Mathematical Modelling and Analysis
Volume 23, Issue 2, 287–308, 2018
https://doi.org/10.3846/mma.2018.018

http://mma.vgtu.lt
ISSN: 1392-6292
eISSN: 1648-3510

Parallel Solution Methods and Preconditioners for Evolution Equations

Owe Axelsson[a,b], **Maya Neytcheva**[b] and **Zhao-Zheng Liang**[c,b]

[a] *Institute of Geonics, Czech Academy of Sciences*
Ostrava, Czech Republic
[b] *Department of Information Technology, Uppsala University*
Uppsala, Sweden
[c] *School of Mathematics and Statistics, Lanzhou University*
Lanzhou, China
E-mail: owe.axelsson@it.uu.se
E-mail(*corresp.*): maya.neytcheva@it.uu.se
E-mail: liangzhzh2014@lzu.edu.cn

Received September 9, 2017; revised March 4, 2018; accepted March 6, 2018

Abstract. The recent development of the high performance computer platforms shows a clear trend towards heterogeneity and hierarchy. In order to utilize the computational power, particular attention must be paid to finding new algorithms or adjust existing ones so that they better match the HPC computer architecture.

In this work we consider an alternative to classical time-stepping methods based on use of time-harmonic properties and discuss solution approaches that allow efficient utilization of modern HPC resources.

The method in focus is based on a truncated Fourier expansion of the solution of an evolutionary problem. The analysis is done for linear equations and it is remarked on the possibility to use two- or multilevel mesh methods for nonlinear problems, which can enable further, even higher degree of parallelization.

The arising block matrix system to be solved admits a two-by-two block form with square blocks, for which a very efficient preconditioner exists. It leads to tight eigenvalue bounds for the preconditioned matrix and, hence, to a very fast convergence of a preconditioned Krylov subspace or iterative refinement method. The analytical background is shown as well as some illustrating numerical examples.

Keywords: parallel solution, evolution equation, preconditioning, PDE-constrained optimization.

AMS Subject Classification: 65F08; 65N22; 35F10.

1 Introduction

Traditional solution methods for time-dependent partial differential equations are based on time-stepping methods. These methods can be explicit or implicit, however they are by nature sequential in time, i.e., need many time steps to be executed one after another, which make these methods slow, thus, demanding much computing time. Parallelism might occur only within each timestep. For reasons of numerical stability, explicit methods may require the use of very small time-steps, which, for real life problems, may require infeasibly many time steps. Implicit methods are stable but the timesteps should still be chosen small enough, to achieve a sufficiently small time-discretization error. Alternatively, higher order stable methods can be used which, however, add even further to the computational complexity.

In the time-stepping framework parallelism is achieved within each timestep. For implicit methods, in each time interval one has to solve a system of equations, usually of elliptic type, thereby applying known solution techniques, with enough degree of parallelism, such as some multilevel or multigrid techniques.

The search for parallelism in time has a long history. As an example, we refer to [33], where the parallelism is sought between the steps of suitably constructed Diagonally Implicit Runge-Kutta (DIRK) methods, originally constructed for solving stiff ordinary differential equations (cf. e.g., [1]). Still, the nature of the computations remains serial and within the time-stepping framework. However, there exist alternatives to time-stepping methods, where one still needs to solve systems, similar to those arising in the implicit methods, but in parallel.

In many practically important applications, such as in optimal control, the control function is periodic, for instance, an alternating current in electromagnetic problems. Then the equation becomes time-harmonic and the solution can be approximated with a truncated Fourier series expansion. Due to the orthogonality of the trigonometric functions, for linear problems the computation of the Fourier coefficients separate and one can compute the solution for each period fully in parallel. Hence, the solution process is perfectly parallelizable across the different frequencies. Due to the large size of the discretized systems one must use iterative solution methods, see e.g. [4], [5], [32], [34]. The preconditioner used should then also be well parallelizable.

For each frequency, a system on two-by-two or four-by-four block matrix form arises. In a series of publications ([6, 7, 8, 10, 12]) it has been demonstrated that for such problems a very efficient preconditioner, based on the PREconditioned Square matrix Block (PRESB) method, exists and leads to very tight eigenvalue bounds and, hence, few iterations. This holds uniformly with respect to mesh size, control cost and other regularization parameters as well as to the problem parameters. The method has been shown to outperform other published methods for the problems considered. We argue further, that the structure of the PRESB method allows for additional levels of parallelization, achieved by applying well-developed and efficient computational toolboxes for the basic operator equations arising in the PRESB method, such as the Algebraic Multigrid (AMG) method, provided via publicly available scientific computing libraries. To illustrate the idea we consider an optimal control problem, constrained by an evolutionary partial differential equation.

For nonlinear problems one can use ideas, presented in [3, 9, 37] and utilize a two-grid or multi-level method, where the nonlinear equation is solved on a coarse grid, then the solution is interpolated to the fine grid and a corrected solution of the linearized Newton type equation is computed just once, which normally suffices to get a solution with an error of the same order as the fine mesh discretization error. Such approaches can save much computational labor.

The idea of using truncated Fourier series expansions is introduced in Section 2. We present an introductory example of an optimal control problem for an evolution equation in Section 3. The description of the solution method and the preconditioner is found in Section 4. Section 5 comprises a short presentation of the suggested approach for nonlinear problems. Some discussion of the computational and communication costs of the methods is given in Section 6. An illustrating example for the heat equation with numerical test results are presented in Section 7. We also shortly refer to an application of the method for solving eddy current electromagnetic problems, as presented in [10].

2 Truncated Fourier series expansions for linear evolution equations

Consider an evolution equation

$$\frac{\partial u}{\partial t} + \mathcal{L}u = f, \quad u = u(x,t), \quad 0 < t < T, \quad x \in \Omega \subset \mathbb{R}^d, \quad d = 1,\ 2,\ 3, \quad (2.1)$$

where $\mathcal{L}$ is a time-independent linear and coercive elliptic type differential operator with boundary and initial conditions $u(x,t) = 0$ on $\partial\Omega$, $u(x,0) = u_0(x)$.

2.1 Truncated Fourier expansion

As is well known, every sufficiently smooth periodic function, say $F(t)$, that is periodic on $-T \le t \le T$, has a Fourier series expansion

$$F(t) = \sum_{k=0}^{\infty} a_k \cos\left(\frac{k\pi}{T}t\right) + \sum_{k=0}^{\infty} b_k \sin\left(\frac{k\pi}{T}t\right). \quad (2.2)$$

Utilizing the Fourier expansion representation, as we show in the sequel, enables us to strongly increase the degree of parallelism of the underlying solution methods. As the problem to solve is, in general, not periodic, we explore the option to extend it and make it periodic.

