PARAOPT: A PARAREAL ALGORITHM FOR OPTIMALITY SYSTEMS

MARTIN J. GANDER\(^\dagger\), FELIX KWOK\(\ddagger\), AND JULIEN SALOMON\(\S\)

Abstract. The time parallel solution of optimality systems arising in PDE constrained optimization could be achieved by simply applying any time parallel algorithm, such as Parareal, to solve the forward and backward evolution problems arising in the optimization loop. We propose here a different strategy by devising directly a new time parallel algorithm, which we call ParaOpt, for the coupled forward and backward nonlinear partial differential equations. ParaOpt is inspired by the Parareal algorithm for evolution equations and thus is automatically a two-level method. We provide a detailed convergence analysis for the case of linear parabolic PDE constraints. We illustrate the performance of ParaOpt with numerical experiments for both linear and nonlinear optimality systems.

Key words. Parareal algorithm, optimal control, preconditioning

AMS subject classifications. 49M27, 68W10, 65K10, 65F08, 93B40

DOI. 10.1137/19M1292291

1. Introduction. Time parallel time integration has become an active research area over the last decade; there is even an annual workshop now dedicated to this topic called the PinT (Parallel in Time) workshop, which started with the first such dedicated workshop at the USI in Lugano in June 2011. The main reason for this interest is the advent of massively parallel computers [5] with so many computing cores that spatial parallelization of an evolution problem saturates long before all cores have been effectively used. There are four classes of such algorithms: methods based on multiple shooting leading to the Parareal algorithm [45, 2, 30, 33, 20, 11, 18, 40], methods based on waveform relaxation [32, 8, 19, 21, 13, 14, 31, 39, 1, 15], methods based on multigrid [27, 34, 53, 28, 6, 17, 7, 41, 4], and direct time parallel methods [42, 49, 50, 35, 10]; for a review of the development of PinT methods, see [9, 46] and the references therein.

A natural area where this type of parallelization could be used effectively is in PDE constrained optimization on bounded time intervals, when the constraint is a time-dependent PDE. In these problems, calculating the descent direction within the optimization loop requires solving both a forward and a backward evolution problem, so one could directly apply time parallelization techniques to each of these solves [23, 24, 25, 26]. Parareal can also be useful in one-shot methods where the preconditioning operator requires the solution of initial value problems; see, e.g., [52]. Another method, which has been proposed in [36, 48] in the context of quantum control,
consists of decomposing the time interval into subintervals and defining intermediate states at subinterval boundaries; this allows one to construct a set of independent optimization problems associated with each subinterval in time. Each iteration of the method then requires the solution of these independent subproblems in parallel, followed by a cheap update of the intermediate states. In this paper, we propose yet another approach based on a fundamental understanding of the Parareal algorithm invented in [33] as a specific approximation of a multiple shooting method [20]. We construct a new time parallel method called ParaOpt for solving directly the coupled forward and backward evolution problems arising in the optimal control context. Our approach is related to the multiple shooting paradigm [43], where the time horizon is decomposed into non-overlapping subintervals, and we solve for the unknown interface state and adjoint variables using an inexact Newton method so that the trajectories are continuous across subintervals. Additionally, a Parareal-like approximation is used to obtain a cheap approximate Jacobian for the Newton solve. There are two potential benefits to our approach: first, it is known that for some control problems, long time horizons lead to difficulties in convergence for the optimization loop. Therefore, a multiple shooting approach allows us to deal with local subproblems on shorter time horizons, where we obtain faster convergence. Such convergence enhancement has also been observed in [3, 37, 38] and also more recently in [48]. Second, if we use Parareal to parallelize the forward and backward sweeps, then the speedup ratio will be bounded above by \( \frac{L}{K} \), where \( L \) is the number of subintervals and \( K \) is the number of Parareal iterations required for convergence. For many problems, especially the nondiffusive ones like the Lotka–Volterra problem we consider in section 4.2, this ratio does not go above 4–5; this limits the potential speedup that can be obtained from this classical approach. By decomposing the control problem directly and conserving the globally coupled structure of the problem, we obtain higher speedup ratios, closer to ones that are achievable for two-level methods for elliptic problems.

Our paper is organized as follows. In section 2, we present our PDE constrained optimization model problem and present ParaOpt for its solution. In section 3 we give a complete convergence analysis of ParaOpt for the case when the PDE constraint is linear and of parabolic type. We then illustrate the performance of ParaOpt by numerical experiments in section 4, for both linear and nonlinear problems. We present our conclusions and an outlook on future work in section 5.


\[
J(c) = \frac{1}{2} \|y(T) - y_{\text{target}}\|^2 + \frac{\alpha}{2} \int_0^T \|c(t)\|^2 dt,
\]

where \( \alpha > 0 \) is a fixed regularization parameter, \( y_{\text{target}} \) is a target state, and the evolution of the state function \( y: [0, T] \to \mathbb{R}^n \) is described by the nonlinear equation

\[
\dot{y}(t) = f(y(t)) + c(t),
\]

with initial condition \( y(0) = y_{\text{init}} \), where \( c(t) \) is the control, which is assumed to enter linearly in the forcing term. The first-order optimality condition then reads

\[
\dot{\lambda} = -\frac{\lambda}{\alpha} (f'(y))^T \lambda,
\]

with the final condition \( \lambda(T) = y(T) - y_{\text{target}} \); see [16] for a detailed derivation.
We now introduce a parallelization algorithm for solving the coupled problem (2.1)--(2.2). The approach we propose follows the ideas of the Parareal algorithm, combining a sequential coarse integration on $[0, T]$ and parallel fine integration on subintervals.

Consider a subdivision of $[0, T] = \cup_{\ell=0}^{L-1} [T_\ell, T_{\ell+1}]$ and two sets of intermediate states $(Y_\ell)_{\ell=0}^L$ and $(\Lambda_\ell)_{\ell=1}^L$ corresponding to approximations of the state $y$ and the adjoint state $\lambda$ at times $T_0, \ldots, T_L$ and $T_1, \ldots, T_L$, respectively.

We denote by $P$ and $Q$ the nonlinear solution operators for the boundary value problem (2.2) on the subinterval $[T_\ell, T_{\ell+1}]$ with initial condition $y(T_\ell) = Y_\ell$ and final condition $\lambda(T_{\ell+1}) = \Lambda_{\ell+1}$, defined so that $P$ propagates the state $y$ forward to $T_{\ell+1}$ and $Q$ propagates the adjoint backward to $T_\ell$:

\[
\begin{pmatrix}
  y(T_{\ell+1}) \\
  \lambda(T_\ell)
\end{pmatrix}
= \begin{pmatrix}
  P(Y_\ell, \Lambda_{\ell+1}) \\
  Q(Y_\ell, \Lambda_{\ell+1})
\end{pmatrix}.
\]

Using these solution operators, we can write the boundary value problem as a system of subproblems, which have to satisfy the matching conditions

\[
\begin{align*}
  Y_0 - y_{\text{init}} &= 0, \\
  Y_1 - P(Y_0, A_1) &= 0, \\
  Y_2 - P(Y_1, A_2) &= 0, \\
  &\vdots \\
  Y_L - P(Y_{L-1}, A_L) &= 0, \\
  \Lambda_1 - Q(Y_1, A_2) &= 0, \\
  \Lambda_2 - Q(Y_2, A_3) &= 0, \\
  &\vdots \\
  \Lambda_L - Y_L + y_{\text{target}} &= 0.
\end{align*}
\]

This nonlinear system of equations can be solved using Newton’s method. Collecting the unknowns in the vector $(Y^T, \Lambda^T) := (Y_0^T, Y_1^T, \ldots, Y_L^T, A_1^T, A_2^T, \ldots, A_L^T)$, we obtain the nonlinear system

\[
\mathcal{F} \left( \begin{array}{c}
  Y \\
  \Lambda
\end{array} \right) := \begin{pmatrix}
  Y_0 - y_{\text{init}} \\
  Y_1 - P(Y_0, A_1) \\
  Y_2 - P(Y_1, A_2) \\
  \vdots \\
  Y_L - P(Y_{L-1}, A_L) \\
  \Lambda_1 - Q(Y_1, A_2) \\
  \Lambda_2 - Q(Y_2, A_3) \\
  \vdots \\
  \Lambda_L - Y_L + y_{\text{target}}
\end{pmatrix} = 0.
\]

Using Newton’s method to solve this system gives the iteration

\[
\mathcal{F}' \left( \begin{array}{c}
  Y^n \\
  \Lambda^n
\end{array} \right) \left( \begin{array}{c}
  Y^{n+1} - Y^n \\
  \Lambda^{n+1} - \Lambda^n
\end{array} \right) = -\mathcal{F} \left( \begin{array}{c}
  Y^n \\
  \Lambda^n
\end{array} \right),
\]

where the Jacobian matrix of $\mathcal{F}$ is given by
These outer iterations is now very expensive, since one must evaluate the propagators 
versa, which is clearly visible in the Jacobian in (2.6). This is in contrast to the initial 
expensive fine grid operations is certainly cheaper than evaluating the derivative on the fine grid; the remaining 
terval problem (2.3), e.g., by using only one time step for the whole subinterval. This 
Note that this system is not triangular: the 
\[ \Lambda = \begin{bmatrix} -P_{y}(Y_{0}, \Lambda_{1}) & 1 \\ \vdots & \ddots & \ddots \\ -Q_{y}(Y_{L-1}, \Lambda_{L}) & 1 & -P_{\lambda}(Y_{L-1}, \Lambda_{L}) \end{bmatrix}. \]

Using the explicit expression for the Jacobian gives us the componentwise linear system we have to solve at each Newton iteration:

\[
Y_{n+1}^{0} = y_{\text{init}},
Y_{1}^{n+1} = -P(Y_{0}^{n}, \Lambda_{1}) + P_{y}(Y_{0}^{n}, \Lambda_{n}^{n})(Y_{0}^{n+1} - Y_{0}^{n}) + P_{\lambda}(Y_{0}^{n}, \Lambda_{n}^{n})(\Lambda_{1}^{n+1} - \Lambda_{1}^{n}),
Y_{2}^{n+1} = -P(Y_{1}^{n}, \Lambda_{2}^{n}) + P_{y}(Y_{1}^{n}, \Lambda_{n}^{n})(Y_{1}^{n+1} - Y_{1}^{n}) + P_{\lambda}(Y_{1}^{n}, \Lambda_{n}^{n})(\Lambda_{2}^{n+1} - \Lambda_{2}^{n}),
\vdots
Y_{L-1}^{n+1} = -P(Y_{L-1}^{n}, \Lambda_{L-1}^{n}) + P_{y}(Y_{L-1}^{n}, \Lambda_{n}^{n})(Y_{L-1}^{n+1} - Y_{L-1}^{n}) + P_{\lambda}(Y_{L-1}^{n}, \Lambda_{n}^{n})(\Lambda_{L-1}^{n+1} - \Lambda_{L-1}^{n}),
\]

\[
\Lambda_{L}^{n+1} = Q(Y_{L}^{n}, \Lambda_{L}^{n}) + Q_{y}(Y_{L}^{n}, \Lambda_{n}^{n})(\Lambda_{L}^{n+1} - \Lambda_{L}^{n}) + Q_{\lambda}(Y_{L}^{n}, \Lambda_{n}^{n})(\Lambda_{L}^{n+1} - \Lambda_{L}^{n}),
\]

Note that this system is not triangular: the \( Y^{n+1}_{\ell} \) are coupled to the \( \Lambda^{n+1}_{\ell} \) and vice versa, which is clearly visible in the Jacobian in (2.6). This is in contrast to the initial value problem case, where the application of multiple shooting leads to a block lower triangular system.

