wathen and T. Rees, *Chebyshev semi-iteration in preconditioning for problems including the mass matrix*, Electron. Trans. Numer. Anal., 34 (2008), pp. 125–135.

[87] P. Wesseling, *An Introduction to Multigrid Methods*, Pure Appl. Math. (N.Y.), John Wiley, New York, 1992.