First we recall that defining $F(t)$ on $-T \le t \le T$ to be a periodic extension of some given function $f(t)$, defined on $0 \le t \le T$ can be done in many different ways. There are two periodic extensions of $f(t)$ that turn out to be particularly useful, namely,

$F(t) = f(t), 0 \le t \le T$,
$F(t)$ is even, i.e., $F(-t) = f(t)$,
F has period $2T$,
$F(t) = a_0 + \sum\limits_{k=1}^{\infty} a_k \cos\left(\frac{k\pi t}{T}\right)$,
'even periodic extension'

$F(t) = f(t), 0 \le t \le T$,
$F(t)$ is odd, i.e., $F(-t) = -f(t)$,
F has period $2T$,
$F(t) = \sum\limits_{k=1}^{\infty} b_k \sin\left(\frac{k\pi t}{T}\right)$,
'odd periodic extension'

We note, that when F is even, it has a cosine Fourier expansion and when F is odd, it has a sine Fourier expansion. Figure 1 illustrates an even extension of $f(x)$, that is, the solution on the interval $[T, 2T]$ is the mirror of the solution on $[0, T]$.

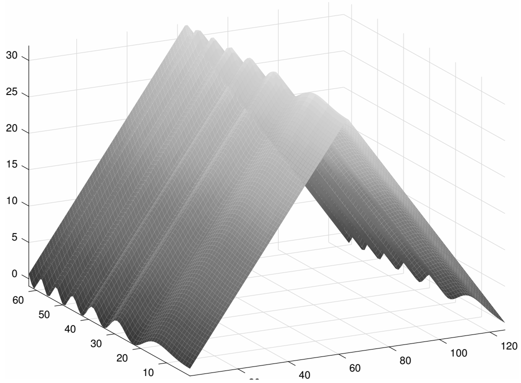


Figure 1. A symmetrically extended function

Following the above lines, in order to get a periodic solution we extend the problem (2.1) symmetrically to the time interval $(0, 2T]$, where $f(x, 2T - t) = -f(x, t)$, $0 < t \leq T$. Hence, it holds also that $u(x, 2T - t) = u(x, t)$, $0 < t \leq T$. The differential equation then becomes an evolution equation with periodic conditions,

$$\begin{aligned} \frac{\partial u}{\partial t} + \mathcal{L}u &= f, \quad 0 < t < 2T, \quad x \in \Omega, \\ u(x,t) &= 0, \quad x \in \partial\Omega \text{ and } u(x,0) = u(x,2T). \end{aligned} \tag{2.3}$$

This makes it possible to use a truncated Fourier series,

$$u(x,t) = u_0(x) + \sum_{k=1}^{\infty} u_k(x) \cos(\omega_k t), \tag{2.4}$$

where $\omega_k = k\frac{\pi}{T}$, $k = 0, 1, \ldots, N$ are the angular frequencies and $u_k(x)$ are the Fourier coefficients at $x \in \Omega$.

The expansions (2.2) and (2.4) represent the *trigonometric form* of a Fourier series. We consider, however, the more general *complex exponential form* of a Fourier series. The expression $a_k \cos(k\theta) + b_k \cos(k\theta)$ can be stated using exponentials and we can rewrite (2.2) as

$$F(t) = \sum_{k=0}^{\infty} c_k \, e^{\mathrm{i}\,\omega_k t} + \sum_{k=0}^{\infty} c_k^* \, e^{-\mathrm{i}\omega_k t},$$

where $c_k = \frac{1}{2}(a_k - \mathrm{i}\, b_k)$ and $c_k^* = \frac{1}{2}(a_k + \mathrm{i}\, b_k)$ and i is the imaginary unit. The complex formulation of a Fourier series is considered to be precursor of the Fourier transform which attempts to apply Fourier analysis for non-periodic functions. The complex form is widely used by engineers, for example in circuit and control theory.

Let assume that the solution of (2.3) is of even more general form, namely,

$$u(x,t)=\sum_{k=0}^{N} u_k^c(x)\cos(\omega_k t)+\mathrm{i}u_k^s(x)\sin(\omega_k t). \tag{2.5}$$

Substituting (2.5) in the differential equation (2.3), we obtain

$$\sum_{k=0}^{N}\Big(-\omega_k u_k^c(x)\sin(\omega_k t)+\mathrm{i}\omega_k u_k^s(x)\cos(\omega_k t)\Big) \\ +\sum_{k=0}^{N}\Big(\mathcal{L}u_k^c(x)\cos(\omega_k t)+\mathrm{i}\mathcal{L}u_k^s(x)\sin(\omega_k t)\Big)=f.$$

A multiplication with $\cos(\omega_k t)$ respectively with $\sin(\omega_k t)$, and integration over the time interval $(0,2T)$, thereby using the orthogonality of the trigonometric functions, results in the equations

$$\left\{\begin{array}{rcl} \mathrm{i}\omega_k u_k^s(x)+\mathcal{L}u_k^c(x) & = & f_k^c(x):=\frac{1}{T}\int_0^{2T} f(x,s)\cos(\omega_k,s)ds, \\ -\omega_k u_k^c(x)+\mathrm{i}\mathcal{L}u_k^s(x) & = & f_k^s(x):=\frac{1}{T}\int_0^{2T} f(x,s)\sin(\omega_k,s)ds=0. \end{array}\right. \tag{2.6}$$

Here we have used the elementary relations

$$\frac{1}{T}\int_0^{2T}\cos^2(\omega_k s)ds=\frac{1}{T}\int_0^{2T}\sin^2(\omega_k s)ds=1$$

and the symmetry of $f(x,t)=f(x,2T-t)$, which gives $f_k^s(x)=0$.

A proper choice of N could be made by trial and error. In some problems one can instead utilize classical truncation error expressions and estimate the solution error via the truncation error. We refer to [21,23] for such an estimate showing that the error in the Fourier expansion decreases at least as $O(1/\sqrt{N})$, $N\to\infty$. In practice, the solution is often much smoother and can be represented by a few-term Fourier expansion, so it is not needed to choose a large value of N. However, as shown below, the bigger N, the more parallelism is enabled in the solution method.

2.2 Galerkin variational FEM

Let $u_k^c\in U$, $u_k^s\in V$, where U and V are appropriate subspaces of $H_0(\Omega)$ and let U_h and V_h be some finite-dimensional (finite element) subspaces. Let M be the corresponding Galerkin mass matrix and L the corresponding stiffness matrix, for the inner product in $L_2(\Omega)$. Then, after multiplying the equations (2.6) with the basis functions in U_h and V_h and forming the finite element (FEM) variational formulation of (2.6), the equations take the algebraic form

$$\left\{\begin{array}{rcl} L\mathbf{u}_k^c+\mathrm{i}\omega_k M\mathbf{u}_k^s & = & \mathbf{f}_k^c, \\ -\omega_k M\mathbf{u}_k^c+\mathrm{i}L\mathbf{u}_k^s(x) & = & \mathbf{0}, \end{array}\right. \tag{2.7}$$

where $\mathbf{u}_k^c, \mathbf{u}_k^s$ are the discrete FE solution vectors and $\mathbf{f}_k^c$ - the vector representation of the source function corresponding to ω_k and the chosen basis functions.