The Parareal approximation idea is to replace the derivative term by a difference computed on a coarse grid in (2.7), i.e., to use the approximations

\[
P_{y}(Y_{t-1}^{n}, \Lambda_{t}^{n}) (Y_{t-1}^{n+1} - Y_{t-1}^{n}) \approx P_{G}(Y_{t-1}^{n+1}, \Lambda_{t}^{n}) - P_{G}(Y_{t-1}^{n}, \Lambda_{t}^{n}),
\]

where \( P_{G} \) and \( Q_{G} \) are propagators obtained from a coarse discretization of the subinterval problem (2.3), e.g., by using only one time step for the whole subinterval. This is certainly cheaper than evaluating the derivative on the fine grid; the remaining expensive fine grid operations \( P(Y_{t-1}^{n}, \Lambda_{t}^{n}) \) and \( Q(Y_{t-1}^{n}, \Lambda_{t}^{n}) \) in (2.7) can now all be performed in parallel. However, since (2.7) does not have a block triangular structure, the resulting nonlinear system would need to be solved iteratively. Each of these outer iterations is now very expensive, since one must evaluate the propagators \( P_{G}(Y_{t-1}^{n+1}, \Lambda_{t}^{n}) \), etc., by solving a coupled nonlinear local control problem. This is in contrast to initial value problems, where the additional cost of solving nonlinear
local problems is justified, because the block lower triangular structure allows one to solve the outer problem by forward substitution, without the need to iterate. In order to reduce the cost of computing outer residuals, our idea is not to use the Parareal approach (2.8) but to use the so-called derivative Parareal variant, where we approximate the derivative by effectively computing it for a coarse problem (see [12]),

\[
P_y(Y_{\ell+1}^n, \Lambda_{\ell}^n)(Y_{\ell+1}^{n+1} - Y_{\ell+1}^n) \approx P_y(Y_{\ell+1}^n, \Lambda_{\ell}^n)(Y_{\ell+1}^{n+1} - Y_{\ell+1}^n),
\[
P_{\lambda}(Y_{\ell+1}^n, \Lambda_{\ell}^n)(\Lambda_{\ell+1}^{n+1} - \Lambda_{\ell}^n) \approx P_{\lambda}(Y_{\ell+1}^n, \Lambda_{\ell}^n)(\Lambda_{\ell+1}^{n+1} - \Lambda_{\ell}^n),
\[
Q_{\lambda}(Y_{\ell+1}^n, \Lambda_{\ell}^n)(\Lambda_{\ell+1}^{n+1} - \Lambda_{\ell}^n) \approx Q_{\lambda}(Y_{\ell+1}^n, \Lambda_{\ell}^n)(\Lambda_{\ell+1}^{n+1} - \Lambda_{\ell}^n),
\]

The advantage of this approximation is that the computation of $P_y^G, P_{\lambda}^G$, etc., only involves linear problems. Indeed, for a small perturbation $\delta y$ in $Y_{\ell+1}$, the quantities $P_y^G(Y_{\ell+1}, \Lambda_{\ell})\delta y$ and $Q_y^G(Y_{\ell+1}, \Lambda_{\ell})\delta y$ can be computed by discretizing and solving the coupled differential equations obtained by differentiating (2.2). If $(y, \lambda)$ is the solution of (2.2) with $y(T_{\ell+1}) = Y_{\ell+1}$ and $\lambda(T_{\ell+1}) = \Lambda_{\ell}$, then solving the linear derivative system

\[
\begin{align*}
\dot{z} &= f(y)z + \mu/\alpha, \\
\dot{\mu} &= -f(y)^T \mu - H(y, z)^T \lambda, \\
z(T_{\ell+1}) &= \delta y, \\
\mu(T_{\ell+1}) &= 0
\end{align*}
\]

on a coarse time grid leads to

\[
z(T_{\ell+1}) = P_y^G(Y_{\ell+1}, \Lambda_{\ell})\delta y, \quad \mu(T_{\ell+1}) = Q_y^G(Y_{\ell+1}, \Lambda_{\ell})\delta y,
\]

where $H(y, z) = \lim_{\lambda \to 0} \frac{1}{\tau}[f'(y + \tau z) - f'(y)]$ is the Hessian of $f$ multiplied by $z$ and is thus linear in $z$. Similarly, to compute $P_{\lambda}^G(Y_{\ell+1}, \Lambda_{\ell})\delta \lambda$ and $Q_{\lambda}^G(Y_{\ell+1}, \Lambda_{\ell})\delta \lambda$ for a perturbation $\delta \lambda$ in $\Lambda_{\ell}$, it suffices to solve the same ODE system as (2.10), except the endpoint conditions must be replaced by $z(T_{\ell+1}) = 0$, $\mu(T_{\ell}) = \delta \lambda$. Therefore, if GMRES is used to solve the Jacobian system (2.5), then each matrix-vector multiplication requires only the solution of coarse, linear subproblems in parallel, which is much cheaper than solving coupled nonlinear subproblems in the standard Parareal approximation (2.8).

To summarize, our new ParaOpt method consists of solving for $n = 0, 1, 2, \ldots$ the system

\[
J^n \left( \begin{array}{c} Y^n \\ \Lambda^n \end{array} \right) \left( \begin{array}{c} Y^{n+1} - Y^n \\ \Lambda^{n+1} - \Lambda^n \end{array} \right) = J^n \left( \begin{array}{c} Y^n \\ \Lambda^n \end{array} \right)
\]

for $Y^{n+1}$ and $\Lambda^{n+1}$, where

\[
J^n \left( \begin{array}{c} Y \\ \Lambda \end{array} \right)
\]

\[
\begin{pmatrix}
-P_y^G(Y_0, \Lambda_1) & I & -P_y^G(Y_0, \Lambda_1) \\
& \ddots & \ddots & \ddots \\
& -P_y^G(Y_{L-1}, \Lambda_L) & I & -P_y^G(Y_{L-1}, \Lambda_L) \\
-Q_y^G(Y_1, \Lambda_2) & I & -Q_y^G(Y_1, \Lambda_2) \\
& \ddots & \ddots & \ddots \\
& -Q_y^G(Y_{L-1}, \Lambda_L) & I & -Q_y^G(Y_{L-1}, \Lambda_L) \\
I & \ddots & \ddots & \ddots \\
\end{pmatrix}
\]
is an approximation of the true Jacobian in (2.6). If the system (2.11) is solved using a matrix-free method, the action of the subblocks \( P^y_G, P^\lambda_G \), etc., can be obtained by solving coarse linear subproblems of the type (2.10). Note that the calculation of \( J^G \)

times a vector (without preconditioning) is embarrassingly parallel, since it requires only the solution of local subproblems of the type (2.10), with no additional coupling to other subintervals. Global communication is required in only two places: within the Krylov method itself (e.g., when calculating inner products) and possibly within the preconditioner. The design of an effective preconditioner is an important and technical topic that will be the subject of a future paper. Of course, for problems with small state spaces (e.g., for ODE control problems), direct methods may also be used, once the coefficients of \( J^G \) are calculated by solving (2.10) for suitable choices of \( \delta y \) and \( \delta \lambda \).

Regardless of how (2.11) is solved, since we use an approximation of the Jacobian, the resulting inexact Newton method will no longer converge quadratically but only linearly; this is true even in the case where the differential equation is linear. In the next section, we will analyze in detail the convergence of the method for the case of a diffusive linear problem.

3. Implicit Euler for the diffusive linear case. We now consider the method in a linear and discrete setting. More precisely, we focus on a control problem

\[
\dot{y}(t) = Ay(t) + c(t),
\]

where \( A \) is a real, symmetric matrix with negative eigenvalues. The matrix \( A \) can, for example, be a finite difference discretization of a diffusion operator in space. We will consider a discretize-then-optimize strategy, so the analysis that follows is done in a discrete setting.

