A Parallel-in-Time Gradient-Type Method for Discrete Time Optimal Control Problems *

Xiaodi Deng † **Matthias Heinkenschloss** ‡

May 3, 2016

Abstract

This paper introduces and analyzes a new parallel-in-time gradient type method for the solution of convex linear-quadratic discrete-time optimal control (DTOC) problems. Each iteration of the classical gradient method requires the solution of the forward-in-time state equation followed by the solution of the backward-in-time adjoint equation to compute the gradient. To introduce parallelism, the time steps are split into N groups corresponding to time subintervals. At the time subinterval boundaries state and adjoint information from the previous iteration is used. On each time subinterval the forward-in-time state equation is solved, the backward-in-time adjoint equation is solved, gradient-type information is generated, and the control are updated. These computations can be performed in parallel across time subintervals. State and adjoint information at time subinterval boundaries is then exchanged with neighboring subintervals and the process is repeated. The resulting iteration can be interpreted as a so-called $(2N-1)$-part iteration scheme. Convergence of the new parallel-in-time gradient type method is proven for suitable step-sizes by showing that an associated block companion matrix has spectral radius less than one. The performance of the new method is demonstrated on a DTOC problem obtained from a discretization of a 3D parabolic optimal control problem. In this example nearly perfect speed-up is observed for moderate number of time subdomains. This speed-up due to time decomposition multiplies existing speed-up due to parallelization in the solution of state and adjoint equations.

Key words Optimal control, gradient method, iterative methods for linear systems, parallel computation

AMS subject classifications 49M05, 49M27, 65F10

*This work was supported in part by NSF grant DMS-1522798, by the Data Analysis and Visualization Cyberinfrastructure funded by NSF under grant OCI-0959097 and Rice University, and a sponsored research agreement with the ExxonMobil Upstream Research Company.

†Department of Computational and Applied Mathematics, MS-134, Rice University, 6100 Main Street, Houston, TX 77005-1892. E-mail: Xiaodi.Deng@rice.edu

‡Department of Computational and Applied Mathematics, MS-134, Rice University, 6100 Main Street, Houston, TX 77005-1892. E-mail: heinken@rice.edu

1 Introduction

We introduce and analyze a new parallel-in-time gradient-type method for convex linear-quadratic discrete-time optimal control (DTOC) problems. Each iteration of the standard gradient method applied to DTOC problems requires the forward in time solution of the state equation, followed by the backward in time solution of the so-called adjoint equation before the gradient can be computed and the control can be updated. To introduce parallelism our method splits the time steps into N groups, corresponding to time subintervals. At the boundary index of each time subinterval we use state and adjoint information from the previous iteration. This allows the computation of state, adjoint, and gradient, and control update on each time subinterval in parallel. After states and adjoints are computed on each time subinterval state and adjoint information at boundary indices of time subintervals are exchanged with neighboring time subintervals. Since for given controls, the state and adjoint equations are typically not satisfied on the entire time interval, there are 'jumps' in the states and adjoints at indices corresponding time subinterval boundaries, we refer to our scheme as a gradient-type method.

We show that the resulting iteration can be interpreted as a so-called $(2N-1)$-part iteration scheme as introduced in [6], [7]. However, in our context the Hessian matrix and its $(2N-1)$-part splitting is never explicitly computed. Moreover, the convergence results in [6] [7] are for specific splittings only and do not apply to our setting. We use the structure of the $(2N-1)$-part iteration scheme and properties of a corresponding block companion matrix to prove convergence of our new parallel-in-time gradient-type method for sufficiently small step-size.

This work is motivated by linear-quadratic optimal control problems governed by parabolic partial differential equations (PDEs) and our example problems are optimal control problems governed by time dependent advection diffusion PDE. We use a 1D problem to illustrate some of the theoretical convergence results, and we use a 3D problem to illustrate the parallel performance of parallel-in-time gradient-type method. For the 3D example problem we observe nearly perfect speed-up for moderate number N of time subdomains. The number N up to which nearly perfect speed-up is observed depends on problem data such as length of the time domain. Our specific example excellent speed-ups are obtained for $N \approx 20$ and $N \approx 50$ time subdomains. It is also important to note that the speed-up due to time decomposition multiplies existing speed-up in the solution of state and adjoint equations, e.g, due to parallel solution of linear systems arising in the time-stepping.

Time-decomposition based methods for linear quadratic optimal control problems have recently been proposed in [1], [5], [10], [18] [2], [15], [12], [4]. Specifically, [1], [5], [10] use time decomposition for parabolic optimal control problems. In [5], [10] the time subdomains are non-overlapping, while [1] use overlaps. All iterative methods in [1], [5], [10] approximately solve time-subdomain optimal control problems and do not directly generalize to nonlinear, non-convex problems. The papers [2], [15] use time domain decomposition based reformulations to derive constrained optimization problems in which the control variables and states at time-domain interfaces are the optimization variables, and apply augmented Lagrangian-type methods for their

solution. The papers [12], [4], explore the connection between time domain decompsition and multiple shooting methods, and discuss space-time adaptation of finite element discretizations, but use standard optimization approaches. Finally the paper [18] introduces a 'parareal' based preconditioner for optimality systems for linear-quadratic parabolic optimal control problems. Although the optimization formulation itself does not use time-decomposition, the 'parareal' method is used for the solution of systems related to the state and adjoint equations and it introduces parallel-in-time computations into the preconditioner. Our parallel-in-time gradient-type method resembles the classical gradient method. A given implementation of the classical gradient method can be easily modified to implement our parallel-in-time gradient-type method. Conceptually our parallel-in-time gradient-type method can be generalized to general nonlinear problems, although our current theory does not capture such extensions.

This paper is organized as follows. In Section 2 we state the DTOC problem and present the classical gradient method for its solution. The results in this section are well known, but the notation introduced is important for the presentation and analysis of our method. Section 3 introduces the parallel-in-time gradient-type method. We then interpret this new method as a $(2N-1)$-part iteration scheme. The convergence of this $(2N-1)$-part iteration scheme is determined by the spectrum of a block companion matrix, parameterized by the step-size of gradient-type method. We prove that for sufficiently small positive step-size the spectral radius of the companion matrix is less than one and, therefore, that our parallel-in-time gradient-type method converges with sufficiently small positive step-size selection. In Section 4 we illustrate the performance of our new method on DTOC problems obtained from discretizations of optimal control problems governed by time dependent advection diffusion PDEs.

2 Gradient Method

We consider DTOC problems with state and control variables $y_0,...,y_K \in \mathbb{R}^{n_y}$ and $u_0,...,u_{K-1} \in \mathbb{R}^{n_u}$. Given symmetric positive semidefinite matrices $Q_1,...,Q_K \in \mathbb{R}^{n_y \times n_y}$, symmetric positive definite matrices $R_0,...,R_{K-1} \in \mathbb{R}^{n_u \times n_u}$, matrices $A_0,...,A_{K-1} \in \mathbb{R}^{n_y \times n_y}$, $B_0,...,B_{K-1} \in \mathbb{R}^{n_y \times n_u}$, and vectors $d_1,...,d_K, c_0,...,c_{K-1} \in \mathbb{R}^{n_y}$, $e_0,...,e_{K-1} \in \mathbb{R}^{n_u}$, the DTOC problem is given by

$$\text{minimize} \sum_{k=1}^{K} \left[\frac{1}{2} y_k^T Q_k y_k + d_k^T y_k + \frac{1}{2} u_{k-1}^T R_{k-1} u_{k-1} + e_{k-1}^T u_{k-1}\right] \tag{2.1a}$$

$$\text{subject to } y_0 = y_{\text{given}}, \tag{2.1b}$$

$$y_{k+1} = A_k y_k + B_k u_k + c_k, \quad k = 0,...,K-1. \tag{2.1c}$$

We can use the constrains (2.1c) to express y_k as a function of $u_0,\ldots,u_{k-1}$. This leads to the following unconstrained formulation of (2.1).

$$\text{Minimize}_{u_0,...,u_{K-1}} J(u_0,...,u_{K-1}), \tag{2.2a}$$

where

$$J(u_0,...,u_{K-1}) = \sum_{k=1}^{K} \Big[\frac{1}{2} y_k(u_0,\ldots,u_{k-1})^T Q_k y_k(u_0,\ldots,u_{k-1}) + d_k^T y_k(u_0,\ldots,u_{k-1}) + \frac{1}{2} u_{k-1}^T R_{k-1} u_{k-1} + e_{k-1}^T u_{k-1}\Big]. \tag{2.2b}$$

The problems (2.1) and (2.2) are equivalent.

It is well-known that the gradient of J can be computed using the adjoint equation approach. See, e.g., the books [3, Sec. 1.9], [14, Sec. 2.4]. The gradient can be computed as follows.

Compute $y_0,\ldots,y_K$ by solving

$$y_0 = y_{\text{given}}, \tag{2.3a}$$
$$y_{k+1} = A_k y_k + B_k u_k + c_k \qquad \text{for } k = 0,...,K-1. \tag{2.3b}$$

Compute $p_0,\ldots,p_{K-1}$ by solving

$$p_{K-1} = d_K + Q_K y_K, \tag{2.3c}$$
$$p_{k-1} = d_k + Q_k y_k + A_k^T p_k \qquad \text{for } k = K-1,...,1. \tag{2.3d}$$

The gradient is given by

$$\nabla_{u_k} J(u_0,...,u_{K-1}) = R_k u_k + e_k + B_k^T p_k \qquad \text{for } k = 0,...,K-1. \tag{2.3e}$$

One step of the gradient method is listed in Algorithm 2. We use subscripts k to denote time steps and superscripts (j) to denote the iteration in the gradient method.