For $k = 0$ we get the solutions $L\mathbf{u}_0 = f_k^c$, $L\mathbf{v}_0 = 0$, i.e., $\mathbf{v}_0 = 0$. We multiply the second equation in (2.7) by the complex unit number i to get the block matrix system,

$$\begin{bmatrix} L & \mathrm{i}\omega_k M \\ -\mathrm{i}\omega_k M & -L \end{bmatrix} \begin{bmatrix} \mathbf{u}_k^c \\ \mathbf{u}_k^s \end{bmatrix} = \begin{bmatrix} \mathbf{f}_k^c \\ 0 \end{bmatrix}. \tag{2.8}$$

Clearly, the system (2.8) can be further reduced to the fourth order problem,

$$(LM^{-1}L + \omega_k^2 M)\mathbf{u}_k^c = LM^{-1}\mathbf{f}_k^c. \tag{2.9}$$

However, to avoid working with fourth order operators we keep the two-by-two block form in (2.8). Note that the matrix in (2.8) is indefinite and Hermitian. If one prefers to use only real arithmetics, the two-by-two block complex matrix can be reformulated as a four-by-four real-valued block matrix, see e.g. [10] for an example of this.

3 An optimal control problem with an evolution state equation

Optimal control problems with a PDE as state equation have been studied in many publications. Such publications, related to this paper, are for instance [17, 18, 22, 23, 29] and the previously mentioned [6, 7, 8, 12]. Here we emphasize the case when the constraint is an evolutionary equation.

In general, initially before the exponential decay caused by the elliptic nature of the problem has had any significant damping influence, the solution to an evolution equation shows a less smooth behaviour. When the diffusion coefficient in $\mathcal{L}$ is relatively small, this may occur particularly late in time. As $t \to \infty$, eventually the solution becomes close to the stationary solution, $\mathcal{L}u_\infty = f$. In some problems, such as electrical machineries, the primary aim is to find the stationary solution to the time-dependent problem. Then one aim is to find an initial value function which makes the convergence to the stationary solution smooth for all times, $0 < t < T$. In addition, we want to find a source control function such that the solution for $t = T$ is close to some given, target solution $u_d(x)$.

However, we pose here the task to find a solution $u(x,t)$ of the evolution equation for all t,

$$\begin{aligned} &\frac{\partial u}{\partial t} + \mathcal{L}u = v(x,t), \quad (x,t) \in \Omega \times (0,T], \Omega \subset \mathbb{R}^d, d = 1,2,3, \\ &u(x,t) = 0, \text{ on } \partial\Omega, \\ &u(x,0) = u_0(x) \end{aligned}$$

and a control function $v(x,t)$ such that $u(x,t)$ is as close as possible to a desired solution $u_d(x,t)$. The boundary conditions are chosen to be homogeneous only for simplicity, Ω is a given bounded Lipschitz domain and $\mathcal{L}$ is an elliptic operator, i.e. satisfies a coercivity condition, $(\mathcal{L}u, w) \geq \alpha(u, w)$, $\alpha > 0$, for all $w \in H^1(\Omega)$. For example, $\mathcal{L}$ can be the convection-diffusion operator, $\mathcal{L}u = -\nu\Delta u + \mathbf{w}\cdot\nabla u + cu$, where $\nu > 0$ and $c - \frac{1}{2}\nabla\cdot\mathbf{w} \geq 0$. The solution is controlled by the source function $v \in L^2(\Omega) \times (0, T]$ which, for simplicity we assume to be distributed in the whole domain and also to satisfy the homogeneous boundary conditions.

Such optimal control problems must be regularized to obtain a well-posed problem. For this purpose, we use the simplest Tikhonov regularization form. Let λ be a Lagrange multiplier to handle the constraint equation, $\frac{\partial u}{\partial t} + \mathcal{L}u = v$. The optimal control problem can then be formulated as a saddle type functional,

$$\min_{u,v} \max_{\lambda} \mathcal{J}(u, v, \lambda),$$

where the Lagrangian functional $\mathcal{J}(u, v, \lambda)$ equals

$$\begin{aligned} \mathcal{J}(u,v,\lambda) &= \frac{1}{2}\int_0^T \int_\Omega (u(x,s) - u_d(x,s))^2 dxds \\ &+ \int_0^T \int_\Omega \lambda \left(\frac{\partial u}{\partial t} + \mathcal{L}u - v\right) dxds + \frac{1}{2}\beta \int_0^T \int_\Omega v^2(x,s)dxds\,. \end{aligned} \tag{3.1}$$

Here $\beta > 0$ is the regularization parameter for the cost of the control function v. The first order necessary conditions, $\nabla_{u,v,\lambda}\ \mathcal{J}(u,v,\lambda) = 0$, are also sufficient for the existence of a solution. Here $\frac{\partial \mathcal{J}}{\partial v} = 0$ leads to the simple relation, $-\lambda + \beta v = 0$, which enables us to reduce the arising linear system. Hence, substituting $\lambda = \beta v$ in (3.1) gives

$$\begin{aligned} \widehat{\mathcal{J}}(u,v) := \mathcal{L}(u,v,\beta v) &= \frac{1}{2}\int_0^T \int_\Omega (u(x,s) - u_d(x,s))^2 dxds \\ + \beta \int_0^T \int_\Omega v \left(\frac{\partial u}{\partial s} + \mathcal{L}u\right) dxds &- \frac{1}{2}\beta \int_0^T \int_\Omega v^2(x,s)ds \\ = \frac{1}{2}\int_0^T \int_\Omega (u(x,s) - u_d(x,s))^2 dxds &+ \beta \int_0^T \int_\Omega (-\frac{\partial v}{\partial s} + \mathcal{L}^* v)u\, dxds \\ - \frac{1}{2}\beta \int_0^T \int_\Omega v^2(x,s)dxds, \end{aligned}$$

where $\mathcal{L}^*$ is the adjoint operator to $\mathcal{L}$ and where we have made use of the homogeneous boundary conditions. The first order conditions give now

$$\left\{ \begin{array}{c} u + \beta\left(-\frac{\partial v}{\partial t} + \mathcal{L}^* v\right) = u_d, \\ \beta\left(\frac{\partial u}{\partial t} + \mathcal{L}u\right) - \beta v = 0, \end{array} \right. \quad \text{or} \quad \left\{ \begin{array}{rcl} u + \sqrt{\beta}\left(-\frac{\partial \widetilde{v}}{\partial t} + \mathcal{L}^*\ \widetilde{v}\right) &=& u_d, \\ -\sqrt{\beta}\left(\frac{\partial u}{\partial t} + \mathcal{L}u\right) + \widetilde{v} &=& 0, \end{array} \right. \tag{3.2}$$

where $\widetilde{v} = \sqrt{\beta}v$. Here u and v satisfy the given homogeneous boundary conditions but there are no initial conditions prescribed.