3.1. Discrete formulation. To fix ideas, we choose the implicit Euler\(^1\) method for the time discretization; other discretizations will be studied in a future paper. Let \( M \in \mathbb{N} \), and \( \delta t = T/M \). Then the implicit Euler method gives\(^2\)

\[
y_{n+1} = y_n + \delta t(Ay_{n+1} + c_{n+1}),
\]

or, equivalently,

\[
y_{n+1} = (I - \delta tA)^{-1}(y_n + \delta tc_{n+1}).
\]

We minimize the cost functional

\[
J_{\delta t}(c) = \frac{1}{2}\|y_M - y_{\text{target}}\|^2 + \frac{\alpha}{2} \delta t \sum_{n=0}^{M-1} \|c_{n+1}\|^2.
\]

For the sake of simplicity, we keep the notation \( y, \lambda, \) and \( c \) for the discrete variables, that is, \( y = (y_n)_{n=0,\ldots,M}, \lambda = (\lambda_n)_{n=0,\ldots,M}, \) and \( c = (c_n)_{n=0,\ldots,M} \). Introducing the Lagrangian (see [29, 16] and also [22, 51, 47] for details)

\[
L_{\delta t}(y, \lambda, c) = J_{\delta t}(c) - \sum_{n=0}^{M-1} \langle \lambda_{n+1}, y_{n+1} - (I - \delta tA)^{-1}(y_n + \delta tc_{n+1}) \rangle,
\]

\(\text{We use the term “implicit Euler” instead of “backward Euler” because the method is applied forward and backward in time.}\)

\(\text{If the ODE system contains mass matrices arising from a finite element discretization, e.g., } M^y_{n+1} = M^y_n + \delta t(A^y_{n+1} + M^c_{n+1}), \text{ then one can analyze ParaOpt by introducing the change of variables } y_n := M^{1/2} y_n, \ c_n := M^{1/2} c_n, \text{ so as to obtain } y_{n+1} = y_n + \delta t(A^y_{n+1} + c_{n+1}), \text{ with } A := M^{-1/2} A M^{-1/2}. \text{ Since } A \text{ is symmetric positive definite whenever } A \text{ is, the analysis is identical to that for (3.2), even though one would never calculate } M^{1/2} \text{ and } A \text{ in actual computations.}\)
the optimality systems reads

\begin{align}
\text{(3.3)} & \quad y_0 = y_{\text{init}}, \\
\text{(3.4)} & \quad y_{n+1} = (I - \delta t A)^{-1}(y_n + \delta t c_{n+1}), \quad n = 0, 1, \ldots, M - 1, \\
\text{(3.5)} & \quad \lambda_M = y_M - y_{\text{target}}, \\
\text{(3.6)} & \quad \lambda_n = (I - \delta t A)^{-1}\lambda_{n+1}, \quad n = 0, 1, \ldots, M - 1, \\
\text{(3.7)} & \quad \alpha c_{n+1} = -(I - \delta t A)^{-1}\lambda_{n+1}, \quad n = 0, 1, \ldots, M - 1,
\end{align}

where we used the fact that $A$ is symmetric. If $A = VDV^T$ is the eigenvalue decomposition of $A$, then the transformation $y_n \mapsto V^T y_n$, $\lambda_n \mapsto V^T \lambda_n$, $c_n \mapsto V^T c_n$ allows us to diagonalize (3.3)–(3.7) and obtain a family of decoupled optimality systems of the form

\begin{align}
\text{(3.8)} & \quad y_0 = y_{\text{init}}, \\
\text{(3.9)} & \quad y_{n+1} = (I - \sigma \delta t)^{-1}(y_n + \delta t c_{n+1}), \quad n = 0, 1, \ldots, M - 1, \\
\text{(3.10)} & \quad \lambda_M = y_M - y_{\text{target}}, \\
\text{(3.11)} & \quad \lambda_n = (I - \sigma \delta t)^{-1}\lambda_{n+1}, \quad n = 0, 1, \ldots, M - 1, \\
\text{(3.12)} & \quad \alpha c_{n+1} = -(I - \sigma \delta t)^{-1}\lambda_{n+1}, \quad n = 0, 1, \ldots, M - 1,
\end{align}

where the $y_n$, $\lambda_n$, and $c_n$ are now scalars, and $\sigma < 0$ is an eigenvalue of $A$. This motivates us to study the scalar Dahlquist problem

$$
\dot{y}(t) = \sigma y(t) + c(t),
$$

where $\sigma$ is a real, negative number. For the remainder of this section, we will study the ParaOpt algorithm applied to the scalar variant (3.8)–(3.12), particularly its convergence properties as a function of $\sigma$.

Let us now write the linear ParaOpt algorithm for (3.8)–(3.12) in matrix form. For the sake of simplicity, we assume that the subdivision is uniform, that is, $T_\ell = \ell \Delta T$, where $N$ satisfies $\Delta T = N \delta t$ and $M = NL$; see Figure 1. We start by eliminating interior unknowns, i.e., ones that are not located at the time points $T_0, T_1, \ldots, T_L$. For $0 \leq n_1 \leq n_2 \leq M$, (3.9) and (3.12) together imply

\begin{equation}
\begin{aligned}
y_{n_2} &= (1 - \sigma \delta t)^{n_1 - n_2} y_{n_1} - \delta t \sum_{j=0}^{n_2 - n_1 - 1} (1 - \sigma \delta t)^{n_1 - n_2 + j} c_{n_1 + j + 1} \\
&= (1 - \sigma \delta t)^{n_1 - n_2} y_{n_1} - \frac{\delta t}{\alpha} \sum_{j=0}^{n_2 - n_1 - 1} (1 - \sigma \delta t)^{n_1 - n_2 + j - 1} \lambda_{n_1 + j + 1}.
\end{aligned}
\end{equation}

On the other hand, (3.11) implies

\begin{equation}
\lambda_{n_1 + j} = (1 - \sigma \delta t)^{n_1 - n_2 + j} \lambda_{n_2}.
\end{equation}

\begin{tabular}{cccccc}
$T_0$ & $T_1$ & $T_\ell$ & $T_{L-1}$ & $T_L$ = T \\
$Y_0$ = $y_{\text{init}}$ & $Y_1$ = $y_N$ & $Y_\ell$ = $y_{\ell N}$ & $Y_{L-1}$ = $y_{(L-1)N}$ & $Y_L$ = $y_{LN}$ \\
$\Lambda_1$ = $\lambda_N$ & $\Lambda_\ell$ = $\lambda_{\ell N}$ & $\Lambda_{L-1}$ = $\lambda_{(L-1)N}$ & $\Lambda_L$ = $\lambda_{LN}$
\end{tabular}

\textbf{Fig. 1.} Notation associated with the parallelization setting.
Combining (3.13) and (3.14) then leads to

\begin{equation}
\begin{aligned}
y_{n_2} &= (1 - \sigma \delta t)^{n_1-n_2} y_{n_1} - \frac{\delta t}{\alpha} \sum_{j=0}^{n_2-n_1-1} (1 - \sigma \delta t)^{2(n_1-n_2+j)} \lambda_{n_2}.
\end{aligned}
\end{equation}

Setting \(n_1 = (\ell - 1)N\) and \(n_2 = \ell N\), and using the notation \(Y_\ell = y_{\ell N}, \Lambda_\ell = \lambda_{\ell N}\) (see Figure 1), we obtain from (3.14) and (3.15) the equations

\begin{align}
Y_0 &= y_{\text{init}}, \\
-\beta_{\delta t} Y_{\ell-1} + Y_\ell + \frac{\gamma_{\delta t}}{\alpha} \Lambda_\ell &= 0, & 1 \leq \ell \leq M, \\
\Lambda_{\ell-1} - \beta_{\delta t} \Lambda_\ell &= 0, & 0 \leq \ell \leq M - 1, \\
Y_\ell + \Lambda_\ell &= y_{\text{target}},
\end{align}

where

\begin{align}
\beta_{\delta t} &= (1 - \sigma \delta t)^{-\Delta T/\delta t}, \\
\gamma_{\delta t} &= \delta t \sum_{j=0}^{N-1} (1 - \sigma \delta t)^{2(j-N)} = \frac{\beta_{\delta t}^2 - 1}{\sigma(2 - \sigma \delta t)}.
\end{align}

In matrix form, this can be written as

\begin{align}
\begin{pmatrix}
1 & & & & \\
-\beta_{\delta t} & \ddots & & & \\
& \ddots & \ddots & & \\
& & -\beta_{\delta t} & 1 & \\
-1 & & & & \ddots
\end{pmatrix}
\begin{pmatrix}
0 \\
\gamma_{\delta t}/\alpha \\
\ddots \\
\ddots \\
\gamma_{\delta t}/\alpha \\
\ddots \\
\ddots \\
\ddots \\
1
\end{pmatrix}
\begin{pmatrix}
Y_0 \\
\vdots \\
\vdots \\
\vdots \\
Y_L \\
\Lambda_1 \\
\vdots \\
\Lambda_L
\end{pmatrix}
= 
\begin{pmatrix}
y_{\text{init}} \\
0 \\
\vdots \\
\vdots \\
0
\end{pmatrix}
\end{align}

or, in a more compact form,

\begin{equation}
A_{\delta t} X = b.
\end{equation}

Note that this matrix has the same structure as the Jacobian matrix \(F\) in (2.6), except that \(Q_\lambda = 0\) for the linear case. In order to solve (3.18) numerically, we consider a second time step \(\Delta t\) such that \(\delta t \leq \Delta t \leq \Delta T\). In other words, for each subinterval of length \(\Delta T\), the derivatives of the propagators \(P_y, Q_y, P_\lambda, Q_\lambda\) are approximated using a coarser time discretization with time step \(\Delta t \leq \Delta T\). The optimality system for this coarser time discretization has the form

\begin{equation}
A_{\Delta t} \hat{X} = b,
\end{equation}

where \(A_{\Delta t}\) has the same form as above, except that \(\beta_{\delta t}\) and \(\gamma_{\delta t}\) are replaced by \(\beta_{\Delta t}\) and \(\gamma_{\Delta t}\), i.e., the values obtained from the formulas (3.16) and (3.17) when one replaces \(\delta t\) by \(\Delta t\). Then the ParaOpt algorithm (2.11)–(2.12) for the linear Dahlquist problem can be written as

\begin{align}
A_{\Delta t}(X^{k+1} - X^k) &= -F(X^k) = -(A_{\delta t} X^k - b)
\end{align}
or, equivalently,
\begin{equation}
X^{k+1} = (I - A_{\Delta t}^{-1}A_{\delta t})X^k + A_{\Delta t}^{-1}b.
\end{equation}
Note that using this iteration, only a coarse matrix needs to be inverted.