Algorithm 1 jth iteration of the gradient method with step size $\alpha > 0$

Given control $u_0^{(j)},\ldots,u_{K-1}^{(j)}$, and initial state $y_0^{(j)} = y_{\text{given}}$.
for $k = 0,...,K-1$ **do** ▷ solve state equation forward in time
    Compute $y_{k+1}^{(j)} = A_k y_k^{(j)} + B_k u_k^{(j)} + c_k$
end for
Compute $p_{K-1}^{(j)} = d_K + Q_K y_K^{(j)}$ ▷ solve adjoint equation backward in time
for $k = k-1,...,1$ **do**
    Compute $p_{k-1}^{(j)} = d_k + Q_k y_k^{(j)} + A_k^T p_k^{(j)}$
end for
for $k = 0,...,K-1$ **do** ▷ update control using negative gradient
    $u_k^{(j+1)} = u_k^{(j)} - \alpha(R_k u_k^{(j)} + e_k + B_k^T p_k^{(j)})$
end for

For the following presentation it will be helpful to write the gradient method in a more compact matrix-vector notation. We define the vectors

$$\mathbf{u} \stackrel{\text{def}}{=} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{K-1} \end{bmatrix} \in \mathbb{R}^{Kn_u}, \qquad \mathbf{y} \stackrel{\text{def}}{=} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix}, \ \mathbf{p} \stackrel{\text{def}}{=} \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_{K-1} \end{bmatrix}, \ \mathbf{y}_0 \stackrel{\text{def}}{=} \begin{bmatrix} A_0 y_0 \\ A_1 A_0 y_0 \\ \vdots \\ A_{K-1} \ldots A_0 y_0 \end{bmatrix} \in \mathbb{R}^{Kn_y}, \tag{2.4a}$$

the matrix

$$\mathbf{L} \stackrel{\text{def}}{=} \left[\begin{array}{c|c|c|c|c|c} I & & & & & \\ A_1 & I & & & & \\ A_2A_1 & A_2 & I & & & \\ A_3A_2A_1 & A_3A_2 & A_3 & I & & \\ \vdots & & & & \ddots & \\ A_{K-1}A_{K-2}\cdots A_1 & A_{K-1}A_{K-2}\cdots A_2 & \cdots & \cdots & A_{K-1} & I \end{array}\right] \in \mathbb{R}^{Kn_y \times Kn_y}, \tag{2.4b}$$

and the quantities

$$\mathbf{Q} \stackrel{\text{def}}{=} diag(Q_1, ..., Q_K), \qquad \mathbf{R} \stackrel{\text{def}}{=} diag(R_0, ..., R_{K-1}), \qquad \mathbf{B} \stackrel{\text{def}}{=} diag(B_0, ..., B_{K-1}), \tag{2.4c}$$

$$\mathbf{c} \stackrel{\text{def}}{=} (c_0^T, ..., c_{K-1}^T)^T, \qquad \mathbf{d} \stackrel{\text{def}}{=} (d_1^T, ..., d_K^T)^T, \qquad \mathbf{e} \stackrel{\text{def}}{=} (e_0^T, ..., e_{K-1}^T)^T. \tag{2.4d}$$

The state computations (2.3a,b) can be written as

$$\mathbf{y} = \mathbf{L}(\mathbf{B}\mathbf{u} + \mathbf{c}) + \mathbf{y}_0, \tag{2.5}$$

the adjoint computations (2.3c,d) can be written as

$$\mathbf{p} = \mathbf{L}^T(\mathbf{Q}\mathbf{y} + \mathbf{d}), \tag{2.6}$$

and the gradient (2.3e) is given by $\nabla J(\mathbf{u}) = \mathbf{R}\mathbf{u} + \mathbf{e} + \mathbf{B}^T\mathbf{p}$. If we insert (2.6) and (2.5) into the previous expression for $\nabla J(\mathbf{u})$, then

$$\begin{aligned} \nabla J(\mathbf{u}) &= \left[\mathbf{R} + (\mathbf{L}\mathbf{B})^T\mathbf{Q}(\mathbf{L}\mathbf{B})\right]\mathbf{u} + \mathbf{e} + (\mathbf{L}\mathbf{B})^T\mathbf{Q}(\mathbf{L}\mathbf{c} + \mathbf{y}_0) + (\mathbf{L}\mathbf{B})^T\mathbf{d} \\ &= \mathbf{H}\mathbf{u} + \mathbf{g}, \end{aligned} \tag{2.7}$$

where

$$\mathbf{H} \stackrel{\text{def}}{=} \mathbf{R} + (\mathbf{L}\mathbf{B})^T\mathbf{Q}(\mathbf{L}\mathbf{B}), \quad \mathbf{g} \stackrel{\text{def}}{=} \mathbf{e} + (\mathbf{L}\mathbf{B})^T\mathbf{Q}(\mathbf{L}\mathbf{c} + \mathbf{y}_0) + (\mathbf{L}\mathbf{B})^T\mathbf{d}. \tag{2.8}$$

Since $R_0, ..., R_{K-1} \in \mathbb{R}^{n_u \times n_u}$ symmetric positive definite and $Q_1, ..., Q_K \in \mathbb{R}^{n_y \times n_y}$ are symmetric positive semidefinite, $\mathbf{H}$ is symmetric positive definite. Alternatively, we can insert (2.5) into (2.2b) to obtain

$$J(\mathbf{u}) = \frac{1}{2}\mathbf{u}^T\mathbf{H}\mathbf{u} + \mathbf{g}^T\mathbf{u} + \text{const.} \tag{2.9}$$

and derive the expression (2.7) this way.

We also recall that the gradient method with constant step-size α, in this context also known as Richardson's iteration,

$$\mathbf{u}^{(j+1)} = \mathbf{u}^{(j)} - \alpha(\mathbf{H}\mathbf{u}^{(j)} + \mathbf{g}), \tag{2.10}$$

is an iterative method derived from the splitting $\mathbf{H} = \alpha^{-1}\mathbf{I} - (\alpha^{-1}\mathbf{I} - \mathbf{H})$. Let $0 < \lambda_{min}(\mathbf{H}) \leq \lambda_{max}(\mathbf{H})$ denote the smallest and largest eigenvalue of $\mathbf{H}$. The spectral radius of the iteration matrix $\mathbf{I} - \alpha\mathbf{H}$ is less than one for all step sizes $0 < \alpha < 2/\lambda_{max}(\mathbf{H})$. The smallest spectral radius $\rho^* = (\lambda_{max}(\mathbf{H}) - \lambda_{min}(\mathbf{H}))/(\lambda_{max}(\mathbf{H}) + \lambda_{min}(\mathbf{H}))$ is achieved with step size $\alpha^* = 2/(\lambda_{max}(\mathbf{H}) + \lambda_{min}(\mathbf{H}))$.

3 Parallel Gradient-Type Method

3.1 Derivation of the Method

The gradient method requires a full forward-in-time state solve followed by a full backward-in-time adjoint solve before the control can be updated. Our parallel-in-time gradient method splits the time indices into N subsets, where the ith one is given by $\{K_i, \ldots, K_{i+1}\}$. $i = 0, \ldots, N-1$, and the time indices satisfy $0 = K_0 < K_1 < \ldots < K_N = K$. Since the indices $K_i, \ldots, K_{i+1}$ correspond to time steps, we refer to $\{K_i, \ldots, K_{i+1}\}$ as the ith time subinterval, and refer to the indices K_i, K_{i+1} as the left/right time interval boundary.

The idea of our parallel-in-time gradient-type method is simple. Suppose we are given the current control on the ith time subinterval, $u_{K_i}^{(j)}, u_{K_i+1}^{(j)}, \ldots, u_{K_{i+1}-1}^{(j)}$. If we knew the state information $y_{K_i}^{(j)}$ at the left time interval boundary K_i and the adjoint information $p_{K_{i+1}}^{(j)}$ at the right time interval boundary K_{i+1}, then we can compute $y_k^{(j)}$, $k = K_i + 1, \ldots, K_{i+1}$, the adjoints $p_k^{(j)}$, $k = K_{i+1} - 1, \ldots, K_i$, and then update the controls with indices $K_i, K_i + 1, \ldots, K_{i+1} - 1$. Since $y_{K_i}^{(j)}$ and $p_{K_{i+1}}^{(j)}$ are only available through an entire state and adjoint solve, we use the values from the previous iteration, and exchange these values after time subinterval computations for state and adjoint information are completed. Thus for the ith time subinterval, $i \in \{1, \ldots, N-2\}$, we proceed as follows (for $i = 0$ and $i = N-1$ initial state information or final adjoint information is given): Given $u_{K_i}^{(j)}, u_{K_i+1}^{(j)}, \ldots, u_{K_{i+1}-1}^{(j)}$ and $y_{K_i}^{(j-1)}$ and $p_{K_{i+1}}^{(j-1)}$ we first compute ith time subinterval states using

$$y_{K_i+1}^{(j)} = A_{K_i} y_{K_i}^{(j-1)} + B_{K_i} u_{K_i}^{(j)} + c_{K_i}, \tag{3.1a}$$

$$y_{k+1}^{(j)} = A_k y_k^{(j)} + B_k u_k^{(j)} + c_k, \qquad k = K_i + 1, \ldots, K_{i+1} - 1. \tag{3.1b}$$

Next we compute ith time subinterval adjoints using

$$p^{(j)}_{K_{i+1}-1} = Q_{K_{i+1}} y^{(j)}_{K_{i+1}} + d_{K_{i+1}} + A^T_{K_{i+1}} p^{(j-1)}_{K_{i+1}}, \tag{3.1c}$$

$$p^{(j)}_{k-1} = Q_k y^{(j)}_k + d_k + A^T_k p^{(j)}_k, \qquad k = K_{i+1}-1,\ldots,K_i+1. \tag{3.1d}$$

Then we update the ith time subinterval controls using

$$u^{(j+1)}_k = u^{(j)}_k - \alpha(R_k u^{(j)}_k + e_k + B^T_k p^{(j)}_k), \qquad k = K_i,\ldots,K_{i+1}-1. \tag{3.1e}$$

Finally, we send $y^{(j)}_{K_{i+1}}$ to processor $i+1$ and $p^{(j)}_{K_i}$ to processor $i-1$, and we receive $y^{(j)}_{K_i}$ from processor $i-1$ and $p^{(j)}_{K_{i+1}}$ from processor $i+1$.