Choosing a proper set of finite element basis functions and multiplying the equations in (3.2)(right) with these basis functions as test functions, we obtain the semi-discrete (FEM) formulation, corresponding to (3.2):

$$\begin{cases} M\mathbf{u}(t)+\sqrt{\beta}\left(-\dfrac{\partial}{\partial t}M\,\tilde{\mathbf{v}}(t)+L^*\,\tilde{\mathbf{v}}(t)\right) &= \mathbf{u}_d(t),\\ -\sqrt{\beta}\left(\dfrac{\partial}{\partial t}M\mathbf{u}(t)+L\mathbf{u}(t)\right)+M\,\tilde{\mathbf{v}}(t) &= 0. \end{cases} \tag{3.3}$$

Here M is the mass matrix corresponding to the basis functions and the L_2-inner product, L is the stiffness matrix corresponding to the operator $\mathcal{L}$, and $\mathbf{u}(t)$, $\mathbf{u}_d(t)$, $\tilde{\mathbf{v}}(t)$ denote the finite element vectors, corresponding to the interior node points in Ω_h, the discrete finite element mesh.

In the sequel, dropping for simplicity the $\tilde{}$ sign, we next approximate the time-dependent vectors $\mathbf{u}(t)$, $\mathbf{u}_d(t)$, $\tilde{\mathbf{v}}(t)$ with truncated Fourier series expansions,

$$\sum_{k=0}^{N}\begin{bmatrix}\mathbf{u}_k^c+\mathrm{i}\mathbf{u}_k^s\\ \mathbf{v}_k^c+\mathrm{i}\mathbf{v}_k^s\end{bmatrix}e^{-i\omega_k t},$$

where ω_k are the frequencies ($\omega_k = k\pi/T$, $k=1,2,\ldots,N$) and where $\begin{bmatrix}\mathbf{u}(t)\\ \mathbf{v}(t)\end{bmatrix}$ has its real-valued part equal to $\sum\limits_{k=1}^{N}\left(\begin{bmatrix}\mathbf{u}_k^c\\ \mathbf{v}_k^c\end{bmatrix}\cos(\omega_k^s)+\begin{bmatrix}\mathbf{u}_k^s\\ \mathbf{v}_k^s\end{bmatrix}\sin(\omega_k^s)\right)$ and its imaginary part - equal to $\sum\limits_{k=1}^{N}\left(\begin{bmatrix}\mathbf{u}_k^s\\ \mathbf{v}_k^s\end{bmatrix}\cos(\omega_k^s)-\begin{bmatrix}\mathbf{u}_k^c\\ \mathbf{v}_k^c\end{bmatrix}\sin(\omega_k^s)\right)$.

Due to the orthogonality of the trigonometric functions, for linear problems, as we consider here, the computations of the different Fourier coefficients separate, so it suffices to consider the resulting equations for just one index k. From (3.3), it follows then,

$$\begin{cases} M(\mathbf{u}_k^c+\mathrm{i}\mathbf{u}_k^s)+\sqrt{\beta}\,(\mathrm{i}\omega_k M(\mathbf{v}_k^c+\mathrm{i}\mathbf{v}_k^s)+L^*(\mathbf{v}_k^c+\mathrm{i}\mathbf{v}_k^s)) &= \mathbf{u}_{d,k}^c+\mathrm{i}\mathbf{u}_{d,k}^s,\\ \sqrt{\beta}\,(\mathrm{i}\omega_k M(\mathbf{u}_k^c+\mathrm{i}\mathbf{u}_k^s)-L(\mathbf{u}_k^c+\mathrm{i}\mathbf{u}_k^s))+M(\mathbf{v}_k^c+\mathrm{i}\mathbf{v}_k^s) &= \mathbf{0}. \end{cases}$$

Here we separate the real and the imaginary parts for each equation, to get

$$\begin{cases} \begin{bmatrix}M & 0\\ 0 & M\end{bmatrix}\begin{bmatrix}\mathbf{u}_k^c\\ \mathbf{u}_k^s\end{bmatrix}+\sqrt{\beta}\omega_k\begin{bmatrix}0 & -M\\ M & 0\end{bmatrix}\begin{bmatrix}\mathbf{v}_k^c\\ \mathbf{v}_k^s\end{bmatrix}+\sqrt{\beta}\begin{bmatrix}L^* & 0\\ 0 & L^*\end{bmatrix}\begin{bmatrix}\mathbf{v}_k^c\\ \mathbf{v}_k^s\end{bmatrix}=\begin{bmatrix}\mathbf{u}_{d,k}^c\\ \mathbf{u}_{d,k}^s\end{bmatrix},\\ \sqrt{\beta}\omega_k\begin{bmatrix}0 & -M\\ M & 0\end{bmatrix}\begin{bmatrix}u_k^c\\ u_k^s\end{bmatrix}-\sqrt{\beta}\begin{bmatrix}L & 0\\ 0 & L\end{bmatrix}\begin{bmatrix}u_k^c\\ u_k^s\end{bmatrix}+\begin{bmatrix}M & 0\\ 0 & M\end{bmatrix}\begin{bmatrix}\mathbf{v}_k^c\\ \mathbf{v}_k^s\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\end{bmatrix} \end{cases}$$

or, in block matrix form,

$$\begin{bmatrix} \begin{bmatrix}M & 0\\ 0 & M\end{bmatrix} & \sqrt{\beta}\begin{bmatrix}L^* & -\omega_k M\\ \omega_k M & L^*\end{bmatrix}\\ -\sqrt{\beta}\begin{bmatrix}L & \omega_k M\\ -\omega_k M & L\end{bmatrix} & \begin{bmatrix}M & 0\\ 0 & M\end{bmatrix} \end{bmatrix}\begin{bmatrix}\mathbf{u}_k^c\\ \mathbf{u}_k^s\\ \mathbf{v}_k^c\\ \mathbf{v}_k^s\end{bmatrix}=\begin{bmatrix}\mathbf{u}_{d,k}^c\\ \mathbf{u}_{d,k}^s\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}. \tag{3.4}$$

Note that the off-diagonal blocks are skew-Hermitian. Introduce the notation

$$\mathbf{M}=\begin{bmatrix}M & 0\\ 0 & M\end{bmatrix},\quad \mathbf{B}=-\sqrt{\beta}\begin{bmatrix}L & \omega_k M\\ -\omega_k M & L\end{bmatrix}.$$