3.2. Eigenvalue problem. In order to study the convergence of the iteration (3.19), we study the eigenvalues of the matrix \( I - A_{\Delta t}^{-1}A_{\delta t} \), which are given by the generalized eigenvalue problem
\begin{equation}
(A_{\Delta t} - A_{\delta t})x = \mu A_{\Delta t}x,
\end{equation}
with \( x = (v_0, v_1, \ldots, v_L, w_1, \ldots, w_L)^T \) being the eigenvector associated with the eigenvalue \( \mu \). Since \( A_{\Delta t} - A_{\delta t} \) has two zero rows, the eigenvalue \( \mu = 0 \) must have multiplicity at least two. Now let \( \mu \neq 0 \) be a nonzero eigenvalue. (If no such eigenvalue exists, then the preconditioning matrix is nilpotent and the iteration converges in a finite number of steps.) Writing (3.20) componentwise yields
\begin{align}
(3.21) & \quad v_0 = 0, \\
(3.22) & \quad \mu (v_{\ell} - \beta v_{\ell-1} + \gamma w_{\ell}/\alpha) = -\delta \beta v_{\ell-1} + \delta \gamma w_{\ell}/\alpha, \\
(3.23) & \quad \mu (w_{\ell} - \beta w_{\ell+1}) = -\delta \beta w_{\ell+1}, \\
(3.24) & \quad \mu (w_L - v_L) = 0,
\end{align}
where we have introduced the simplified notation
\begin{equation}
\beta = \beta_{\Delta t}, \quad \gamma = \gamma_{\Delta t}, \quad \delta \beta = \beta_{\Delta t} - \beta_{\delta t}, \quad \delta \gamma = \gamma_{\Delta t} - \gamma_{\delta t}.
\end{equation}
The recurrences (3.22) and (3.23) are of the form
\begin{equation}
\ell = a v_{\ell-1} + b w_{\ell}, \quad w_{\ell} = a w_{\ell+1},
\end{equation}
where
\begin{equation}
a = \beta - \mu^{-1} \delta \beta, \quad b = \frac{-\gamma + \mu^{-1} \delta \gamma}{\alpha}.
\end{equation}
Solving the recurrence (3.26) in \( v \) together with the initial condition (3.21) leads to
\begin{equation}
v_L = \sum_{\ell=1}^{L} a^{L-\ell} b w_{\ell},
\end{equation}
whereas the recurrence (3.26) in \( w \) simply gives
\begin{equation}
w_{\ell} = a^{L-\ell} w_L.
\end{equation}
Combining (3.27) and (3.28), we obtain
\begin{equation}
v_L = \left( \sum_{\ell=1}^{L} a^{2(L-\ell)} b \right) w_L,
\end{equation}
so that (3.24) gives rise to \( P(\mu)w_L = 0 \), with
\begin{equation}
P(\mu) = \alpha \mu^{2L-1} + (\mu \gamma - \delta \gamma) \sum_{\ell=0}^{L-1} \mu^{2(L-\ell)-1}(\mu \beta - \delta \beta)^{2\ell}.
\end{equation}
Since we seek a nontrivial solution, we can assume \( w_L \neq 0 \). Therefore, the eigenvalues of \( I - A_{\Delta t}^{-1}A_{\delta t} \) consist of the number zero (with multiplicity two), together with the \( 2L - 1 \) roots of \( P(\mu) \), which are all nonzero. In the next subsection, we will give a precise characterization of the roots of \( P(\mu) \), which depend on \( \alpha \), as well as on \( \sigma \) via the parameters \( \beta, \delta \beta, \gamma, \) and \( \delta \gamma \).
3.3. Characterization of eigenvalues. In the next two results, we describe the location of the roots of \( P(\mu) \) from the last section or, equivalently, the nonzero eigenvalues of the iteration matrix \( I - A_{\Delta t}^{-1} A_{\delta t} \). We first establish the sign of a few parameters in the case \( \sigma < 0 \), which is true for diffusive problems.

**Lemma 3.1.** Let \( \sigma < 0 \). Then we have \( 0 < \beta < 1, 0 < \delta \beta < \beta, \gamma > 0, \) and \( \delta \gamma < 0 \).

**Proof.** By the definitions (3.16) and (3.25), we see that

\[
\beta = \beta_{\Delta t} = (1 - \sigma \Delta t)^{-\Delta t/\Delta t},
\]

which is between 0 and 1, since \( 1 - \sigma \Delta t > 1 \) for \( \sigma < 0 \). Moreover, \( \beta_{\Delta t} \) is an increasing function of \( \Delta t \) by direct calculation, so that

\[
\delta \beta = \beta_{\Delta t} - \beta_{\delta t} > 0,
\]

which shows that \( 0 < \delta \beta < \beta \). Next, we have by definition

\[
\gamma = \frac{\beta^2 - 1}{\sigma(2 - \sigma \Delta t)}.
\]

Since \( \beta < 1 \) and \( \sigma < 0 \), both the numerator and the denominator are negative, so \( \gamma > 0 \). Finally, we have

\[
\delta \gamma = \frac{1}{|\sigma|} \left( \frac{1 - \beta_{\Delta t}^2}{2 + |\sigma| \Delta t} - \frac{1 - \beta_{\delta t}^2}{2 + |\sigma| \delta t} \right) < 0
\]

since \( 1 - \beta_{\Delta t}^2 < 1 - \beta_{\delta t}^2 \) and \( 2 + |\sigma| \Delta t > 2 + |\sigma| \delta t \), so the first quotient inside the parentheses is necessarily smaller than the second quotient.

We are now ready to prove a first estimate for the eigenvalues of the matrix \( I - A_{\Delta t}^{-1} A_{\delta t} \).

**Theorem 3.2.** Let \( P \) be the polynomial defined in (3.29). For \( \sigma < 0 \), the roots of \( P \) are contained in the set \( D_\sigma \cup \{ \mu^* \} \), where

\[
D_\sigma = \{ \mu \in \mathbb{C} : |\mu - \mu_0| < \delta \beta/(1 - \beta^2) \},
\]

where \( \mu_0 = -\delta \beta/(1 - \beta^2) \), and \( \mu^* < 0 \) is a real negative number.

**Proof.** Since zero is not a root of \( P(\mu) \), we can divide \( P(\mu) \) by \( \mu^{2L-1} \) and see that \( P(\mu) \) has the same roots as the function

\[
\hat{P}(\mu) = \alpha + (\gamma - \mu^{-1} \delta \gamma) \sum_{\ell=0}^{L-1} (\beta - \mu^{-1} \delta \beta)^{2\ell}.
\]

Recall the change of variables

\[
a = \beta - \mu^{-1} \delta \beta \iff \mu = \frac{\delta \beta}{\beta - a};
\]

substituting \( a \) into \( \hat{P}(\mu) \) and multiplying the result by \( \delta \beta/|\delta \gamma| \) shows that \( P(\mu) = 0 \) is equivalent to

\[
Q(a) := \frac{\alpha \delta \beta}{|\delta \gamma|} + (C - a) \sum_{\ell=0}^{L-1} a^{2\ell} = 0,
\]
We now concentrate on finding values of $\theta$ such that $\arg(f(e^{i\theta})) = \pi \pmod{2\pi}$. Since $f(\pi) = f(\alpha)$, it suffices to consider the range $0 \leq \theta \leq \pi$, and the other half of the range will follow by conjugation. Since $f$ is a product, we deduce that

$$\arg(f(e^{i\theta})) = \arg(C - e^{i\theta}) + \arg(1 + e^{2i\theta} + \cdots + e^{2(L-1)i\theta}).$$

We consider the two terms on the right separately. 

- For the first term, we have for all $0 < \theta < \pi$
  $$\theta - \pi < \arg(-e^{i\theta}) < \arg(C - e^{i\theta}) < 0,$$
  since $C$ is real and positive. For $\theta = \pi$, we obviously have $\arg(C - e^{i\theta}) = 0$, whereas for $\theta = 0$, we have $\arg(C - e^{i\theta}) = -\pi$ if $C < 1$, and $\arg(C - e^{i\theta}) = 0$ otherwise.

- For the second term, observe that for $0 < \theta < \pi$, we have
  $$1 + e^{2i\theta} + \cdots + e^{2(L-1)i\theta} = \frac{1 - e^{2iL\theta}}{1 - e^{2i\theta}} = e^{(L-1)i\theta} \cdot \frac{\sin(L\theta)}{\sin(\theta)}.$$

Therefore, the second term is piecewise linear with slope $L - 1$, with a jump of size $\pi$ whenever $\sin(L\theta)$ changes sign, i.e., at $\theta = k\pi/L$, $k = 1, \ldots, L - 1$. Put within the range $(-\pi, \pi)$, we can write

$$\arg\left(\frac{1 - e^{2iL\theta}}{1 - e^{2i\theta}}\right) = (L - 1)\theta - \left\lfloor \frac{L\theta}{\pi} \right\rfloor \pi =: g(\theta), \quad 0 < \theta < \pi.$$

We also have $g(0) = g(\pi) = 0$ by direct calculation. The function $g$ satisfies the property $-\pi \leq g(\theta) \leq \pi - \theta$; see Figure 2.

From the above, we deduce that $\arg(f(e^{i\theta})) < \pi$ for all $0 \leq \theta \leq \pi$. Moreover,

$$\arg(f(e^{i\theta})) = \begin{cases} 0 & \text{if } \theta = 0 \text{ and } C > 1, \\ -\pi & \text{if } \theta = 0 \text{ and } C < 1, \\ \arg(C - e^{i\theta}) + g(\theta) > -\pi & \text{if } 0 < \theta < \pi, \\ 0 & \text{if } \theta = \pi. \end{cases}$$
Thus, the winding number around the point $-\alpha\delta\beta/|\delta\gamma|$ cannot exceed one, so at most one of the roots of $Q$ can lie inside the unit disc. If there is indeed such a root $a^*$, it must be real, since the conjugate of any root of $Q$ is also a root. Moreover, it must satisfy $a^* > C$, since $Q(a) > 0$ for any $a \leq C$. This implies

$$\beta - a^* < \beta - C = -\frac{\gamma\delta\beta}{|\delta\gamma|} < 0,$$

so the corresponding $\mu^* = \delta\beta/(\beta - a^*)$ must also be negative.

We have seen that the existence of $\mu^*$ depends on whether the constant $C$ is larger than 1. The following lemma shows that we indeed have $C < 1$.

**Lemma 3.3.** Let $\sigma < 0$. Then the constant $C = \beta + \gamma\delta\beta/|\delta\gamma|$, defined in (3.31), satisfies $C < 1$.