The complete statement, taking into account the modifications for time subintervals $i=0$ and $i=N-1$ is given in Algorithm 3.1. Since at given controls $u^{(j)}_0,\ldots,u^{(j)}_{K-1}$ the state equation (2.3a,b) and the adjoint equation (2.3c,d) are not satisfied (there are 'jumps' at the time interval boundaries $K_1,\ldots,K_{N-1}$) when Algorithm 3.1 is used, we call it 'gradient-type'.

Algorithm 2 jth iteration of the parallel-in-time gradient-type method with step size $\alpha > 0$. Describes the tasks executed by processor of rank $i \in \{0, ..., N-1\}$

Input control $u_{K_i}^{(j)}, u_{K_i+1}^{(j)}, \ldots, u_{K_{i+1}-1}^{(j)}$. ▷ initialization of the iteration
if $i > 0$ and $j = 0$ **then**
    Input initial $y_{K_i}^{(-1)}$
end if
if $i < N-1$ and $j = 0$ **then**
    Input initial $p_{K_{i+1}}^{(-1)}$
end if

if $i = 0$ **then** ▷ solve the state equation forward in time
    $y_{K_i+1}^{(j)} = A_{K_i} y_{\text{given}} + B_{K_i} u_{K_i}^{(j)} + c_{K_i}$
else
    $y_{K_i+1}^{(j)} = A_{K_i} y_{K_i}^{(j-1)} + B_{K_i} u_{K_i}^{(j)} + c_{K_i}$
end if
for $k = K_i + 1, \ldots, K_{i+1} - 1$ **do**
    $y_{k+1}^{(j)} = A_k y_k^{(j)} + B_k u_k^{(j)} + c_k$
end for

if $i = N-1$ **then** ▷ solve the adjoint equation backward in time
    $p_{K_{i+1}-1}^{(j)} = Q_{K_{i+1}} y_{K_{i+1}}^{(j)} + d_{K_{i+1}}$
else
    $p_{K_{i+1}-1}^{(j)} = Q_{K_{i+1}} y_{K_{i+1}}^{(j)} + d_{K_{i+1}} + A_{K_{i+1}}^T p_{K_{i+1}}^{(j-1)}$
end if
for $k = K_{i+1} - 1, \ldots, K_i + 1$ **do**
    $p_{k-1}^{(j)} = Q_k y_k^{(j)} + d_k + A_k^T p_k^{(j)}$
end for

for $k = K_i, \ldots, K_{i+1} - 1$ **do** ▷ update control
    $u_k^{(j+1)} = u_k^{(j)} - \alpha(R_k u_k^{(j)} + e_k + B_k^T p_k^{(j)})$
end for

if $i > 0$ **then** ▷ communication between processors
    send $p_{K_i}^{(j)}$ to rank $i-1$
    receive $y_{K_i}^{(j)}$ from rank $i-1$
end if
if $i < N-1$ **then**
    send $y_{K_{i+1}}^{(j)}$ to rank $i+1$
    receive $p_{K_{i+1}}^{(j)}$ from rank $i+1$
end if

3.2 Interpretation as a $(2N-1)$-Part Iteration Scheme

We express the parallel gradient-type method using a compact form, which will allow us to interpret this new method as a $(2N-1)$-part iteration scheme corresponding to (2.10), and which will form the basis of our convergence proof.

We define the ordered product of matrices

$$\prod_{h=i}^{j} A_h \stackrel{\text{def}}{=} \begin{cases} A_j A_{j-1} \times \ldots \times A_{i+1} A_i, & i \le j, \\ I, & i > j. \end{cases}$$

Using this notation and recursive substitutions of (3.1a,b) we obtain the following. For $j \ge 0$, $0 \le i \le N-1$, and $1 \le s \le K_{i+1} - K_i$, note that $y_{K_i}^{(-1)}$ is initialized when the algorithm starts,

$$y_{K_i+s}^{(j)} = \Big[\prod_{h=K_i}^{K_i+s-1} A_h\Big] y_{K_i}^{(j-1)} + \sum_{t=0}^{s-1} \Big[\prod_{h=K_i+s-t}^{K_i+s-1} A_h\Big] (B_{K_i+s-t-1} u_{K_i+s-t-1}^{(j)} + c_{K_i+s-t-1}). \tag{3.2}$$

For $j \ge 1$ and $i \ge 1$, applying (3.2) with j replaced by $j-1$ and K_i replaced by K_{i-1}, and with $s = K_i - K_{i-1}$ gives

$$y_{K_i}^{(j-1)} = \Big[\prod_{h=K_{i-1}}^{K_i-1} A_h\Big] y_{K_{i-1}}^{(j-2)} + \sum_{t=0}^{K_i-K_{i-1}-1} \Big[\prod_{h=K_i-t}^{K_i-1} A_h\Big] (B_{K_i-t-1} u_{K_i-t-1}^{(j-1)} + c_{K_i-t-1}). \tag{3.3}$$

Inserting (3.3) into (3.2) gives for $j \ge 1$ and $i \ge 1$,

$$\begin{aligned} y_{K_i+s}^{(j)} = \Big[\prod_{h=K_{i-1}}^{K_i+s-1} A_h\Big] y_{K_{i-1}}^{(j-2)} &+ \sum_{t=0}^{K_i-K_{i-1}-1} \Big[\prod_{h=K_i-t}^{K_i+s-1} A_h\Big] (B_{K_i-t-1} u_{K_i-t-1}^{(j-1)} + c_{K_i-t-1}) \\ &+ \sum_{t=0}^{s-1} \Big[\prod_{h=K_i+s-t}^{K_i+s-1} A_h\Big] (B_{K_i+s-t-1} u_{K_i+s-t-1}^{(j)} + c_{K_i+s-t-1}). \end{aligned} \tag{3.4}$$

Repeating the previous steps leads to the following expression for $j \ge i$,

$$\begin{aligned} y_{K_i+s}^{(j)} = &\Big[\prod_{h=0}^{K_i+s-1} A_h\Big] y_{\text{given}} \\ &+ \sum_{d=1}^{i} \sum_{t=0}^{K_{i-d+1}-K_{i-d}-1} \Big[\prod_{h=K_{i-d+1}-t}^{K_i+s-1} A_h\Big] (B_{K_{i-d+1}-t-1} u_{K_{i-d+1}-t-1}^{(j-d)} + c_{K_{i-d+1}-t-1}) \\ &+ \sum_{t=0}^{s-1} \Big[\prod_{h=K_i+s-t}^{K_i+s-1} A_h\Big] (B_{K_i+s-t-1} u_{K_i+s-t-1}^{(j)} + c_{K_i+s-t-1}). \end{aligned} \tag{3.5}$$

Rearranging (3.5) into matrix-vector product form gives, for $j \geq i$,

$$
\begin{aligned}
y^{(j)}_{K_i+s} = & \Big[\prod_{h=0}^{K_i+s-1} A_h \Big] y_{\text{given}} \\
& + \sum_{d=1}^{i} \Big[\textstyle\prod_{h=K_{i-d}+1}^{K_i+s-1} A_h, \prod_{h=K_{i-d}+2}^{K_i+s-1} A_h, \ldots, \prod_{h=K_{i-d+1}}^{K_i+s-1} A_h \Big] \begin{bmatrix} B_{K_{i-d}} u^{(j-d)}_{K_{i-d}} + c_{K_{i-d}} \\ B_{K_{i-d}+1} u^{(j-d)}_{K_{i-d}+1} + c_{K_{i-d}+1} \\ \vdots \\ B_{K_{i-d+1}-1} u^{(j-d)}_{K_{i-d+1}-1} + c_{K_{i-d+1}-1} \end{bmatrix} \\
& + \Big[\textstyle\prod_{h=K_i+1}^{K_i+s-1} A_h, \prod_{h=K_i+2}^{K_i+s-1} A_h, \ldots, \prod_{h=K_i+s-1}^{K_i+s-1} A_h, \; I \Big] \begin{bmatrix} B_{K_i} u^{(j)}_{K_i} + c_{K_i} \\ B_{K_i+1} u^{(j)}_{K_i+1} + c_{K_i+1} \\ \vdots \\ B_{K_i+s-1} u^{(j)}_{K_i+s-1} + c_{K_i+s-1} \end{bmatrix} \qquad (3.6a) \\
= & \Big[\prod_{h=0}^{K_i+s-1} A_h \Big] y_{\text{given}} \\
& + \sum_{d=1}^{i} \Big[\overbrace{0, \ldots, 0}^{K_{i-d} \text{ zero blocks}}, \prod_{h=K_{i-d}+1}^{K_i+s-1} A_h, \prod_{h=K_{i-d}+2}^{K_i+s-1} A_h, \ldots, \prod_{h=K_{i-d+1}}^{K_i+s-1} A_h, \overbrace{0, \ldots, 0}^{(K-K_{i-d+1}) \text{ zero blocks}} \Big] (\mathbf{B}\mathbf{u}^{(j-d)} + \mathbf{c}) \\
& + \Big[\overbrace{0, \ldots, 0}^{K_i \text{ zero blocks}}, \prod_{h=K_i+1}^{K_i+s-1} A_h, \prod_{h=K_i+2}^{K_i+s-1} A_h, \ldots, \prod_{h=K_i+s-1}^{K_i+s-1} A_h, \; I, \overbrace{0, \ldots, 0}^{(K-K_i)-s \text{ zero blocks}} \Big] (\mathbf{B}\mathbf{u}^{(j)} + \mathbf{c}). \qquad (3.6b)
\end{aligned}
$$

In (3.6b) each zero block is of size $n_y \times n_y$. Notice that the row vectors in (3.6b) are row blocks of $\mathbf{L}$.