Then the block matrix in (3.4) can be written as,

$$\boldsymbol{\mathcal{A}} = \begin{bmatrix} \mathbf{M} & -\mathbf{B}^* \\ \mathbf{B} & \mathbf{M} \end{bmatrix}.$$

4 Preconditioning method

Following the approach used in several earlier papers, [6,7,8,12] as a preconditioner to $\boldsymbol{\mathcal{A}}$ we choose

$$\boldsymbol{\mathcal{C}} = \begin{bmatrix} \mathbf{M} + \mathbf{B} + \mathbf{B}^* & -\mathbf{B}^* \\ \mathbf{B} & \mathbf{M} \end{bmatrix} = \begin{bmatrix} \mathbf{M} + (\mathbf{L} + \mathbf{L}^*) & -\mathbf{B}^* \\ \mathbf{B} & \mathbf{M} \end{bmatrix}, \tag{4.1}$$

where $\mathbf{L} = \sqrt{\beta} \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix}$. The preconditioned matrix takes the form

$$\boldsymbol{\mathcal{C}}^{-1}\boldsymbol{\mathcal{A}} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} - \boldsymbol{\mathcal{C}}^{-1} \begin{bmatrix} \mathbf{L} + \mathbf{L}^* & 0 \\ 0 & 0 \end{bmatrix}.$$

Further, as shown below, $\boldsymbol{\mathcal{C}}^{-1}$ possesses a block-factorized form

$$\boldsymbol{\mathcal{C}}^{-1} = \begin{bmatrix} I & (\mathbf{M} + \mathbf{L})^{-1} \\ 0 & (\mathbf{M} + \mathbf{L})^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -\mathbf{L} & I \end{bmatrix} \begin{bmatrix} (\mathbf{M} + \mathbf{L}^*)^{-1} & -(A + \mathbf{L}^*)^{-1} \\ 0 & I \end{bmatrix},$$

so,

$$\boldsymbol{\mathcal{C}}^{-1} \begin{bmatrix} \mathbf{L} + \mathbf{L}^* & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} I & (\mathbf{M} + \mathbf{L})^{-1} \\ 0 & (\mathbf{M} + \mathbf{L})^{-1} \end{bmatrix} \begin{bmatrix} (\mathbf{M} + \mathbf{L}^*)^{-1}(\mathbf{L} + \mathbf{L}^*) & 0 \\ -\mathbf{L}(\mathbf{M} + \mathbf{L}^*)^{-1}(\mathbf{L} + \mathbf{L}^*) & 0 \end{bmatrix}.$$

Therefore, besides some matrix-vector multiplication, an action of the preconditioned matrix, involves only a solution with the elliptic type matrices $\mathbf{M} + \mathbf{L}^*$ and $\mathbf{M} + \mathbf{L}$, which consist of blocks that are a sum of mass matrices and discretized elliptic operators.

In addition, as outlined below and has been shown in the above cited earlier publications of the first and the second author, the preconditioning leads to tightly clustered eigenvalues and, hence, very fast convergence.

We present now the preconditioner for a matrix in the general form

$$\mathcal{A} = \begin{bmatrix} A & B_2 \\ B_1 & -A \end{bmatrix},$$

where A, B_1 and B_2 are generic matrices of order $n \times n$, A is assumed to be symmetric and positive definite and $A + B_i$, $i = 1, 2$ are nonsingular. Let

$$\mathcal{C} = \begin{bmatrix} A + B_1 + B_2 & B_2 \\ B_1 & -A \end{bmatrix} \tag{4.2}$$

be a preconditioner to $\mathcal{A}$, to be used in a Krylov subspace type of iteration method, such as GMRES [31] or MINRES [28]. Given a linear matrix preconditioning equation

$$\mathcal{C} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix}, \tag{4.3}$$

by changing the sign of the second equation and adding the first, the system can be written in the equivalent form,

$$\left\{\begin{array}{l}(A+B_1+B_2)\mathbf{x}+B_2\mathbf{y}=\mathbf{f},\\(A+B_2)\mathbf{x}+(A+B_2)\mathbf{y}=\mathbf{f}{-}\mathbf{g},\end{array}\right. \text{ i.e., } \begin{bmatrix}A+B_1 & B_2\\ 0 & A+B_2\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \mathbf{z}\end{bmatrix}=\begin{bmatrix}\mathbf{f}\\ \mathbf{f}{-}\mathbf{g}\end{bmatrix},$$

where $\mathbf{z}=\mathbf{x}+\mathbf{y}$. Hence, $\mathbf{z}=(A+B_2)^{-1}(\mathbf{f}-\mathbf{g})$ and $(A+B_1)\mathbf{x}=\mathbf{f}-B_2\mathbf{z}$. Therefore, the algorithm to compute the solution of (4.3) can be written as $\begin{array}{ll}\text{(1) Solve } (A+B_2)\mathbf{z}=\mathbf{f}-\mathbf{g}, & \text{(3) Solve } (A+B_1)\mathbf{x}=\widetilde{\mathbf{f}},\\ \text{(2) Compute } \widetilde{\mathbf{f}}=\mathbf{f}-B_2\mathbf{z}, & \text{(4) Compute } \mathbf{y}=z-\mathbf{x}.\end{array}$ Alternatively, it is readily seen that

$$\mathcal{C}=\begin{bmatrix}I & 0\\ I & -(A+B_2)\end{bmatrix}\begin{bmatrix}I & B_2\\ 0 & I\end{bmatrix}\begin{bmatrix}A+B_1 & 0\\ I & I\end{bmatrix},$$

so,

$$\mathcal{C}^{-1}=\begin{bmatrix}(A+B_1)^{-1} & 0\\ -(A+B_1)^{-1} & I\end{bmatrix}\begin{bmatrix}I & -B_2\\ 0 & I\end{bmatrix}\begin{bmatrix}I & 0\\ (A+B_2)^{-1} & -(A+B_2)^{-1}\end{bmatrix}.$$

Hence, besides some vector additions, the algorithm involves a solution of a linear systems with $A+B_2$, a matrix-vector multiplication with B_2 and a solution with the matrix $A+B_1$. In practice, the solution of the two linear systems contribute to the major cost of computing an action of $\mathcal{C}^{-1}$. In our applications they correspond to discrete elliptic operators, for which very efficient solution methods are known to exist.

As Proposition 1 shows, the eigenvalues are tightly clustered near the unity. See also [8].