**Proof.** We first transform the relation $C < 1$ into a sequence of equivalent inequalities. Starting with the definition of $C$, we have

$$C = \beta_{\Delta t} + \frac{\gamma_{\Delta t}(\beta_{\Delta t} - \beta_{\delta t})}{\gamma_{\delta t} - \gamma_{\Delta t}} < 1 \iff \beta_{\Delta t}(\gamma_{\delta t} - \gamma_{\Delta t}) + \gamma_{\Delta t}(\beta_{\Delta t} - \beta_{\delta t}) < \gamma_{\delta t} - \gamma_{\Delta t}$$

$$\iff \gamma_{\Delta t}(1 - \beta_{\delta t}) < \gamma_{\delta t}(1 - \beta_{\Delta t})$$

$$\iff \frac{(1 - \beta_{\delta t}^2)(1 - \beta_{\Delta t})}{|\sigma|(2 + |\sigma|\Delta t)} < \frac{(1 - \beta_{\Delta t}^2)(1 - \beta_{\delta t})}{|\sigma|(2 + |\sigma|\delta t)}$$

$$\iff \frac{1 + \beta_{\Delta t}}{2 + |\sigma|\Delta t} < \frac{1 + \beta_{\delta t}}{2 + |\sigma|\delta t},$$

where the last equivalence is obtained by multiplying both sides of the penultimate inequality by $|\sigma|$ and then dividing it by $(1 - \beta_{\Delta t})(1 - \beta_{\delta t})$. By the definition of $\beta_{\Delta t}$ and $\beta_{\delta t}$, the last inequality can be written as $f(|\sigma|\Delta t) < f(|\sigma|\delta t)$, where

$$f(x) := \frac{1 + (1 + x)^{-k/x}}{2 + x}$$
with \( k = |\sigma| \Delta T > 0 \). Therefore, it suffices to show that \( f(x) \) is decreasing for \( 0 < x \leq k \). In other words, we need to show that

\[
f'(x) = \frac{(1 + x)^{-k/x}}{2 + x} \left[ \frac{k \ln(1 + x)}{x^2} - \frac{k}{x(1 + x)} \right] - \frac{1 + (1 + x)^{-k/x}}{(2 + x)^2} < 0.
\]

This is equivalent to showing

\[
f'(x) = (2 + x) \left[ \frac{k \ln(1 + x)}{x^2} - \frac{k}{x(1 + x)} \right] - 1 < (1 + x)^{k/x}.
\]

Using the fact that \( \ln(1 + x) \leq x \), we see that the left-hand side is bounded above by

\[
(2 + x) \left[ \frac{k \ln(1 + x)}{x^2} - \frac{k}{x(1 + x)} \right] - 1 < (2 + x) \left[ \frac{k}{x^2} - \frac{k}{x(1 + x)} \right] - 1 = k \left( \frac{2 + x}{1 + x} \right) - 1.
\]

But for every \( k > 0 \) and \( 0 < x < k \) we have

\[
(1 + x)^{k/x} > k \left( \frac{2 + x}{1 + x} \right) - 1;
\]

see the proof in the appendix. Therefore, (3.33) is satisfied by all \( k > 0 \) and \( 0 < x < k \), so \( f \) is in fact decreasing. It follows that \( C < 1 \), as required.

**Theorem 3.4.** Let \( \sigma < 0 \) be fixed, and let

\[
L_0 := \frac{C - \beta}{\gamma(1 - C)}.
\]

Then the spectrum of \( I - A_\delta^{-1} A_{\delta t} \) has an eigenvalue \( \mu^* \) outside the disc \( D_\sigma \) defined in (3.30) if and only if the number of subintervals \( L \) satisfies \( L > \alpha L_0 \), where \( \alpha \) is the regularization parameter.

**Proof.** The isolated eigenvalue exists if and only if the winding number of \( Q(e^{i\theta}) \) about the origin is nonzero. Since \( Q(e^{i\theta}) \) only intersects the negative real axis at most once, we see that the winding number is nonzero when \( Q(-1) < 0 \), i.e., when

\[
\frac{\alpha \delta \beta}{\delta \gamma} + (C - 1) L < 0.
\]

Using the definition of \( C \), this leads to

\[
\frac{\alpha(C - \beta)}{\gamma} + (C - 1) L < 0 \iff L > \frac{\alpha(C - \beta)}{\gamma(1 - C)},
\]

hence the result. \( \square \)

### 3.4. Spectral radius estimates

The next theorem now gives a more precise estimate on the isolated eigenvalue \( \mu^* \).

**Theorem 3.5.** Suppose that the number of intervals \( L \) satisfies \( L > \alpha L_0 \), with \( L_0 \) defined in (3.35). Then the real negative eigenvalue \( \mu^* \) outside the disc \( D_\sigma \) is bounded below by

\[
\mu^* > -\frac{|\delta \gamma| + \alpha \delta \beta(1 + \beta)}{\gamma + \alpha(1 - \beta^2)}.
\]
Proof. Suppose \( a^* = \beta - \delta \beta / \mu^* \) is a real root of \( Q(a) \) inside the unit disc. We have seen at the end of the proof of Theorem 3.2 (immediately before (3.32)) that \( a^* > C \); moreover, since \( a^* \) is assumed to be inside the unit circle, we must have \( a^* < 1 \). Therefore, \( a^* \) satisfies \( C < a^* < 1 \). This implies

\[
\frac{\alpha \delta \beta}{|\delta \gamma|} + \frac{C - a^*}{1 - (a^*)^2} = \frac{(C - a^*)(a^*)^{2L}}{1 - (a^*)^2} < 0.
\]

Therefore, \( a^* \) satisfies

\[
(1 - (a^*)^2)\alpha \delta \beta + |\delta \gamma|(C - a^*) < 0,
\]

which means

\[
a^* > -\frac{|\delta \gamma| + \sqrt{|\delta \gamma|^2 + 4\alpha \delta \beta (\alpha \delta \beta + C|\delta \gamma|)}}{2\alpha \delta \beta} = -\frac{|\delta \gamma| + \sqrt{|\delta \gamma|^2 + 2\alpha \delta \beta} - 4(1 - C)\alpha \delta \beta |\delta \gamma|}{2\alpha \delta \beta}.
\]

Therefore,

\[
\mu^* = \frac{\delta \beta}{\beta - a^*} > \frac{2\alpha \delta \beta^2}{2\alpha \delta \beta^2 - (2\alpha \delta \beta + |\delta \gamma|) - \sqrt{(\delta \gamma)^2 + 2\alpha \delta \beta} - 4(1 - C)\alpha \delta \beta |\delta \gamma|}
\]

\[
= \frac{(2\alpha \delta \beta + |\delta \gamma|) + \sqrt{(\delta \gamma)^2 + 2\alpha \delta \beta} - 4(1 - C)\alpha \delta \beta |\delta \gamma|}{2\alpha \delta \beta (\beta^2 - 1)}
\]

\[
= -\frac{|\delta \gamma| + \alpha \delta \beta (1 + \beta)}{\gamma + \alpha (1 - \beta^2)},
\]

where the last inequality is obtained by dropping the term containing \( (1 - C) \) inside the square root, which makes the square root larger since \( C < 1 \).

To illustrate the above theorems, we show in Figures 3 and 4 the spectrum of the iteration matrix \( I - A_{\alpha}^{-1}A_{\delta} \) for different values of \( \sigma \) and for \( \alpha = 1 \) and 1000. Here, the time interval \([0, T]\) is subdivided into \( L = 30 \) subintervals, and each subinterval contains 50 coarse time steps and 5000 fine time steps. Table 1 shows the values of the relevant parameters. For \( \alpha = 1 \), we see that there is always one isolated eigenvalue on the negative real axis, since \( L > L_0 \) in all cases, and its location is predicted rather accurately by the formula (3.36). The rest of the eigenvalues all lie within the disc \( D_\sigma \) defined in (3.30). For \( \alpha = 1000 \), the bounding disc is identical to the previous case; however, since we have \( L < \alpha L_0 \) for all cases except for \( \sigma = -16 \), we observe no eigenvalue outside the disc, except for the very last case. In that very last case, we have \( |\delta \gamma| = 0.0107 \), so (3.36) gives the lower bound \( \mu^* > -1.07 \times 10^{-5} \), which again is quite accurate when compared with the bottom right panel of Figure 4.
Corollary 3.6. Let $T$, $\Delta T$, $\Delta t$, $\delta t$, $\alpha$, and $\sigma$ be fixed. Then the spectral radius $\rho$ of the matrix $I - A_{\Delta t}^{-1} A_{\delta t}$ satisfies

$$\rho \leq \frac{|\delta \gamma| + \alpha \delta \beta (1 + \beta)}{\gamma + \alpha (1 - \beta^2)}.$$  

Note that the inequality (3.36) is valid for all $L > 0$, i.e., regardless of whether the isolated eigenvalue $\mu^*$ exists.

**Proof.** When the number of subintervals $L$ satisfies $L > \alpha L_0$, the spectral radius is determined by the isolated eigenvalue, which according to Theorem 3.5 is estimated by

$$|\mu^*| < \frac{|\delta \gamma| + \alpha \delta \beta (1 + \beta)}{\gamma + \alpha (1 - \beta^2)}.$$
Fig. 4. Spectrum of the iteration matrix for \( T = 100, L = 30, \Delta T/\Delta t = 50, \Delta t/\delta t = 100, \alpha = 1000 \) and for \( \sigma = -1/8, -1/4, -1/2, -1, -2, -16 \), from top left to bottom right.