Equation (3.6b) represents the $(K_i + s)$th block (of length n_y) component of the vector $\mathbf{y}^{(j)}$. To combine (3.6b) into a representation for the entire vector $\mathbf{y}^{(j)}$ we define matrices $\mathbf{I}_{-d} \in \{0,1\}^{Kn_y \times Kn_y}$, $d = 0, ..., N-1$ as follows. We use Matlab notation to indicate submatrices. The matrix $\mathbf{I}_{-d}$ is a zero matrix except for the submatrices

$$
\mathbf{I}_{-d}(K_{i-1}n_y + 1 : K_i n_y, \; K_{i-1-d}n_y + 1 : K_{i-d}n_y), \quad i = d+1, ...N,
$$

which are matrices of all ones (i.e. the matrix entries of $\mathbf{I}_{-d}$ in the intersection of rows $K_{i-1}n_y + 1$ to $K_i n_y$ and of columns $K_{i-1-d}n_y + 1$ to $K_{i-d}n_y$ are equal to one). See Figure 3.1 for an illustration. Note that there are no overlapping nonzero entries for matrices $\mathbf{I}_{-d}$ for $d = 0, ..., N-1$ and that $\sum_{d=0}^{N-1} \mathbf{I}_{-d}$ has entires '1' in all positions where $\mathbf{L}$ is nonzero. Now, if '$\circ$' represents the Hadamard product, then (3.6) can be written as

$$
\mathbf{y}^{(j)} = \sum_{d=0}^{N-1} (\mathbf{I}_{-d} \circ \mathbf{L})(\mathbf{B}\mathbf{u}^{(j-d)} + \mathbf{c}) + \mathbf{y}_0, \qquad \text{for } j \geq N-1. \qquad (3.7)
$$

Similar to the derivation of (3.7) we can show that the equations (3.1c,d) for the adjoints lead

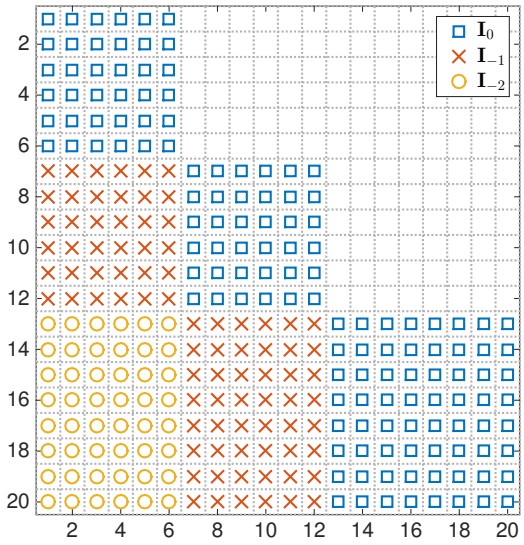

Figure 3.1: Illustration of the positions of '1's in $\mathbf{I}_0, \mathbf{I}_{-1}$ and $\mathbf{I}_{-2}$ in an example where the state dimension is $n_y = 2$, the $K = 10$ time steps are split into $N = 3$ subintervals with $K_0 = 0, K_1 = 3, K_2 = 6, K_3 = 10$.

to the compact representation

$$\mathbf{p}^{(j)} = \sum_{d=0}^{N-1} (\mathbf{I}_{-d} \circ \mathbf{L})^T (\mathbf{Q}\mathbf{y}^{(j-d)} + \mathbf{d}), \qquad \text{for } j \geq N-1. \tag{3.8}$$

Using (3.1e), (3.7) and (3.8) gives the following representation of our parallel-in-time gradient type iteration $j \geq 2N-2$,

$$\begin{aligned} \mathbf{u}^{(j+1)} &= \mathbf{u}^{(j)} - \alpha(\mathbf{R}\mathbf{u}^{(j)} + \mathbf{e} + \mathbf{B}^T \mathbf{p}^{(j)}) \\ &= \mathbf{u}^{(j)} - \alpha\Big[\mathbf{R}\mathbf{u}^{(j)} + \sum_{r=0}^{N-1} \sum_{l=0}^{N-1} (\mathbf{I}_{-r} \circ \mathbf{L}\mathbf{B})^T \mathbf{Q} (\mathbf{I}_{-l} \circ \mathbf{L}\mathbf{B}) \mathbf{u}^{(j-r-l)} + \mathbf{g}\Big]. \end{aligned} \tag{3.9}$$

We define

$$\mathbf{H}_0 \stackrel{\text{def}}{=} \mathbf{R} + (\mathbf{I}_0 \circ \mathbf{L}\mathbf{B})^T \mathbf{Q} (\mathbf{I}_0 \circ \mathbf{L}\mathbf{B}), \tag{3.10a}$$

$$\mathbf{H}_d \stackrel{\text{def}}{=} \sum_{\substack{l,r \in \{0,\ldots,N-1\} \\ l+r=d}} (\mathbf{I}_{-r} \circ \mathbf{L}\mathbf{B})^T \mathbf{Q} (\mathbf{I}_{-l} \circ \mathbf{L}\mathbf{B}), \qquad d = 1, \ldots, 2N-2. \tag{3.10b}$$

Note that the Hessian (2.8) can be split as

$$\mathbf{H} = \sum_{d=0}^{2N-2} \mathbf{H}_d. \tag{3.11}$$

Inserting (3.10) into (3.9) gives

$$\mathbf{u}^{(j+1)} = \mathbf{u}^{(j)} - \alpha\Big(\sum_{d=0}^{2N-2} \mathbf{H}_d \mathbf{u}^{(j-d)} + \mathbf{g}\Big). \tag{3.12}$$

The presentation (3.12) reveals that our parallel-in-time gradient-type method is a $(2N-1)$-part iteration scheme as defined in [6], [7] derived from the $(2N-1)$-part additive splitting

$$\mathbf{H} = \alpha^{-1}\mathbf{I} - (\alpha^{-1}\mathbf{I} - \mathbf{H}_0) - (-\mathbf{H}_1) - \ldots - (-\mathbf{H}_{2N-2}),$$

which results by adding/subtracting $\alpha^{-1}\mathbf{I}$ in (3.11).

The convergence results in [6] [7] are for specific splittings (in our notation, for specific matrices $\mathbf{H}, \mathbf{H}_0, \ldots, \mathbf{H}_{2N-2}$) only and do not apply to our setting. In the next section we prove convergence for sufficiently small step-size $\alpha > 0$.

3.3 Convergence Proof

Using the representation (2.9) it follows immediately that the optimal control $\mathbf{u}^{(*)}$ satisfies $\mathbf{H}\mathbf{u}^{(*)} + \mathbf{g} = \mathbf{0}$ and, using (3.11),

$$\mathbf{u}^{(*)} = \mathbf{u}^{(*)} - \alpha\Big(\sum_{d=0}^{2N-2} \mathbf{H}_d \mathbf{u}^{(*)} + \mathbf{g}\Big). \tag{3.13}$$

Subtracting (3.12) from (3.13) shows that the errors

$$\varepsilon^{(j)} = \mathbf{u}^{(*)} - \mathbf{u}^{(j)} \tag{3.14}$$

obey the recursion

$$\varepsilon^{(j+1)} = \varepsilon^{(j)} - \alpha\Big(\sum_{d=0}^{2N-2} \mathbf{H}_d \varepsilon^{j-d}\Big). \tag{3.15}$$

If we define the block companion matrix

$$\mathbf{C}(\alpha) = \begin{bmatrix} \mathbf{I} - \alpha\mathbf{H}_0 & -\alpha\mathbf{H}_1 & -\alpha\mathbf{H}_2 & \ldots & -\alpha\mathbf{H}_{2N-3} & -\alpha\mathbf{H}_{2N-2} \\ \mathbf{I} & & & & & \\ & \mathbf{I} & & & & \\ & & \mathbf{I} & & & \\ & & & \ddots & & \\ & & & & \mathbf{I} & \mathbf{0} \end{bmatrix}, \tag{3.16}$$

then the recursion (3.15) for the errors is equivalent to

$$\begin{bmatrix} \varepsilon^{(j+1)} \\ \varepsilon^{(j)} \\ \vdots \\ \varepsilon^{(j-2N+3)} \end{bmatrix} = \mathbf{C}(\alpha) \begin{bmatrix} \varepsilon^{(j)} \\ \varepsilon^{(j-1)} \\ \vdots \\ \varepsilon^{(j-2N+2)} \end{bmatrix}. \tag{3.17}$$

Thus, convergence of our parallel-in-time gradient method is guaranteed if the spectral radius of the block companion matrix (3.16) is less than one. In the remainder of this section we will show that this is true for sufficiently small step size.

Theorem 3.1 *For sufficiently small step size $\alpha > 0$, the matrix $\mathbf{C}(\alpha)$ defined in* (3.16) *has spectral radius less than one.*

The rest of this section is devoted to the proof of this convergence theorem. Specifically, Theorem 3.1 is a special case of Theorem 3.3 below.