Proposition 1. *Let*

$$\mathcal{A}=\begin{bmatrix}A & B_2\\ -B_1 & A\end{bmatrix}, \quad \mathcal{C}=\mathcal{A}+\begin{bmatrix}B_1+B_2 & 0\\ 0 & 0\end{bmatrix},$$

where we assume that A*, of order* $n\times n$*, is symmetric positive definite,* $B_2=B_1^*$*,* $B_1=B$*,* $B+B^*$ *is positive semidefinite and* $A+B$ *is nonsingular. Then the eigenvalues of* $\mathcal{C}^{-1}\mathcal{A}$ *are real and satisfy*

$$\frac{1}{2}\le\frac{1}{1+\alpha}\le\lambda\le 1,$$

where $\alpha=\max\{Re(\mu)/|\mu|\}$*, and* μ *is an eigenvalue of* $B\mathbf{z}=\mu A\mathbf{z}$*,* $\mathbf{z}\neq 0$*. The dimension of the eigenvalue* $\lambda=1$ *is* $n+n_0$*, where* n_0 *is the dimension of the nontrivial nullspace of* B_1+B_2*. It follows that*

$$\lambda=\frac{1+|\mu|^2}{1+|\mu|^2+2Re(\mu)}=\frac{1}{1+\frac{2Re(\mu)}{1+|\mu|^2}}\ge\frac{1}{1+\frac{Re(\mu)}{|\mu|}}. \tag{4.4}$$

Proof. From

$$\lambda\mathcal{C}\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}=\mathcal{A}\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}, \quad \|\mathbf{x}\|+\|\mathbf{y}\|\neq 0$$

it follows

$$(1-\lambda)\mathcal{C}\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} = \begin{bmatrix}(B_1+B_2)\mathbf{x}\\ 0\end{bmatrix}.$$

Since B_1+B_2 is positive semidefinite, it follows that $\lambda \le 1$. Further, for $\lambda \ne 1$, it holds $B_1\mathbf{x} = A\mathbf{y}$. Hence

$$(1-\lambda)(A+B_1+B_2+B_2A^{-1}B_1)\mathbf{x} = (B_1+B_2)\mathbf{x}$$

or, since A is spd,

$$(1-\lambda)(I+\widetilde{B}+\widetilde{B}^*+\widetilde{B}^*\widetilde{B})\mathbf{x} = (\widetilde{B}+\widetilde{B}^*)\mathbf{x},$$

where $\widetilde{B} = A^{-1/2}BA^{-1/2}$. Therefore $(1-\lambda)(1+2Re(\mu)+|\mu|^2) = 2Re(\mu)$ or

$$1-\lambda = \frac{2Re(\mu)/|\mu|}{1/|\mu|+|\mu|+2Re(\mu)/|\mu|} \le \frac{\alpha}{1+\alpha}.$$

The estimate (4.4) follows directly from the latter relation.

Proposition 1 shows that the relative size, $Re(\mu)/|\mu|$, of the real part of the spectrum of $\widetilde{B} = A^{-1/2}BA^{-1/2}$ determines the lower eigenvalue bound of $\mathcal{C}^{-1}\mathcal{A}$ and, hence, the rate of convergence of the preconditioned iterative solution method. For a small such relative part the convergence of the iterative solution method will be exceptionally rapid. Such small parts can occur for harmonic problems with a large value of the frequency.

It is further readily seen that the eigenvalue problem has a full eigenvector space, i.e. the preconditioned matrix $\mathcal{C}^{-1}\mathcal{A}$ is diagonalizable by a similarity transformation formulation, see e.g. [4].

5 A multilevel-mesh solution method in the context of nonlinear evolution equations

For a nonlinear problem, such as $\frac{\partial u}{\partial t} + \mathcal{L}(u)u = f$, where $\mathcal{L}(a)u = -\nabla\cdot(a(u))\nabla u + \mathbf{w}(\mathbf{u})\cdot\nabla u = f$ the arising block matrix system is nonlinear and one must use a nonlinear solver.

In a similar way as holds for the classical Newton method, to ensure fast convergence of the nonlinear solver, we need a high quality initial guess see e.g. [3]. For this purpose we can use (nested) discretization meshes, $\mathcal{T}_0 \subset \mathcal{T}_1 \subset \ldots \subset \mathcal{T}_m$, obtained by regular refinements of some given coarse mesh ($\mathcal{T}_0$). A scheme to obtain a good starting vector, used e.g., in [29] in the framework of PDE-constrained optimal control problems, is depicted in Algorithm 1.

Thus, in order to solve the nonlinear system on level m we solve it also on all coarser meshes. This, of course, makes the solution procedure computationally demanding.

In [9, 12] the following idea is shown to be very fruitful: solve (accurately) the nonlinear problem on some coarse mesh, prolong the so-obtained solution to the finest ($\mathcal{T}_m$) and solve the linearized problem only once, gaining under certain conditions the order of the accuracy of the discretization of the finest

Algorithm 1 Constructing an initial guess for the nonlinear solver using the full mesh hierarchy

```
Initialize $a, b$ and the solution at $\mathcal{T}_0$
For all $i = 0, 1, \dots, m$
    Solve the nonlinear system on level $i$
    If $i < m$, prolong the current solution to level $i + 1$
EndFor
```

level. We note that this idea is not always directly applicable for optimal control problems. For instance, in the framework of the semi-smooth Newton method applied for problems with a box-constrained solution, the stopping criterion is not related to a certain tolerance but to a set of point identifiers where the box-constraint values are taken exactly and that stop changing ([12]). In this sense we cannot control the accuracy of the solution on a mesh by a tolerance and continue the nonlinear iterations until convergence is achieved. The two-level framework for the semi-smooth Newton method can then be applied as follows. Instead of using all levels in the mesh hierarchy, we choose one (or more, not necessarily consecutive coarser meshes), solve the nonlinear problem there, prolong to $\mathcal{T}_m$ and perform only one Newton step there, thus, solve the linear problem only once. In [12], the effect of using different combinations of meshes is nicely illustrated.

6 Computational and communication complexity of the proposed approach

The suggested method, based on periodic function extension and truncated Fourier expansions, enables much work in parallel. To compute the vectors $\mathbf{u}_k$ and $\mathbf{v}_k$ for a linear problem as in (3.4), we are able to solve N systems (3.4) for each frequency ω_k, independently. These can be assigned to N subsystems of the hardware resource (consisting of a number of cores or nodes, or accelerators), with size large enough to solve the systems efficiently. We refer to this as a first level of parallelism. These solutions do not require any communication. Communication occurs only when the individual components per frequency must be summed up.