Table 1

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>( \beta )</th>
<th>( \gamma )</th>
<th>( C )</th>
<th>( L_0 )</th>
<th>Radius of ( D_\sigma )</th>
<th>( \mu^* ) bound (( \alpha = 1 ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-1/8)</td>
<td>0.6604</td>
<td>2.2462</td>
<td>0.8268</td>
<td>0.4280</td>
<td>( 2.00 \times 10^{-3} )</td>
<td>(-6.08 \times 10^{-3} )</td>
</tr>
<tr>
<td>(-1/4)</td>
<td>0.4976</td>
<td>1.6037</td>
<td>0.6960</td>
<td>0.5300</td>
<td>( 3.67 \times 10^{-3} )</td>
<td>(-9.34 \times 10^{-3} )</td>
</tr>
<tr>
<td>(-1/2)</td>
<td>0.1941</td>
<td>0.9466</td>
<td>0.4713</td>
<td>0.5539</td>
<td>( 5.35 \times 10^{-3} )</td>
<td>(-1.24 \times 10^{-2} )</td>
</tr>
<tr>
<td>(-1)</td>
<td>0.0397</td>
<td>0.4831</td>
<td>0.1588</td>
<td>0.2930</td>
<td>( 3.97 \times 10^{-3} )</td>
<td>(-1.36 \times 10^{-2} )</td>
</tr>
<tr>
<td>(-2)</td>
<td>0.0019</td>
<td>0.2344</td>
<td>0.0116</td>
<td>0.0417</td>
<td>( 6.36 \times 10^{-4} )</td>
<td>(-1.30 \times 10^{-2} )</td>
</tr>
<tr>
<td>(-16)</td>
<td>( 1.72 \times 10^{-16} )</td>
<td>0.0204</td>
<td>( 5 \times 10^{-16} )</td>
<td>( 1.61 \times 10^{-14} )</td>
<td>( 1.72 \times 10^{-16} )</td>
<td>(-1.05 \times 10^{-2} )</td>
</tr>
</tbody>
</table>

Otherwise, when \( L \leq \alpha L_0 \), all the eigenvalues lie within the bounding disc \( D_\sigma \), so no eigenvalue can be farther away from the origin than

\[
\text{Radius}(D_\sigma) + |\text{Center}(D_\sigma)| = \frac{\delta \beta}{1 - \beta^2} + \frac{\beta \delta \beta}{1 - \beta^2} = \frac{\delta \beta}{1 - \beta^2}.
\]
A straightforward calculation shows that

\[
\frac{1}{\gamma + \alpha(1 - \beta^2)} \delta \gamma + \alpha \delta \beta (1 + \beta) > \frac{\delta \beta}{1 - \beta} \quad \text{if and only if} \quad \beta + \frac{\gamma \delta \beta}{\delta \gamma} < 1,
\]

which is true by Lemma 3.3. Thus, the inequality (3.36) holds in both cases.

The above corollary is of interest when we apply our ParaOpt method to a large system of ODEs (arising from the spatial discretization of a PDE, for example), where the eigenvalues lie in the range \( \sigma \in [-\sigma_{\text{max}}, -\sigma_{\text{min}}] \), with \( \sigma_{\text{max}} \to \infty \) when the spatial grid is refined. As we can see from Figure 5, the upper bound follows the actual spectral radius rather closely for most values of \( \sigma \), and its maximum occurs roughly...
at the same value of $\sigma$ as the one that maximizes the spectral radius. In the next two results, we will use the estimate (3.36) of the spectral radius of $I - A_{\Delta t}^{-1} A_t$ to derive a criterion for the convergence of the method.

**Lemma 3.7.** Let $T$, $\Delta T$, $\Delta t$, $\delta t$ be fixed. Then for all $\sigma < 0$, we have

\begin{equation}
|\delta \gamma| \leq 1.58|\sigma|(\Delta t - \delta t),
\end{equation}

\begin{equation}
\frac{\delta \beta}{1 - \beta} \leq 0.3.
\end{equation}

**Proof.** To bound $|\delta \gamma|/\gamma$, we start by bounding a scaled version of the quantity. We first use the definition of $\gamma$ and $\gamma_{\delta t}$ (cf. (3.17)) to obtain

\begin{equation}
\frac{|\delta \gamma|}{\gamma} = \frac{1}{|\sigma| (\Delta t - \delta t)} (\gamma_{\delta t} - \gamma) = \frac{2 + |\gamma| \Delta t}{(1 - \beta^2)|\sigma| (\Delta t - \delta t)} (1 - \beta_{\delta t}^2) = \frac{1 - \beta_{\delta t}^2}{(2 + \sigma \delta t)(1 - \beta^2)} + \frac{\beta^2 - \beta_{\delta t}^2}{|\sigma| (\Delta t - \delta t)(1 - \beta^2)} =: A + B.
\end{equation}

To estimate the terms $A$ and $B$ above, we define the mapping

\begin{equation}
h_{\Delta T}(\tau) := (1 + |\sigma| \tau)^{-\Delta T/\tau},
\end{equation}

so that $\beta = h_{\Delta T}(\Delta t)$, $\beta_{\delta t} = h_{\Delta T}(\delta t)$. Using the fact that $\ln(1 + x) > \frac{x}{1 + x}$ for $x > 0$ (see Lemma A.2 in Appendix A), we see that

\begin{equation}
h'_{\Delta T}(\tau) = h_{\Delta T}(\tau) \left[ \frac{\Delta T}{\tau^2} \ln(1 + |\sigma| \tau) - \frac{|\sigma| \Delta T}{\tau(1 + |\sigma| \tau)} \right] > 0,
\end{equation}

so $h_{\Delta T}$ is increasing. Therefore, we have

\begin{equation}
\lim_{\tau \to 0} h_{\Delta T}(\tau) = e^{-|\sigma| \Delta T} \leq \beta_{\delta t} \leq \beta = \frac{1}{1 + |\sigma| \Delta T} = h_{\Delta T}(\Delta T).
\end{equation}

It then follows that

\begin{equation}
A := \frac{1 - \beta_{\delta t}^2}{(2 + \sigma \delta t)(1 - \beta^2)} \leq \frac{1 - e^{-2|\sigma| \Delta T}}{(2 + \sigma \delta t)(1 - (1 + |\sigma| \Delta T)^{-2})} \leq \frac{(1 - e^{-2|\sigma| \Delta T})(1 + |\sigma| \Delta T)^2}{2|\sigma| \Delta T(2 + |\sigma| \Delta T)}.
\end{equation}

The last quotient is a function in $|\sigma| \Delta T$ only, whose maximum over all $|\sigma| \Delta T > 0$ is approximately 0.5773 < 0.58; therefore, we have

\begin{equation}
A \leq 0.58.
\end{equation}

For the second term, we use the mean value theorem and the fact that $\beta^2 = h_{2\Delta T}(\Delta t)$, $\beta_{\delta t}^2 = h_{2\Delta T}(\delta t)$ to obtain

\begin{equation}
\beta^2 - \beta_{\delta t}^2 = (\Delta t - \delta t) h'_{2\Delta T}(\tau^*)
\end{equation}

for some $\delta t < \tau^* < \Delta t$, with

\begin{equation}
h'_{2\Delta T}(\tau) = h_{2\Delta T}(\tau) \left[ \frac{2\Delta T}{\tau^2} \ln(1 + |\sigma| \tau) - \frac{2|\sigma| \Delta T}{\tau(1 + |\sigma| \tau)} \right].
\end{equation}
Using the fact that $\ln(1 + x) \leq x$ for all $x \geq 0$, we deduce that

$$h_{2\Delta T}(\tau^*) \leq h_{2\Delta T}(\tau^*) \frac{2|\sigma|^2 \Delta T}{1 + |\sigma|^2} \leq \frac{2\beta^2 |\sigma|^2 \Delta T}{1 + |\sigma|^2 \delta t},$$

so that

$$B := \frac{\beta^2 - \beta_t^2}{|\sigma|(\Delta t - \delta t)(1 - \beta^2)} \leq \frac{2|\sigma| \Delta T}{(1 + |\sigma| \Delta T)^2}, \quad \frac{(1 + |\sigma| \Delta T)^2}{|\sigma| \Delta T(2 + |\sigma| \Delta T)} \leq 1.$$

Combining the estimates for $A$ and $B$ and multiplying by $|\sigma| |\Delta t - \delta t|$ gives the first inequality in (3.37). For the second inequality, we use (3.38) to obtain

$$\frac{-\beta - \beta_t}{1 - \beta} \leq \frac{(1 + |\sigma| |\Delta T|)^{-1} - e^{-|\sigma| |\Delta T|}}{1 - (1 + |\sigma| |\Delta T|)^{-1}} = \frac{1 - (1 + |\sigma| |\Delta T|) e^{-|\sigma| |\Delta T|}}{|\sigma| |\Delta T|}.$$

This is again a function in a single variable $|\sigma| |\Delta T|$, whose maximum over all $|\sigma| |\Delta T| > 0$ is approximately 0.2984 < 0.3.

**Theorem 3.8.** Let $\Delta T$, $\Delta t$, $\delta t$, and $\alpha$ be fixed. Then for all $\sigma < 0$, the spectral radius of $I - A_{\Delta t}^{-1} A_{\delta t}$ satisfies

$$\max_{\sigma < 0} \rho(\sigma) \leq \frac{0.79 \Delta t}{\alpha + \sqrt{\alpha \Delta t}} + 0.3.$$

Thus, if $\alpha > 0.4544 \Delta t$, then the linear ParaOpt algorithm (3.19) converges.

**Proof.** Starting with the spectral radius estimate (3.36), we divide the numerator and denominator by $\gamma$, then substitute its definition in (3.17) to obtain

$$\rho(\sigma) \leq \frac{|\delta \gamma| + \alpha \delta \beta (1 + \beta)}{\gamma + \alpha (1 - \beta^2)} = \frac{|\delta \gamma| + \alpha \delta \beta |\sigma| (2 + |\sigma| \Delta t)}{\gamma + \alpha |\sigma| (2 + |\sigma| \Delta t)} \leq \frac{|\delta \gamma|}{\gamma (1 + \alpha |\sigma| (2 + |\sigma| \Delta t))} + \frac{\alpha \delta \beta}{1 - \beta}.$$

Now, by Lemma 3.7, the first term is bounded above by

$$f(\sigma) := \frac{1.58 |\sigma| \Delta t}{1 + \alpha |\sigma| (2 + |\sigma| \Delta t)},$$

whose maximum occurs at $\sigma^* = -1/\sqrt{\alpha \Delta t}$ with

$$f(\sigma^*) = \frac{0.79 \Delta t}{\sqrt{\alpha \Delta t} + \alpha}.$$

Together with the estimate on $\delta \beta/(1 - \beta)$ in Lemma 3.7, this proves (3.39). Thus, a sufficient condition for the method (3.19) to converge can be obtained by solving the inequality

$$\frac{0.79 \Delta t}{\alpha + \sqrt{\alpha \Delta t}} + 0.3 < 1.$$

This is a quadratic equation in $\sqrt{\alpha}$; solving it leads to $\alpha > 0.4544 \Delta t$, as required. \(\square\)

In Figure 6, we show the maximum spectral radius of $I - A_{\Delta t}^{-1} A_{\delta t}$ over all negative $\sigma$ for different values of $\alpha$ for a model decomposition with $T = 100$, 30 subintervals, one coarse time step per subinterval, and a refinement ratio of $10^4$ between the coarse and the fine grid. We see in this case that the estimate (3.39) is indeed quite accurate.
Fig. 6. Behavior of $\mu_{\alpha<0}(\sigma)$ as a function of $\alpha$, $T = 100$, $L = 30$, $\Delta T = \Delta t$, $\Delta t/\delta t = 10^{-4}$. The data for $\mu_{\max}(\alpha)$ has been generated by solving the generalized eigenvalue problem (3.20) using `eig` in MATLAB.