In our convergence proof the positive definiteness of the Hessian is important, but not the particular structure of the matrices $\mathbf{H}_d$ in the splitting (3.11). Therefore, given complex $m \times m$ matrices $\mathbf{M}_0, \mathbf{M}_1, ..., \mathbf{M}_n$ with $\sum_{i=0}^{n} \mathbf{M}_i$ Hermitian and positive definite, we consider the block companion matrix

$$\widetilde{\mathbf{C}}(\alpha) \stackrel{\text{def}}{=} \begin{bmatrix} \mathbf{I} - \alpha\mathbf{M}_0 & -\alpha\mathbf{M}_1 & -\alpha\mathbf{M}_2 & \dots & -\alpha\mathbf{M}_{n-1} & -\alpha\mathbf{M}_n \\ \mathbf{I} & & & & & \\ & \mathbf{I} & & & & \\ & & \mathbf{I} & & & \\ & & & \ddots & & \\ & & & & \mathbf{I} & \mathbf{0} \end{bmatrix}. \tag{3.18}$$

We will prove that the spectral radius of $\widetilde{\mathbf{C}}(\alpha)$ is strictly less than one for sufficient small step sizes $\alpha > 0$.

Our proof is based on an analysis of the location of the roots of the characteristic polynomial of $\widetilde{\mathbf{C}}(\alpha)$,

$$Q(\alpha, \lambda) \stackrel{\text{def}}{=} \det(\widetilde{\mathbf{C}}(\alpha) - \lambda\mathbf{I}), \tag{3.19}$$

for small $\alpha > 0$. We also define

$$P(\alpha, \lambda) \stackrel{\text{def}}{=} \lambda^{n+1}\mathbf{I} + \lambda^n(\alpha\mathbf{M}_0 - \mathbf{I}) + \alpha \sum_{i=1}^{n} \lambda^{n-i}\mathbf{M}_i. \tag{3.20}$$

Note that $Q(\alpha, \lambda) = (-1)^{mn} \det(P(\alpha, \lambda))$ (see, e.g., [8, p.17]). The following three statements are equivalent

i. λ_α is an eigenvalue of $\widetilde{\mathbf{C}}(\alpha)$.

ii. λ_α is an latent root of $P(\alpha,\cdot)$, i.e. $P(\alpha,\lambda_\alpha)$ is singular.

iii. λ_α is a root of $Q(\alpha,\cdot)$.

Since $P(0,\lambda) = \lambda^n(\lambda-1)\mathbf{I}$ and $\det(P(0,\lambda)) = \lambda^{mn}(\lambda-1)^m$, the companion matrix $\widetilde{\mathbf{C}}(0)$ only has eigenvalues 0 and 1. By continuity of polynomial roots with respect to polynomial coefficients [17], the roots λ_α of $Q(\alpha,\cdot)$ are contained in small balls around 0 and around 1 for sufficiently small $\alpha > 0$. See Figure 3.2. For the roots λ_α in the small ball around 1, we can actually show that they must be contained in the open cone C_k defined in (3.21) below, and that they have magnitude less than one. See Figure 3.2 and Lemma 3.2 below. This implies that the spectral radius of $\widetilde{\mathbf{C}}(\alpha)$ is less than one for sufficiently small $\alpha > 0$ (see Theorem 3.3 below).

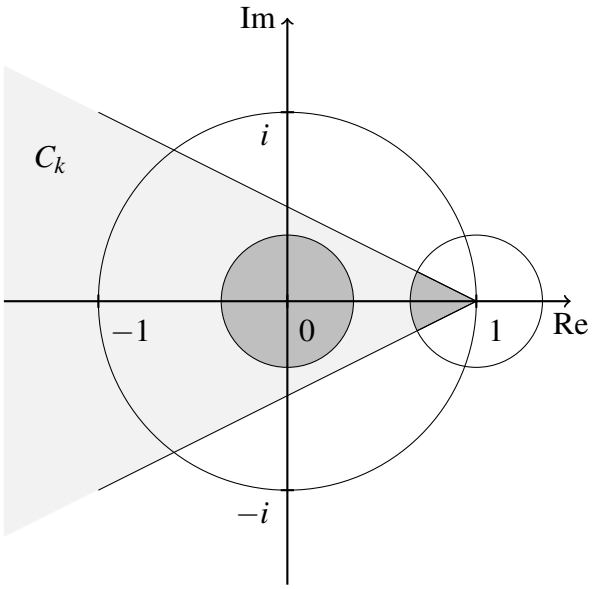

Figure 3.2: For sufficiently small $\alpha > 0$ the eigenvalues λ_α of the companion matrix $\widetilde{\mathbf{C}}(\alpha)$, defined in (3.18) with $\sum_{i=0}^n \mathbf{M}_i$ Hermitian and positive definite, lie in the union of a small open ball around 0 and of the intersection of a small open ball around 1 and the open cone C_k, indicated by the dark shaded regions. In particular, all eigenvalues are inside the unit disk.

In the following $B(c,r)$ denotes the open ball in the complex plane of radius r centered at c. The real and imganizary parts of a complex number z are denoted by $\mathrm{Re}(z)$ and $\mathrm{Im}(z)$, respectively.

Lemma 3.2 *If $\mathbf{M}_0, \mathbf{M}_1, ..., \mathbf{M}_n$ are $m \times m$ complex matrices with $\sum_{i=0}^n \mathbf{M}_i$ Hermitian and positive definite, then the following two statement are valid.*

i. For any δ_2 there exists a $\delta_1 > 0$ such that for all $\alpha \in (0,\delta_1) \subset \mathbb{R}$ all latent roots of $P(\alpha,\lambda)$ are contained in $B(0,\delta_2) \cup B(1,\delta_2) \subset \mathbb{C}$.

ii. Let $\delta_2 \in (0,1/2)$ *and* δ_1 *be given as in part i. For all* $k>0$ *there exists* $\delta_3 \in (0,\delta_1)$ *such that for all* $\alpha \in (0,\delta_3) \subset \mathbb{R}$ *the latent roots* $\lambda \in B(1,\delta_2)$ *of* $P(\alpha,\lambda)$ *satisfy*

$$\lambda \in C_k \stackrel{\text{def}}{=} \{z \in \mathbb{C} : \ \mathrm{Re}(z) < 1 \text{ and } |\mathrm{Im}(z)|/(1-\mathrm{Re}(z)) < k\}. \tag{3.21}$$

Proof: i. The first statement is a direct consequence of the fact that the roots of $\det(P(0,\lambda)) = \det(\lambda^n(\lambda-1)\mathbf{I}) = \lambda^{mn}(\lambda-1)^m$ are $\lambda=0$ and $\lambda=1$ and of the continuity of polynomial roots with respect to polynomial coefficients. (See, e.g., [17].)

ii. We first prove the second part of the lemma for the case of $n=0$. In this case $P(\alpha,\lambda) = \lambda\mathbf{I} - (\mathbf{I}-\alpha\mathbf{M}_0)$ and $\mathbf{M}_0$ is Hermitian and positive definite. All latent roots of $P(\alpha,\lambda) = \lambda\mathbf{I} - (\mathbf{I}-\alpha\mathbf{M}_0)$ are given by $\lambda_\alpha = 1-\alpha\sigma$, where $\sigma>0$ is an eigenvalue of $\mathbf{M}_0$. Let $\sigma_{\max}$ denote the largest eigenvalue of $\mathbf{M}_0$. For $\alpha \in (0,\delta_3)$, $\delta_3 < \min\{2/\sigma_{\max}(\mathbf{M}_0),\delta_1\}$, the latent roots of $P(\alpha,\lambda)$ are contained in $(-1,1) \subset C_k$ for any $k>0$.

Now let $n \geq 1$. We can write

$$\begin{aligned} P(\alpha,\lambda) &= \lambda^{n+1}\mathbf{I} + \lambda^n(\alpha\mathbf{M}_0 - \mathbf{I}) + \alpha\sum_{i=1}^{n}\lambda^{n-i}\mathbf{M}_i \\ &= \alpha\sum_{i=0}^{n}\mathbf{M}_i + (\lambda-1)\mathbf{I} + \Big[(\lambda^n-1)(\lambda-1)\mathbf{I} + \alpha\sum_{i=0}^{n-1}(\lambda^{n-i}-1)\mathbf{M}_i\Big]. \end{aligned} \tag{3.22}$$

The last term in (3.22) can be estimated using

$$\begin{aligned} &\big\|(\lambda^n-1)(\lambda-1)\mathbf{I} + \alpha\sum_{i=0}^{n-1}(\lambda^{n-i}-1)\mathbf{M}_i\big\|_2 \\ &= \big\|(\lambda-1)^2\sum_{i=0}^{n-1}\lambda^i\mathbf{I} + \alpha(\lambda-1)\sum_{i=0}^{n-1}\Big(\sum_{j=0}^{n-i-1}\lambda^j\Big)\mathbf{M}_i\big\|_2 \\ &\leq |\lambda-1|\Big[|\lambda-1|\sum_{i=0}^{n-1}|\lambda|^i + \alpha\sum_{i=0}^{n-1}\Big(\sum_{j=0}^{n-i-1}|\lambda|^j\Big)\|\mathbf{M}_i\|_2\Big]. \end{aligned} \tag{3.23}$$

There exist $\varepsilon_1,\varepsilon_2>0$ such that for all $\alpha \in (0,\varepsilon_1)$, $\lambda \in B(1,\varepsilon_2)$,

$$|\lambda-1|\sum_{i=0}^{n-1}|\lambda|^i + \alpha\sum_{i=0}^{n-1}\Big(\sum_{j=0}^{n-i-1}|\lambda|^j\Big)\|\mathbf{M}_i\|_2 < \frac{1}{\sqrt{2}}$$

and, hence

$$\big\|(\lambda^n-1)(\lambda-1)\mathbf{I} + \alpha\sum_{i=0}^{n}(\lambda^{n-i}-1)\mathbf{M}_i\big\|_2 \leq \frac{1}{\sqrt{2}}|\lambda-1|. \tag{3.24}$$