The solution of the system in (3.4) is the second level of parallelism. The framework is very similar to solving a single discrete PDE problem and the approaches for parallelizing the computations are well studied. The system is assumed to be solved by an iterative solution method that accommodates variable preconditioning, such as FGMRES ([31]). Each iteration requires one matrix-vector multiplication, a few vector updates and scalar products, and a solution of a system with the preconditioner $\boldsymbol{\mathcal{C}}$. In turn, the latter requires two solutions with the matrices

$$\mathbf{F} = \begin{bmatrix} M + \sqrt{\beta}L & -\omega_k\sqrt{\beta}M \\ \omega_k\sqrt{\beta}M & M + \sqrt{\beta}L \end{bmatrix} \text{ and } \mathbf{F}^* = \begin{bmatrix} M + \sqrt{\beta}L^* & \omega_k\sqrt{\beta}M \\ -\omega_k\sqrt{\beta}M & M + \sqrt{\beta}L^* \end{bmatrix} \tag{6.1}$$

plus one multiplication with $\begin{bmatrix} L^* & -\omega_k M \\ \omega_k M & L^* \end{bmatrix}$.

Using our preconditioner, the solution with $\mathbf{F}$ and $\mathbf{F}^*$ boils down to solutions with the matrices

$$Q = (1 + \omega_k \sqrt{\beta})M + \sqrt{\beta}L \quad \text{and} \quad Q^* = (1 + \omega_k \sqrt{\beta})M + \sqrt{\beta}L^* \tag{6.2}$$

and a multiplication with $\omega_k \sqrt{\beta} M$. For the solution with Q and Q^* we can use the Conjugate Gradient method preconditioned by some Algebraic Multigrid (AMG) method, see e.g. [20, 27] or by some other well-parallelizable preconditioner such as special AMG versions [10,11,13] or such as the gaining popularity Monte Carlo-based approximate inverse preconditioning methods (cf. e.g. [2]).

Assuming that the partitioning of the discretization mesh is done using an appropriate software tool, such as `ParMetis` ([19]), we expect a well-balanced distribution of the computational work, good data locality, no communication bottlenecks and good scalability and efficiency of the proposed solution method.

The preconditioner $\boldsymbol{\mathcal{C}}$ in (4.2) has been used in the context of multi-phase fluid simulations and parallel performance results are available in [30].

To sum up, we propose a computational procedure that is suitable both for large homogeneous or heterogeneous computer platforms, with two-level parallelism. One level of (coarse grain) parallelism is enabled through the fully decoupled computation of the Fourier coefficients of the state and the control, that can be assigned to relatively large computing entities (nodes, groups of nodes, GPUs, etc., depending on the number of the degrees of freedom in the fine discretization mesh). The second level of parallelism is imposed via the solution of the algebraic systems for each individual frequency. To solve those, we advocate a method, that allows for using ready toolboxes from already available highly optimized parallel libraries, such as HYPRE ([24]), Trilinos ([16]), MFEM ([15]), PETSc ([14]), to name some. The potential of the approach is similar to that of the method to solve fractional diffusion problems reported in [35, 36], where also some parallel tests are included.

7 Illustrating test problems

For the purpose of illustrating the approximation of the time-dependent solution and the robustness of the method with respect to mesh and regularization parameters we consider first the heat problem. Then, to further illustrate the robustness of the preconditioning method with respect to a larger set of parameters – meshsize, regularization, angular frequency and other problem parameters, we shortly comment also on an eddy-current time-harmonic problem.

All numerical experiments, reported here, are performed in Matlab on a laptop Lenovo ThinkPad T500 2055 Core 2 Duo T9400 / 2.53 GHz Centrino 2 with 8GB memory.

7.1 The heat equation as constraint

Consider the transient heat equation as a state constraint in the following optimal control problem.

Problem 1.

$$\begin{aligned} &\frac{\partial}{\partial t}u(\mathbf{x},t) - \nabla\cdot(\kappa(\mathbf{x})\nabla u(\mathbf{x},t)) = v(\mathbf{x},t) \;\text{ for all } (\mathbf{x},t)\in \Omega\times(0,1],\\ &u(\mathbf{x},t) = 0 \quad \text{for all } (\mathbf{x},t)\in\partial\Omega\times(0,1],\\ &u(\mathbf{x},0) = 0 \quad \text{for all } \mathbf{x}\in\Omega, \end{aligned} \tag{7.1}$$

where u_d is the desired solution. The form of u_d is taken as Example 3 in [23], namely,

$$u_d = \chi_{[\frac{1}{4},\frac{3}{4}]}(t)\chi_{[\frac{1}{2},1]^2}(x).$$

In (7.1), $\Omega = [0,1]^2$ and $\kappa(\mathbf{x})$ is the thermal conductivity. The function χ is the characteristic function, being equal to one when the argument is in the given interval and zero elsewhere. In alignment with the assumptions, we infer that the solution u, the desired solution u_d and the control v are of the form (2.5) and that u_d is extended to the interval $[0,2]$ by use of symmetric mirroring as

$$u^c_{d,k} = \frac{1}{T}\Big(\int_0^T u_d(x,t)\cos(\omega_k t)dt + \int_T^{2T} u_d(x,2T-t)\cos(\omega_k t)dt\Big),$$

$$u^s_{d,k} = \frac{1}{T}\Big(\int_0^T u_d(x,t)\sin(\omega_k t)dt + \int_T^{2T} u_d(x,2T-t)\sin(\omega_k t)dt\Big).$$

The space discretization is done using bilinear finite element basis functions on a square discretization mesh.

In the experiments we use only up to five terms in the extension, i.e., $N = 5$. The solution procedure is as follows. The outer solution method and the solver for the blocks $\mathbf{F}$ and $\mathbf{F}^*$ (cf. (6.1)), appearing in the preconditioner $\boldsymbol{\mathcal{C}}$ (cf. (4.1)) is FGMRES. The very inner solver for the blocks Q and Q^* (cf. (6.2)) is the Conjugate Gradient method, preconditioned by AGMG (cf. [27]). The (relative) stopping criteria are 10^{-6}, 10^{-3} and 10^{-3}, correspondingly. The results for various h, N and β are presented in Table 1. As in [21], ω is fixed as 2π. The iterations are shown in the form $IT_{outer}(IT^{(av)}_{inner}/IT^{(av)}_{agmg})$, where IT_{outer} is the number of outer FGMRES iterations, $IT^{(av)}_{inner}$ is the average number of inner FGMRES iterations and $IT^{(av)}_{agmg}$ is the average number of the AGMG-preconditioned conjugate gradient. Even though the implementation is in Matlab, we include as a reference the total execution time (in seconds).

As shown in Table 1, for each individual frequency ω_k, $k = 0,1,\ldots,5$, and for all chosen values of h and β, the number of outer iterations varies between 6 and 8. The number of inner iterations varies between 1 and 4 and the innermost iterations using the AGMG-preconditioned conjugate gradient vary also within a very short interval, between 2 and 7. This means that the method shows a very robust behaviour. It is worth noting that even though we use Matlab, the total computing time increases with an almost constant factor between 4.5 and 6.8, when h^{-1} is doubled, which is relatively close to the optimal value 4.