Remarks.

1. (Dependence on $\alpha$) Theorem 3.8 states that in order to guarantee convergence, one should make sure that the coarse time step $\Delta t$ is sufficiently small relative to $\alpha$. In that case, the method converges.

2. (Weak scalability) Note that the estimate (3.39) depends on the coarse time step $\Delta t$ but not explicitly on the number of subintervals $L$. One may then consider weak scalability, i.e., cases where the problem size per processor is fixed, under two different regimes: (i) keeping the subinterval length $\Delta T$ and refinement ratios $\Delta T/\Delta t$, $\Delta t/\delta t$ fixed, such that adding subintervals increases the overall time horizon $T = L\Delta T$; and (ii) keeping the time horizon $T$ fixed and refinement ratios $\Delta T/\Delta t$, $\Delta t/\delta t$ fixed, such that adding subintervals decreases their length $\Delta T = T/L$. In the first case, $\Delta t$ remains fixed, so the bound (3.39) remains bounded as $L \to \infty$. In the second case, $\Delta t \to 0$ as $L \to \infty$, so in fact (3.39) decreases to 0 as $L \to \infty$. Therefore, the method is weakly scalable under both regimes.

3. (Contraction rate for high and low frequencies) Let $\alpha > 0$ be fixed, and let $\rho(\sigma)$ be the spectral radius of $I - A_{\Delta t}^{-1}A_{\delta t}$ as a function of $\sigma$ given by (3.36). Then for $\Delta t/\delta t \geq 2$, an asymptotic expansion shows that we have

$$
\rho(\sigma) = \begin{cases} 
|\sigma|(|\Delta t - \delta t|) + O(|\sigma|^2) & \text{as } |\sigma| \to 0, \\
\frac{1}{\sigma^2} + O(|\sigma|^{-2}) & \text{as } |\sigma| \to \infty \text{ if } \Delta T = \Delta t, \\
\frac{1}{\sigma^3} + O(|\sigma|^{-3}) & \text{as } |\sigma| \to \infty \text{ if } \Delta T/\Delta t \geq 2.
\end{cases}
$$

In other words, the method reduces high and low frequency error modes very quickly, and the overall contraction rate is dominated by mid frequencies (where

---

On the contrary, strong scalability deals with cases where the total problem size is fixed.
“mid” depends on $\alpha, \Delta t$, etc.). This is also visible in Figure 5, where $\rho$ attains its maximum at $|\sigma| = O(1/\sqrt{\alpha})$ and decays quickly for both large and small $|\sigma|$.

Finally, we note that for the linear problem, it is possible to use Krylov acceleration to solve for the fixed point of (3.19), even when the spectral radius is greater than 1. However, the goal of this linear analysis is to use it as a tool for studying the asymptotic behavior of the nonlinear method (2.11); since a contractive fixed point map must have a Jacobian with spectral radius less than 1 at the fixed point, Theorem 3.8 shows which conditions are sufficient to ensure asymptotic convergence of the nonlinear ParaOpt method.

4. Numerical results. In the previous section, we have presented numerical examples related to the efficiency of our bounds with respect to $\sigma$ and $\alpha$. We now study in more detail the quality of our bounds with respect to the discretization parameters. We complete these experiments with a nonlinear example and a PDE example.

4.1. Linear scalar ODE: Sensitivity with respect to the discretization parameters. In this part, we consider the case where $\alpha = 1$, $\sigma = -16$, and $T = 1$ and investigate the dependence of the spectral radius of $I - A_{\Delta t}^{-1}A_{\delta t}$ when $L, \Delta t, \delta t$ vary.

We start with variations in $\Delta t$ and $\delta t$ and a fixed number of subintervals $L = 10$. In this way, we compute the spectral radius of $I - A_{\Delta t}^{-1}A_{\delta t}$ for three cases: first with a fixed $\Delta t = 10^{-4}$ and $\delta t = \frac{\Delta t}{2^k}, k = 1, \ldots, 15$; then with a fixed $\delta t = 10^{-2} \cdot 2^{-20}$ and $\Delta t = 2^{-k}, k = 0, \ldots, 20$; and finally with a fixed ratio $\frac{\delta t}{\Delta t} = 10^{-2}$ with $\Delta t = 2^k, k = 1, \ldots, 15$. The results are shown in Figure 7.

![Graph showing spectral radius of the preconditioned matrix for varying $\delta t$ (with fixed $\Delta t$), $\Delta t$ (with fixed $\delta t$), and $\Delta t$ (with fixed $\frac{\delta t}{\Delta t}$).](image)

**Fig. 7.** Spectral radius of the preconditioned matrix. Top left: varying $\delta t$ (with fixed $\Delta t$); top right: varying $\Delta t$ (with fixed $\delta t$); bottom: varying $\Delta t$ (with fixed $\frac{\delta t}{\Delta t}$).
Fig. 8. Spectral radius of the preconditioned matrix as a function of $L$. Left: fixed value of $T$ (with $T = 1$); right: $T = L\Delta T$.

In all cases, we observe a very good agreement between the estimate obtained in (3.36) and the true spectral radius. Note that the largest possible $\Delta t$ for this problem is when $\Delta t$ equals the length of the subinterval, i.e., when $\Delta t = \Delta T = 0$. For this $\Delta t$, the estimates (3.36) and (3.39) are very close to each other, because (3.39) is obtained from (3.36) by making $\Delta t$ as large as possible, i.e., by letting $\Delta t = \Delta T$.

We next study the scalability properties of ParaOpt. More precisely, we examine the behavior of the spectral radius of the preconditioned matrix when the number of subintervals $L$ varies. In order to fit with the paradigm of numerical efficiency, we set $\Delta T = \Delta t$ which corresponds somehow to a coarsening limit. We consider two cases: the first case uses a fixed value of $T$, namely $T = 1$, and the second case uses $T = L\Delta T$ for the fixed value of $\Delta T = 1$. The results are shown in Figure 8.

In both cases, we observe perfect scalability of ParaOpt, in the sense that the spectral radius is uniformly bounded with respect to the number of subintervals considered in the time parallelization.

4.2. A nonlinear example. We now consider a control problem associated with a nonlinear vectorial dynamics, namely the Lotka–Volterra system. The problem consists of minimizing the cost functional

$$J(c) = \frac{1}{2} |y(T) - y_{\text{target}}|^2 + \frac{\alpha}{2} \int_0^T |c(t)|^2 \, dt$$

with $y_{\text{target}} = (100, 20)^T$, subject to the Lotka–Volterra equations

$$\begin{align*}
\dot{y}_1 &= g(y) := a_1 y_1 - b_1 y_1 y_2 + c_1, \\
\dot{y}_2 &= \tilde{g}(y) := a_2 y_1 y_2 - b_2 y_2 + c_2
\end{align*}$$

(4.1)

with $a_1 = b_2 = 10$, $b_1 = a_2 = 0.2$ and initial conditions $y(0) = (20, 10)^T$. In this nonlinear setting, the computation of each component of $\mathcal{F}(Y, A)$ for given $Y$ and $A$ requires a series of independent iterative inner loops. In our test, these computations are carried out using a Newton method. As in section 3, the time discretization of (2.2) is performed with an implicit Euler scheme.

In a first test, we set $T = 1/3$ and $\alpha = 5 \times 10^{-2}$ and fix the fine time discretization step to $\delta t = T / N_0$, with $N_0 = 12 \cdot 10^{-5}$. In Figure 9, we show the rate of convergence of ParaOpt for $L = 10$ and various values of the ratio $r = \frac{\Delta t}{\Delta T}$. Here, the error is defined as the maximum difference between the interface state and adjoint values obtained...
from a converged fine grid solution, and the interface values obtained at each inexact Newton iteration by ParaOpt.

As can be expected when using a Newton method, we observe that quadratic convergence is obtained in the case \( r = 1 \). When \( r \) becomes smaller, the preconditioning becomes a coarser approximation of the exact Jacobian, and thus convergence becomes a bit slower.

In our experiments, we observed that the initial guess plays a significant role in the convergence of the method. This follows from the fact that ParaOpt is an exact (if \( \Delta t = \delta t \)) or approximate (otherwise) Newton method. The initial guess we consider is \( c(t) = 1 \), \( y(T) = (1 - T/\ell)y_0 + T/\ell y_{\text{target}} \), and \( \lambda(T) = (1, 1)^T \). While for \( T = 1/3 \) we observe convergence for all \( L \), if we increase \( T \) to \( T = 1 \), we do not observe convergence any more for \( L < 10 \); in fact, without decomposing the time domain, the sequential version of our solver with \( L = 1 \) does not converge, even if we use the exact Jacobian without the coarse approximation. This shows that using a time-domain decomposition actually helps in solving the nonlinear problem, a phenomenon already observed for a different time parallelization method in [48]. These convergence problems we observed are also related to the existence of multiple solutions. Indeed, if we coarsen the outer iteration by replacing the Newton iteration with a Gauss–Newton iteration, i.e., by removing the second–order derivatives of \( g \) and \( \tilde{g} \) in Newton’s iterative formula, we obtain another solution, as illustrated in Figure 10 on the left for \( T = 1 \) and \( r = 1 \). For both solutions, we observe that the eigenvalues associated with the linearized dynamics

\[
\begin{align*}
\delta \dot{y}_1 &= a_1 \delta y_1 - b_1 \delta y_1 y_2 - b_1 y_1 \delta y_2 + \delta c_1, \\
\delta \dot{y}_2 &= a_2 \delta y_1 y_2 + a_2 y_1 \delta y_2 - b_2 \delta y_2 + \delta c_2
\end{align*}
\]

in a neighborhood of the local minima remain strictly positive along the trajectories, in contrast to the situation analyzed in section 3. Their values are presented in Figure 10 on the right.