Let $\sigma_{\min}(\cdot)$ denote the minimum singular value of a matrix and recall that $\sum_{i=0}^{n}\mathbf{M}_i$ is Hermitian and positive definite. For $\alpha > 0$ and $\lambda \in \mathbb{C}$ the first two terms in (3.22) can be estimated using

$$\begin{aligned}\sigma_{\min}\Big(\alpha\sum_{i=0}^{n}\mathbf{M}_i + (\lambda-1)\mathbf{I}\Big) &= \Big|\alpha\sigma_{\min}(\sum_{i=0}^{n}\mathbf{M}_i) + (\lambda-1)\Big| \\ &= \Big[\Big(\alpha\sigma_{\min}(\sum_{i=0}^{n}\mathbf{M}_i) + \mathrm{Re}(\lambda-1)\Big)^2 + \mathrm{Im}(\lambda-1)^2\Big]^{1/2}. \qquad (3.25a)\end{aligned}$$

Furthermore, if $\mathrm{Re}(\lambda) \geq 1$, then

$$\begin{aligned}\sigma_{\min}\Big(\alpha\sum_{i=0}^{n}\mathbf{M}_i + (\lambda-1)\mathbf{I}\Big) &= \Big[\Big(\alpha\sigma_{\min}(\sum_{i=0}^{n}\mathbf{M}_i) + \mathrm{Re}(\lambda-1)\Big)^2 + \mathrm{Im}(\lambda-1)^2\Big]^{1/2} \\ &\geq \frac{1}{\sqrt{2}}\Big|\alpha\sigma_{\min}(\sum_{i=0}^{n}\mathbf{M}_i) + \mathrm{Re}(\lambda-1)\Big| + \frac{1}{\sqrt{2}}\Big|\,\mathrm{Im}(\lambda-1)\Big| \\ &= \frac{1}{\sqrt{2}}\alpha\sigma_{\min}(\sum_{i=0}^{n}\mathbf{M}_i) + \frac{1}{\sqrt{2}}\,\mathrm{Re}(\lambda-1) + \frac{1}{\sqrt{2}}\Big|\,\mathrm{Im}(\lambda-1)\Big| \\ &\geq \frac{1}{\sqrt{2}}\alpha\sigma_{\min}(\sum_{i=0}^{n}\mathbf{M}_i) + \frac{1}{\sqrt{2}}|\lambda-1|. \qquad (3.25b)\end{aligned}$$

Combining (3.24) and (3.25b) shows that for all $\alpha \in (0,\varepsilon_1)$ and all $\lambda \in B(1,\varepsilon_2)$ with $\mathrm{Re}(\lambda) \geq 1$,

$$\sigma_{\min}(\alpha\sum_{i=0}^{n}\mathbf{M}_i + (\lambda-1)\mathbf{I}) > \|(\lambda^n-1)(\lambda-1)\mathbf{I} + \alpha\sum_{i=0}^{n}(\lambda^{n-i}-1)\mathbf{M}_i\|_2 \qquad (3.26)$$

and, consequently, that $P(\alpha,\lambda)$ is non-singular. Therefore, for all $\alpha \in (0,\varepsilon_1)$ all latent roots of $P(\alpha,\lambda)$ that are contained in $B(1,\varepsilon_2)$ satisfy $\mathrm{Re}(\lambda_\alpha) < 1$.

Given an arbitrary $k > 0$. For λ with $\mathrm{Re}(\lambda) < 1$ and $\lambda \notin C_k$, $|\,\mathrm{Im}(\lambda-1)| = |\,\mathrm{Im}(\lambda)| \geq k(1-\mathrm{Re}(\lambda)) = k\,\mathrm{Re}(1-\lambda) = k|\,\mathrm{Re}(\lambda-1)|$ and, thus,

$$\frac{k+1}{k}|\,\mathrm{Im}(\lambda-1)| \geq |\,\mathrm{Im}(\lambda-1)| + |\,\mathrm{Re}(\lambda-1)| \geq |\lambda-1|.$$

Combining this with (3.25a) we obtain the estimate

$$\sigma_{\min}(\alpha\sum_{i=0}^{n}\mathbf{M}_i + (\lambda-1)\mathbf{I}) \geq |\,\mathrm{Im}(\lambda-1)| \geq \frac{k}{k+1}|\lambda-1|. \qquad (3.27)$$

There exist $\varepsilon_{1,k},\varepsilon_{2,k} > 0$ such that for all $\alpha \in (0,\varepsilon_{1,k})$, $\lambda \in B(1,\varepsilon_{2,k})$,

$$|\lambda-1|\sum_{i=0}^{n-1}|\lambda|^i + \alpha\sum_{i=0}^{n-1}(\sum_{j=0}^{n-i-1}|\lambda|^{(j)})\|\mathbf{M}_i\|_2 < \frac{k}{k+1}.$$

Combining this inequality with (3.23) and (3.27) shows that for all $\alpha \in (0, \varepsilon_{1,k})$ and all $\lambda \in B(1, \varepsilon_{2,k})$ with $\mathrm{Re}(\lambda) < 0$ and $\lambda \notin C_k$ the inequality (3.26) holds and, thus, that $P(\alpha, \lambda)$ is non-singular. Therefore, for all $\alpha \in (0, \min\{\varepsilon_1, \varepsilon_{1,k}\})$ all latent roots of $P(\alpha, \lambda)$ that are contained in $B(1, \min\{\varepsilon_2, \varepsilon_{2,k}\})$ satisfy $\mathrm{Re}(\lambda_\alpha) < 1$ and $\lambda_\alpha \in C_k$.

Given $k > 0$, choose $\delta_3 \leq \min\{\varepsilon_1, \varepsilon_{1,k}\}$ such that for all $\alpha \in (0, \delta_3)$ the latent roots of $P(\alpha, \cdot)$ are in $B(0, \delta_2) \cup B(1, \min\{\delta_2, \varepsilon_2, \varepsilon_{2,k}\}) \subset \mathbb{C}$. This is possible by continuity of polynomial roots with respect to polynomial coefficients. From the previous steps it follows that all latent roots $\lambda_\alpha \in B(1, \delta_2)$ of $P(\alpha, \cdot)$ are in fact contained in $B(1, \min\{\delta_2, \varepsilon_2, \varepsilon_{2,k}\})$ and satisfy $\lambda_\alpha \in C_k$. □

Theorem 3.3 *If $\mathbf{M}_0, \mathbf{M}_1, ..., \mathbf{M}_n$ are $m \times m$ complex matrices with $\sum_{i=0}^{n} \mathbf{M}_i$ Hermitian and positive definite, then the block companion matrix* (3.18) *has spectral radius strictly less than* 1 *for sufficiently small real $\alpha > 0$.*

Proof: Let $\delta_2 \in (0, 1/2)$ and $k > 0$ arbitrary. Lemma 3.2 guarantees the existence of $\delta_1 > 0$ and $\delta_3 \in (0, \delta_1)$ such that for all $\alpha \in (0, \delta_3)$ the latent roots of $P(\alpha, \cdot)$ are contained in $B(0, \delta_2) \cup \big(B(1, \delta_2) \cap C_k\big) \subset B(0, 1)$. Therefore, for all $\alpha \in (0, \delta_3)$, the spectral radius of $\widetilde{\mathbf{C}}(\alpha)$ is strictly less than 1. □

Because $\sum_{i=0}^{n} \mathbf{H}_i = \mathbf{H}$ by (3.11) and $\mathbf{H}$ is symmetric positive definite by (2.8), Theorem 3.1 is a special case of Theorem 3.3.

4 Numerical Example

4.1 Optimal Dirichlet Boundary Control of a 1D Advection Diffusion Equation

Our first example is a Dirichlet boundary control problem governed by a linear advection-diffusion-reaction equation. Given $T > 0$, $\kappa > 0$, $\beta > 0$, $\gamma, v \geq 0$, and functions $\widehat{y} \in L^2(0, T)$, $f \in L^2((0,1) \times (0, T))$, the optimal control problem is

$$\text{minimize } \frac{1}{2} \int_0^T (y(1,t) - \widehat{y}(t))^2 dt + \frac{\beta}{2} \int_0^T u^2(t) dt \tag{4.1a}$$

subject to

$$\frac{\partial y(x,t)}{\partial t} - \kappa \frac{\partial^2 y(x,t)}{\partial x^2} + v \frac{\partial y(x,t)}{\partial x} + \gamma y(x,t) = f(x,t) \qquad x \in (0,1), t \in (0,T), \tag{4.1b}$$

$$\frac{\partial y(1,t)}{\partial x} = 0 \qquad t \in (0,T), \tag{4.1c}$$

$$y(0,t) = u(t) \qquad t \in (0,T), \tag{4.1d}$$

$$y(x,0) = y_0(x) \qquad x \in (0,1). \tag{4.1e}$$

This problem has a unique solution $u \in L^2(0,T)$ and corresponding state $y \in W(0,T)$. See, e.g., [16, Ch.3] or [13] for details.

We will discretize the problem using piecewise linear finite elements in space and backward Euler in time. This leads to a fully discretized problem of the type (2.1). We will present the details below. Theorem 3.1 guarantees convergence of the parallel-in-time gradient-type method for sufficiently small step size α, but unfortunately does not provide bounds on the feasible step sizes. We use this simple example to numerically explore the dependence of step sizes.

The optimal control problem is discretized in space using piecewise linear finite elements. This leads to

$$\text{minimize } \frac{1}{2}\int_0^T (y(t)-\hat{y}(t))^T Q(y(t)-\hat{y}(t))dt + \frac{1}{2}\int_0^T u(t)^T R u(t) dt \tag{4.2a}$$

subject to

$$M\frac{d}{dt}y(t) + Ay(t) = Bu(t) + f(t), \qquad t \in (0,T), \tag{4.2b}$$

$$y(0) = y_{\text{given}}. \tag{4.2c}$$

Here $y(t) \in \mathbb{R}^{n_y}$, n_y is the number of spatial subintervals in $(0,1)$, and $u(t) \in \mathbb{R}^{n_u}$, $n_u = 1$.

To discretize (4.2) in time we use the backward Euler method with time step size $\Delta t = T/K$ and time steps $t_k = \Delta t\, k$, $k = 0, \ldots, K$. This leads to the problem

$$\text{minimize } \frac{\Delta t}{2}\sum_{k=0}^{K-1} (y_{k+1}-\hat{y}(t_{k+1}))^T Q\,(y_{k+1}-\hat{y}(t_{k+1})) + \frac{\Delta t}{2}\sum_{k=0}^{K-1} u_{k+1}^T R\, u_{k+1} \tag{4.3a}$$

subject to

$$(M+\Delta t A)y_{k+1} = \Delta t B u_{k+1} + \Delta t M y_k + \Delta t f(t_{k+1}), \qquad k = 0, \ldots, K-1, \tag{4.3b}$$

$$y_0 = y_{\text{given}}. \tag{4.3c}$$

The fully discretized problem (4.3) is of the type (2.1) if we set

$$A_k = \Delta t(M+\Delta t A)^{-1}M, \qquad B_k = \Delta t(M+\Delta t A)^{-1}B, \qquad c_k = \Delta t(M+\Delta t A)^{-1} f(t_{k+1}),$$
$$Q_k = \Delta t Q, \qquad R_k = \Delta t R, \qquad d_k = -\Delta t Q\hat{y}(t_k), \quad e_k = 0,$$

and perform an index shift $u_{k+1} \to u_k$. The matrices $(M+\Delta tA)^{-1}M$ and $(M+\Delta tA)^{-1}B$ are never formed explicitly, but linear systems with matrix $(M+\Delta tA)$ are solved whenever matrix-vector products of the type $(M+\Delta tA)^{-1}Mv$ or $(M+\Delta tA)^{-1}Bw$ have to be computed.

On our computations we use $T=10$, $\beta=0.1$, $\kappa=0.001$, $v=5$, $\gamma=1$, $f(x,t)=0$, $y_0(x)=0$, and $\widehat{y}(t)=\sin(t^2)$. Furthermore, our discretized uses $n_y=128$ spatial subintervals in $(0,1)$, and $K=1000$ time steps.

The speed of convergence of the parallel gradient-type method depends the step size α. To estimate the optimal step size for a given number N of time subintervals, we conduct the following experiment. Given a number of time subintervals N we subdivide $0,\ldots,K$ into approximately equally sized groups. In the case where N does not divide the number of time steps K, we split the time steps so that $0 \le (K_i - K_{i-1}) - (K_j - K_{j-1}) \le 1$ for all $1 \le i < j \le N$. Then for a number of equally distributed time steps $\alpha \in (0.3,5)$ we run the parallel gradient-type algorithm started with zero initial guess until the error between computed control $\mathbf{u}^{(j)}$ and exact control $\mathbf{u}^{(*)}$, computed solving (4.3) with high accuracy, is less 10^{-6}, or a maximum number of iterations is exceeded. For $N \in \{1,...,100\}$ we report the step size α for the parallel gradient-type algorithm requires the fewest iterations. These step size α are shown in Figure 4.1. We note that the optimal step sizes depend on other problem data as well and, thus, step sizes that are good for the 1D example problem (4.1) may not be good for other problems, such as (4.4) below.

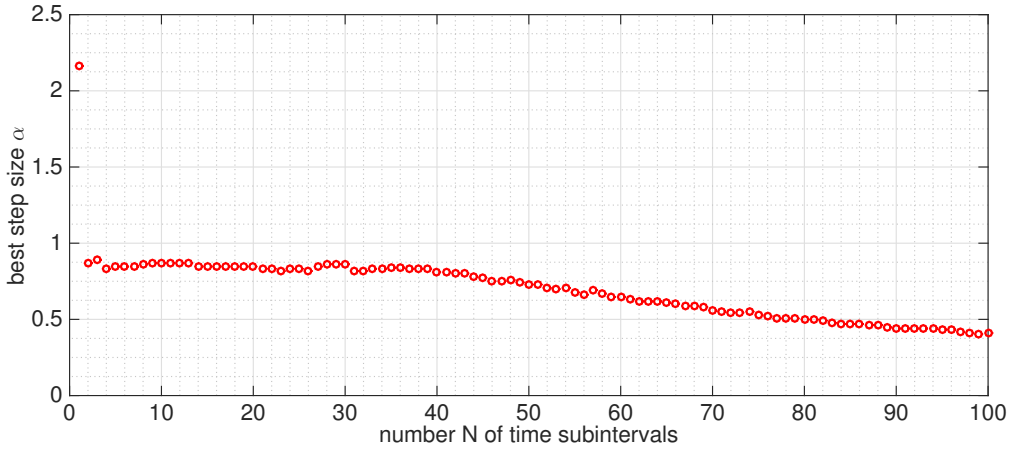

Figure 4.1: Optimal step-sizes α for varying number N of time subdomains for the 1D example problem.

4.2 Optimal Control of a 3D Advection Diffusion Equation

We consider an optimal control problem governed by an advection diffusion reaction PDE posed on the spatial domain $\Omega \subset \mathbb{R}^3$ and time domain$(0,T)$. Let $\Omega_o \subset \Omega$ be the observation region and

let $\Omega_c \subset \Omega$ be the domain on which the control is applied. Let χ_{Ω_c} denote the indicator function of the control domain Ω_c. Given scalars $\kappa > 0$, $\beta > 0$, $\gamma \geq 0$, the advection $v \in \mathbb{R}^3$, and functions $\widehat{y} \in L^2(\Omega_o \times (0,T))$, $f \in L^2(\Omega \times (0,T))$, the optimal control problem is

$$\text{minimize } \frac{1}{2}\int_0^T \int_{\Omega_o} (y(x,t) - \widehat{y}(x,t))^2 dxdt + \frac{\beta}{2}\int_0^T \int_{\Omega_c} u^2(x,t) dxdt \tag{4.4a}$$

subject to

$$\frac{\partial y(x,t)}{\partial t} - \kappa \Delta y(x,t) + v \cdot \nabla y(x,t) + \gamma y(x,t) = f(x,t) + \chi_{\Omega_c}(x) u(x,t), \quad x \in \Omega, t \in (0,T), \tag{4.4b}$$

$$\nabla y(x,t) \cdot n = 0, \quad x \in \partial\Omega, t \in (0,T), \tag{4.4c}$$

$$y(x,0) = y_{\text{given}}(x), \quad x \in \Omega. \tag{4.4d}$$

This problem has a unique solution $u \in L^2(\Omega_c \times (0,T))$ and corresponding state $y \in W(0,T)$. See, e.g., [16, Ch.3] or [13] for details.

We consider $\Omega = (0,1)^3$. The optimal control problem is discretized in space using a standard cell-centered finite volume method with hexahedral cells of size $(1/n_1) \times (1/n_2) \times (1/n_3)$ [9, Sec. 3.3]. We assume that the observation region Ω_o is a hexahedron given as the union of ℓ cells and that the control region Ω_c is a union of hexahedra given as the union of n_u cells. This leads to the semidiscretized problem (4.2) where $y(t) \in \mathbb{R}^n$, $n_y = n_1 n_2 n_3$, and $u(t) \in \mathbb{R}^{n_u}$.

To discretize (4.2) in time we use the backward Euler method with time step size $\Delta t = T/K$ and time steps $t_k = \Delta t\, k$, $k = 0, \ldots, K$. This leads to the fully discretized problem (4.3), which is of the type (2.1).

Our implementation uses Epetra linear algebra libraries, AztecOO linear solvers, and ML multigrid preconditioning packages from the Trilinos Project [11] to solve the linear systems with matrices $M + \Delta t A$ and $(M + \Delta t A)^T$. Computations are performed on the Rice University DAVinCI cluster. [1]

For our numerical example we choose the control region $\Omega_c = (0.1, 0.3) \times (0.2, 0.8) \times (0.2, 0.8) \cup (0.7, 0.9) \times (0.2, 0.8) \times (0.2, 0.8)$, the observation region $\Omega_o = (0.4, 0.6) \times (0,1) \times (0,1)$, and the final time $T = 8$ or $T = 16$. In the objective (4.4a) we have $\hat{y}(x,t) = 10[(x_2 - 0.5)\cos(2\pi t) + (x_3 - 0.5)\sin(2\pi t)]^3$ and $\beta = 0.001$. In the PDE (4.4b), $\kappa = 0.1$, $v = [1,1,1]^T$, $\gamma = 0.01$, and $y_{\text{given}}(x) = 0$. Our discretization we use $K = 200$ (if $T = 8$) or $K = 400$ (if $T = 16$) time steps and a spatial discretization $n_1 = n_2 = n_3 = 10$.

So far, there is no good a-priori way of determining a good step size for the gradient type method, and adaptive steps size selection is part of future research. Currently, we select the the step size by examining the iteration for a few different step sizes and then selecting our α from these trials. For the case $T = 8$, $K = 200$, this resulted in the step size $\alpha = 2400$ for $N = 1, 4, 6, 8, 12, 16, 20$ time subintervals, and $\alpha = 2060, 1860, 1500, 1000$ for $N = 25, 30, 40, 60$.