We next test the numerical efficiency of our algorithm. The example we consider corresponds to the last curve of Figure 9, i.e., \( T = 1/3 \) and \( r = 10^{-4} \), except that we use various values of \( L \in \{1, 3, 6, 12, 24\} \) using the corresponding number of processors.
Fig. 10. Left: two local minima of the cost functional $J$, obtained with Newton (plain line) and Gauss–Newton (dashed line) in the outer loop, for $T = 1$. The cost functional values are $J \approx 1064.84$ and $J \approx 15.74$. The green cross and the red circle indicate $y_0$ and $y_{\text{target}}$. Right: (real) eigenvalues associated with the linearized dynamics in a neighborhood of the local minima obtained with Newton (top) and Gauss–Newton (bottom).

Table 2

<table>
<thead>
<tr>
<th>$L$</th>
<th>Newton its.</th>
<th>$T_{\text{cpu}}$</th>
<th>Parallel computing time</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14</td>
<td>777.53</td>
<td>777.42</td>
<td>1.00</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>172.13</td>
<td>167.36</td>
<td>4.52</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>82.10</td>
<td>79.67</td>
<td>9.47</td>
</tr>
<tr>
<td>12</td>
<td>9</td>
<td>43.31</td>
<td>42.49</td>
<td>17.95</td>
</tr>
<tr>
<td>24</td>
<td>9</td>
<td>25.75</td>
<td>24.74</td>
<td>30.20</td>
</tr>
</tbody>
</table>

We execute our code in parallel on workers of a parallel pool, using the MATLAB Parallel Processing Toolbox on a 24-core machine that is part of the SciBlade cluster at Hong Kong Baptist University. The results are presented in Table 2, where we also indicate the total parallel computing time without communication, as well as the number of outer Newton iterations required for convergence to a tolerance of $10^{-13}$.

We observe that our cluster enables us to get very good scalability; the total computing time is roughly divided by two when the number of processors is doubled. Though not reported in the table, we have observed that even in the case $L = 1$, i.e., without parallelization, ParaOpt outperforms the Newton method (777.53 s versus 865.76 s in our test).

To see how this compares with speedup ratios that can be expected from more classical approaches, we run Parareal on the initial value problem (4.1) with the same initial conditions and no control, i.e., $c_1 = c_2 = 0$. For $L = 3, 6, 12$, and 24 subintervals and a tolerance of $10^{-13}$, Parareal requires $K = 3, 6, 8$, and 13 iterations to converge. (For a more generous tolerance of $10^{-8}$, Parareal requires $K = 3, 6, 6$, and 7 iterations.) Since the speedup obtained by Parareal cannot exceed $L/K$, the maximum speedup that can be obtained if Parareal is used as a subroutine for forward and backward sweeps does not exceed 4 for our problem. Note that this result is specific to the nondiffusive character of the considered equation. This speedup would change if the constraint type changed to parabolic; see [44, Chapter 5].

4.3. A PDE example. We finally consider a control problem involving the heat equation. More precisely, (2.1) is replaced by
PARAOPT: PARAREAL FOR OPTIMALITY SYSTEMS

\[ \partial_t y - \Delta y = Bc , \]

where the unknown \( y = y(x,t) \) is defined on \( \Omega = [0,1] \) with periodic boundary conditions and on \([0,T]\) with \( T = 10^{-2} \). Initial and target states are

\[
y_{\text{init}} = \exp(-100(x-1/2)^2), \quad y_{\text{target}} = \frac{1}{2} \exp(-100(x-1/4)^2) + \frac{1}{2} \exp(-100(x-3/4)^2). \]

The operator \( B \) is the indicator function of a subinterval \( \Omega_c \) of \( \Omega \); in our case, \( \Omega_c = [1/3, 2/3] \). We also set \( \alpha = 10^{-4} \). The corresponding solution is shown in Figure 11.

We use a finite difference scheme with 50 grid points for the spatial discretization. As in the previous subsection, an implicit Euler scheme is used for the time discretization, and we consider a parallelization involving \( L = 10 \) subintervals, with \( \delta t = 10^{-7} \) and \( \delta t = 10^{-9} \) so that the rate of convergence of the method can be tested for various values of \( r = \frac{\delta t}{\Delta t} \). For \( \alpha = 10^{-4} \), the evolution of the error along the iterations is shown in Figure 12. Here, the error is defined as the maximum difference between the iterates and the reference discrete solution, evaluated at subinterval interfaces.

Observe also that the convergence curves corresponding to \( r = 10^{-1} \) and \( r = 10^{-2} \) on the left panel look nearly identical to the curves for \( r = 10^{-3} \) and \( r = 10^{-4} \) on the right panel. This is because they correspond to the same values of \( \Delta t \), namely \( \Delta t = 10^{-6} \) and \( \Delta t = 10^{-5} \). This behavior is consistent with Theorem 3.8, where the convergence estimate depends only on \( \Delta t \), rather than on the ratio \( \frac{\delta t}{\Delta t} \). Cases of divergence can also be observed, in particular for \( T = 1 \) and small values of \( \alpha \) and \( r \), as shown in Figure 13.

Of course, one can envisage using different spatial discretizations for the coarse and fine propagators; this may provide additional speedup, provided suitable restriction and prolongation operators are used to communicate between the two discretizations. This will be the subject of investigation in a future paper.
Fig. 12. Convergence of the method for various values of the ratio $r = \frac{\delta t}{\Delta t}$. Left: $\delta t = 10^{-7}$; right: $\delta t = 10^{-9}$.

Fig. 13. Top left: spectral radius of the preconditioned matrix as a function of $\alpha$, with $\delta t = 10^{-5}$ and $\Delta t = \Delta T = 10^{-1}$. Top right: spectral radius of the preconditioned matrix as a function of $\Delta t/\Delta T$, with $\delta t = 10^{-8}$ and $\alpha = 10^{-4}$. Bottom left: spectral radius of the preconditioned matrix as a function of $\alpha$ and $\Delta t/\Delta T$, with $\delta t = 10^{-7}$. Bottom right: estimate (3.39) as a function of $\alpha$ and $\Delta t/\Delta T$.

5. Conclusions. We introduced a new time parallel algorithm we call ParaOpt for time-dependent optimal control problems. Instead of applying Parareal to solve separately the forward and backward equations as they appear in an optimization loop, we propose in ParaOpt to partition the coupled forward-backward problem directly in time and to use a Parareal-like iteration to incorporate a coarse correction when solving this coupled problem. We analyzed the convergence properties of ParaOpt and
proved in the linear diffusive case that its convergence is independent of the number of subintervals in time and thus is scalable. We also tested ParaOpt on scalar linear optimal control problems, a nonlinear nondiffusive optimal control problem involving the Lotka–Volterra system, and on a control problem governed by the heat equation. A small scale parallel implementation of the Lotka–Volterra case also showed scalability of ParaOpt for this nonlinear problem.

Our ongoing work consists of analyzing the algorithm for nondiffusive problems. Also, for problems with large state spaces, e.g., for discretized PDEs in three spatial dimensions, the approximate Jacobian $\mathcal{J}_{\mathcal{G}}$ in (2.12) may become too large to solve by direct methods. Thus, we are currently working on designing efficient preconditioners for solving such systems iteratively. Finally, we are currently studying ParaOpt by applying it to realistic problems from applications, in order to better understand its behavior in such complex cases.

Appendix A. Proof of inequality (3.34). Our goal is to prove the following lemma, which is needed for the proof of Lemma 3.3.

**Lemma A.1.** For every $k > 0$ and $0 < x \leq k$, we have

\[(1 + x)^{k/x} > k \left( \frac{2 + x}{1 + x} \right) - 1. \tag{A.1} \]

First, we need the following property of logarithmic functions.

**Lemma A.2.** For any $x > 0$, we have

\[\ln(1 + x) \geq \frac{x}{x + 1} + \frac{1}{2} \left( \frac{x}{x + 1} \right)^2. \]

**Proof.** Let $u = \frac{x}{x + 1} < 1$. Then

\[
\ln(1 + x) = -\ln \left( \frac{1}{1 + x} \right) = -\ln(1 - u) = u + \frac{u^2}{2} + \frac{u^3}{3} + \cdots \geq u + \frac{u^2}{2}.
\]

The conclusion now follows. \qed

**Lemma A.1.** Let $g$ and $h$ denote the left- and right-hand sides of (A.1), respectively. We consider two cases, namely when $0 < k \leq 1$ and when $k > 1$. When $k \leq 1$, we have

\[h(x) \leq \frac{2 + x}{1 + x} - 1 = \frac{1}{1 + x} < 1 < (1 + x)^{k/x} = g(x).\]

For the case $k > 1$, we will show that $g(k) > h(k)$ and $g'(x) - h'(x) < 0$ for $0 < x < k$, which together imply that $g(x) > h(x)$ for all $0 < x \leq k$. The first assertion follows from the fact that

\[g(k) - h(k) = 1 + k - k \cdot \frac{k + 2}{k + 1} + 1 = 2 - \frac{k}{k + 1} > 0.\]

To prove the second part, we note that
\[ g'(x) = (1 + x)^{k/x} \left[ -\frac{k}{x^2} \ln(1 + x) + \frac{k}{x(1 + x)} \right] \]
\[ = -\frac{k}{x^2} (1 + x)^{k/x-1} [ (1 + x) \ln(1 + x) - x ] \]
\[ < -\frac{k}{x^2} (1 + x)^{k/x-1} \cdot \frac{x^2}{2(x+1)} = -\frac{k}{2} (1 + x)^{k/x-2} < 0, \]
\[ h'(x) = -\frac{k}{(1 + x)^2} < 0. \]

Therefore, we have
\[ g'(x) - h'(x) < -\frac{k}{(1 + x)^2} \left[ \frac{1}{2} (1 + x)^{k/x} - 1 \right] \leq -\frac{k}{(1 + x)^2} \left[ \frac{1 + k}{2} - 1 \right] < 0. \]

Thus, \( g(x) > h(x) \) for all \( 0 < x < k \), as required.

\[ \square \]

Acknowledgment. We thank the anonymous referees for their valuable suggestions, which greatly improved our paper.

REFERENCES