[1] This work was supported in part by the Data Analysis and Visualization Cyberinfrastructure funded by NSF under grant OCI-0959097 and Rice University.

For the case $T = 16$, $K = 400$, this resulted in the step size $\alpha = 950$ for $N = 1, 30, 40, 50$ time subintervals, and $\alpha = 900, 800, 650$ for $N = 60, 70, 80$.

Figure 4.2 shows the speeds-ups. For each of the two cases ($T = 8$, $K = 200$ and $T = 16$, $K = 400$) we include two speeds-up curves. One shows speed-up measured by number of iterations of gradient method divided by the number of iterations of the parallel-in-time gradient method for various N over the number of cores N. Thus, if $I(N)$ is the number of iterations needed by parallel-in-time method with N subintervals and cores, one speed-up curve shows $I(1)/(I(N)/N)$. Although this ignores communication cost it is a good indicator of actual speed-up, which is shown in the other curve. The actual speed-up is measured by run time of the gradient method divided by the run time of the parallel-in-time gradient method for various N. That is, if $t(N)$ is the run time required by parallel-in-time method with N subintervals and cores, the actual speed-up is $t(1)/t(N)$. The number N of time-subintervals changes the parallel-in-time gradient-type method and its convergence rate. If the number of time-subintervals is too large, convergence deteriorates and negatively impact speed-up obtained by parallelizing the work performed in each iteration of the parallel-in-time gradient-type method. For our problem settings we observe excellent speed-ups for up to $N \approx 20$ and $N \approx 50$ time subdomains. It is also important to note that the speed-up due to time decomposition multiplies existing speed-up in the solution of the systems with matrices $M + \Delta t A$ or $(M + \Delta t A)^T$ that arise in the solutions of sub-time interval state and adjoint equations. Our spatial discretization is small so that no such parallelization in the time-stepping was useful.

5 Conclusions

We have introduced a new parallel-in-time gradient-type method for convex linear-quadratic DTOC problems and proved its convergence for sufficiently small step-size. Since our new method resembles the classical gradient method, a given implementation of the classical gradient method can be easily modified to implement our parallel-in-time gradient-type method. We observed nearly perfect speed-up for modest numbers time subdomains. The speed-up due to time decomposition multiplies existing speed-up in the solution of state and adjoint equations, e.g, due to parallel solution of linear systems arising in the time-stepping.

The convergence theory presented in this paper uses the structure of linear-quadratic DTOC problems, but the formulation of our parallel-in-time gradient-type method can be extended to optimal control problems governed by ordinary or partial differential equations. Moreover, it can conceptually be extended to to general nonlinear problems. Applications and analyses of these extensions are part of ongoing research.

Although the gradient method can overall be effective for large-scale optimal control problems, other methods, such as the conjugate-gradient method for convex linear-quadratic problems and Newton-type methods for general nonlinear problems, exhibit superior convergence properties. Therefore, parallel-in-time extensions of those methods will be useful.

We prove that the spectral radius of the block companion matrix, the iteration matrix cor-

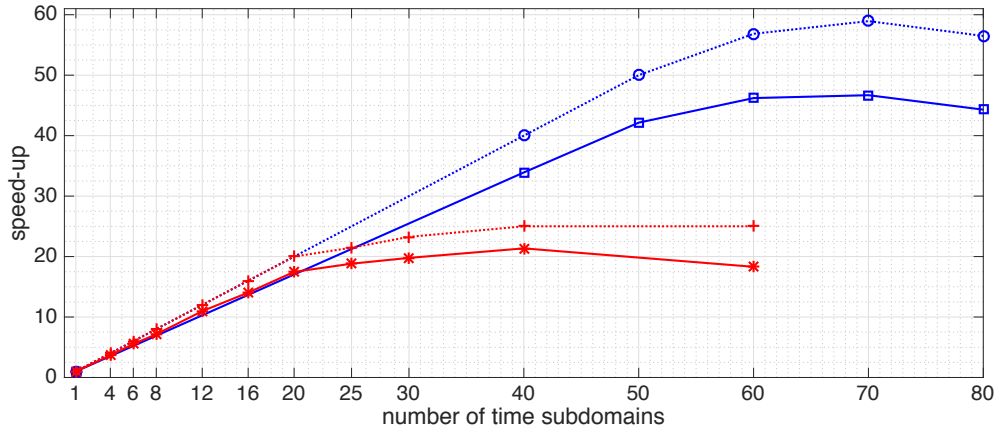

Figure 4.2: Speed-ups the parallel-in-time gradient method for two examples problems with end time $T = 8$ and $K = 200$ time steps (red curves), and with end time $T = 16$ and $K = 400$ time steps (blue curves). For each case two speed-up curves are shown. If $I(N)$ denotes the number of iterations needed by parallel-in-time method with N subintervals and cores, the speed-up curves indicated by '$\cdots\circ$' and '$\cdots+$' show $I(1)/(I(N)/N)$. If $t(N)$ denotes the run time needed by parallel-in-time method with N subintervals and cores, the curves indicated by '$-\square$' and '$-*$' show $t(1)/t(N)$. Excellent speed-ups are obtained for up to $N = 20$ time domains when $T = 8$ and $K = 200$ and for up to $N = 50$ time domains when $T = 16$ and $K = 400$.

responding to our parallel-in-time gradient-type method, is strictly less than one for sufficiently small step sizes and, therefore that our parallel-in-time gradient-type method converges. It would be useful to have information about how small the step size has to be, relative to problem data, and how small the spectral radius can become.

References

[1] A. T. BARKER AND M. STOLL, *Domain decomposition in time for PDE-constrained optimization*, Comput. Phys. Commun., 197 (2015), pp. 136–143.

[2] M. BERGGREN AND M. HEINKENSCHLOSS, *Parallel solution of optimal-control problems by time-domain decomposition*, in Computational Science for the 21st Century, M.-O. Bristeau, G. Etgen, W. Fitzgibbon, J. L. Lions, J. Periaux, and M. F. Wheeler, eds., Chichester, 1997, J. Wiley, pp. 102–112.

[3] D. P. BERTSEKAS, *Nonlinear Programming*, Athena Scientific, Belmont, Massachusetts, second ed., 1999.

[4] T. CARRARO, M. GEIGER, AND R. RANNACHER, *Indirect multiple shooting for nonlinear parabolic optimal control problems with control constraints*, SIAM J. Sci. Comput., 36 (2014), pp. A452–A481.

[5] A. COMAS, *Time-Domain Decomposition Preconditioners for the Solution of Discretized Parabolic Optimal Control Problem*, PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2005. Available as CAAM TR06–01.

[6] J. DE PILLIS, *k-part splittings and operator parameter overrelaxation*, J. Math. Anal. Appl., 53 (1976), pp. 313–342.

[7] J. DE PILLIS AND M. NEUMANN, *Iterative methods with k-part splittings*, IMA J. Numer. Anal., 1 (1981), pp. 65–79.

[8] J. E. DENNIS JR., J. F. TRAUB, AND R. P. WEBER, *On the matrix polynomial, lamda-matrix and block eigenvalue problems*, Tech. Rep. CMU-CS-71-110, Computer Science Department, Carnegie Mellon University, 1971.

[9] R. EYMARD, T. GALLOUËT, AND R. HERBIN, *Finite volume methods*, in Handbook of numerical analysis, Vol. VII, P. G. Ciarlet and J. L. Lions, eds., Handb. Numer. Anal., VII, North-Holland, Amsterdam, 2000, pp. 713–1020.

[10] 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.

[11] M. A. HEROUX, R. A. BARTLETT, V. E. HOWLE, R. J. HOEKSTRA, J. J. HU, T. G. KOLDA, R. B. LEHOUCQ, K. R. LONG, R. P. PAWLOWSKI, E. T. PHIPPS, A. G. SALINGER, H. K. THORNQUIST, R. S. TUMINARO, J. M. WILLENBRING, A. WILLIAMS, AND K. S. STANLEY, *An overview of the trilinos project*, ACM Trans. Math. Softw., 31 (2005), pp. 397–423.

[12] H. K. HESSE AND G. KANSCHAT, *Mesh adaptive multiple shooting for partial differential equations. I. Linear quadratic optimal control problems*, J. Numer. Math., 17 (2009), pp. 195–217.

[13] J.-L. LIONS, *Optimal Control of Systems Governed by Partial Differential Equations*, Springer Verlag, Berlin, Heidelberg, New York, 1971.

[14] E. POLAK, *Computational Methods in Optimization. A Unified Approach*, Academic Press, New York, London, Paris, San Diego, San Francisco, 1971.

[15] V. RAO AND A. SANDU, *A time-parallel approach to strong-constraint four-dimensional variational data assimilation*, arXiv:1505.04515v1, (2015).

[16] F. TRÖLTZSCH, *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*, vol. 112 of Graduate Studies in Mathematics, American Mathematical Society, Providence, RI, 2010.

[17] D. J. UHERKA AND A. M. SERGOTT, *On the continuous dependence of the roots of a polynomial on its coefficients*, Amer. Math. Monthly, 84 (1977), pp. 368–370.

[18] S. ULBRICH, *Preconditioners based on 'parareal' time-domain decomposition for time-dependent PDE-constrained optimization*, in Multiple Shooting and Time Domain Decomposition Methods. MuS-TDD, Heidelberg, May 6-8, 2013, T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, eds., vol. 9 of Contributions in Mathematical and Computational Sciences, Springer-Verlag, Heidelberg, 2015, pp. 203–232.