For comparison, in Table 2 we include the performance of the block-diagonally preconditioned MINRES, as described in [21]. The stopping criteria are 10^{-6} for the outer MINRES and 10^{-3} for the blocks, solved by AGMG.

Table 1. Problem 1: Performance results for FGMRES, $\boldsymbol{\mathcal{A}}$ preconditioned by $\boldsymbol{\mathcal{C}}$ in (4.1)

		$\beta=10^{-2}$		$\beta=10^{-4}$		$\beta=10^{-6}$		$\beta=10^{-8}$	
h	k	IT	CPU	IT	CPU	IT	CPU	IT	CPU
$\frac{1}{128}$	0	7(1/3)	0.60	8(1/3)	0.67	8(1/2)	0.63	8(1/2)	0.47
	1	7(3/7)	2.38	8(2/6)	1.74	8(2/5)	1.53	8(1/4)	0.62
	2	7(4/6)	2.92	8(3/6)	2.52	8(2/5)	1.50	8(1/4)	0.63
	3	6(4/6)	2.60	8(3/6)	2.47	8(2/5)	1.52	8(1/4)	0.69
	4	6(4/6)	2.67	8(3/6)	2.44	8(2/5)	1.51	8(1/4)	0.77
	5	6(4/6)	2.62	8(4/6)	3.06	8(2/5)	1.56	8(2/4)	0.81
$\frac{1}{256}$	0	7(1/4)	3.16	9(1/3)	3.83	8(1/3)	3.03	8(1/2)	2.64
	1	7(3/7)	11.62	8(2/6)	8.72	8(2/5)	7.73	8(1/4)	3.93
	2	7(4/7)	14.01	8(3/6)	12.13	8(2/5)	7.66	8(1/4)	4.50
	3	6(4/7)	12.99	8(3/6)	12.05	8(2/5)	7.70	8(1/4)	4.63
	4	6(4/7)	13.68	8(3/6)	11.99	8(2/5)	7.67	8(1/4)	4.89
	5	6(4/7)	13.76	8(4/6)	14.52	8(2/5)	7.75	8(2/4)	5.53
$\frac{1}{512}$	0	7(1/4)	22.52	8(1/4)	18.81	8(1/3)	17.72	8(1/3)	16.29
	1	7(3/7)	66.40	8(2/7)	45.75	8(2/6)	41.77	8(1/5)	23.15
	2	7(4/7)	71.25	8(3/7)	63.22	8(2/6)	40.81	8(1/5)	24.59
	3	6(4/7)	65.58	8(3/7)	63.87	8(2/6)	40.54	8(1/5)	26.15
	4	6(4/7)	66.63	8(3/7)	67.86	8(2/6)	41.56	8(2/5)	30.06
	5	6(4/7)	65.91	8(4/7)	79.63	8(2/6)	41.18	8(2/5)	33.74

Table 2. Problem 1: Performance of the block-diagonally preconditioned MINRES, equations (33) and (34) in [21]

		$\beta=10^{-2}$		$\beta=10^{-4}$		$\beta=10^{-6}$		$\beta=10^{-8}$	
h	k	IT	CPU	IT	CPU	IT	CPU	IT	CPU
$\frac{1}{128}$	0	14(3)	0.83	20(3)	1.04	24(2)	1.14	25(2)	0.78
	1	16(6)	1.51	22(5)	1.83	22(4)	1.53	24(4)	0.99
	2	18(6)	1.68	22(5)	1.79	24(4)	1.75	24(4)	0.95
	3	18(6)	1.65	22(5)	1.79	24(4)	1.66	24(4)	0.95
	4	20(6)	1.82	24(5)	1.95	24(4)	1.68	24(4)	0.96
	5	20(5)	1.76	24(5)	1.97	24(4)	1.66	24(4)	0.97
$\frac{1}{256}$	0	14(3)	3.93	20(3)	5.21	24(2)	5.68	26(2)	4.88
	1	16(6)	7.46	22(6)	9.29	24(5)	8.77	26(4)	7.14
	2	18(6)	8.27	24(6)	9.98	25(5)	9.18	26(4)	7.19
	3	20(6)	9.05	24(6)	10.17	24(5)	8.82	26(4)	7.29
	4	20(6)	8.96	24(6)	10.18	24(5)	8.96	26(4)	7.25
	5	20(6)	9.07	25(6)	10.56	24(5)	9.25	26(4)	7.17
$\frac{1}{512}$	0	14(3)	24.90	21(3)	35.52	24(3)	26.88	26(2)	36.89
	1	16(7)	39.67	22(6)	48.09	25(5)	42.68	26(4)	37.73
	2	18(6)	41.76	24(6)	54.47	25(5)	49.75	26(5)	43.82
	3	20(6)	42.83	24(6)	49.37	25(5)	43.17	26(5)	54.94
	4	20(6)	44.64	24(6)	60.88	24(5)	43.05	26(5)	55.67
	5	20(6)	43.25	26(6)	63.45	24(5)	40.96	26(5)	42.47

Table 2 confirms the robust behaviour of MINRES, documented in [21]. We see, however, that for small values of β, which must be used to obtain a solution sufficiently close to the desired target, our method outperforms MINRES.

To test the accuracy of the solution the resulting approximate solution is compared with the solution of an implicit time-stepping method, the midpoint trapezoidal (or the implicit midpoint) method, briefly recalled for completeness. Consider the ordinary differential equation $x'(t) = f(t, x(t))$ with initial condition $x(0) = x_0$. The midpoint trapezoidal methods reads as follows:

$$x_{k+1} = x_k + \Delta_t f\left(\frac{t_k + t_{k+1}}{2}, \frac{x_k + x_{k+1}}{2}\right),\ k \geq 0,$$

where Δ_t is the timestep. The method is implicit, second order accurate and unconditionally stable for linear problems, as the one in hand. Consider the equation in (2.1). After space discretization and applying the above time-stepping scheme, the problem becomes as follows:

$$\begin{aligned} &\mathit{For}\ k = 0, 1, \ldots, m, \\ &\quad \mathit{Solve} \quad (M + \tfrac{1}{2}\Delta_t L)\mathbf{u}_{k+1} = (M - \tfrac{1}{2}\Delta_t L)\mathbf{u}_k + \Delta_t \mathbf{v}^{opt}_{k+1/2}. \end{aligned}$$

Here, $m = T/\Delta_t$ and $\mathbf{v}^{opt}$ is the optimal control, obtained from the optimization problem, evaluated at the required time. For the solution of systems with $M + \frac{1}{2}\Delta_t L$ we use the AGMG-preconditioned conjugate gradient method with a relative stopping criterion 10^{-3}. We compare the solutions at the final time T. The results are shown in Table 3.

Table 3. Problem 1: Performance of the midpoint trapezoidal method ($N = 5$)

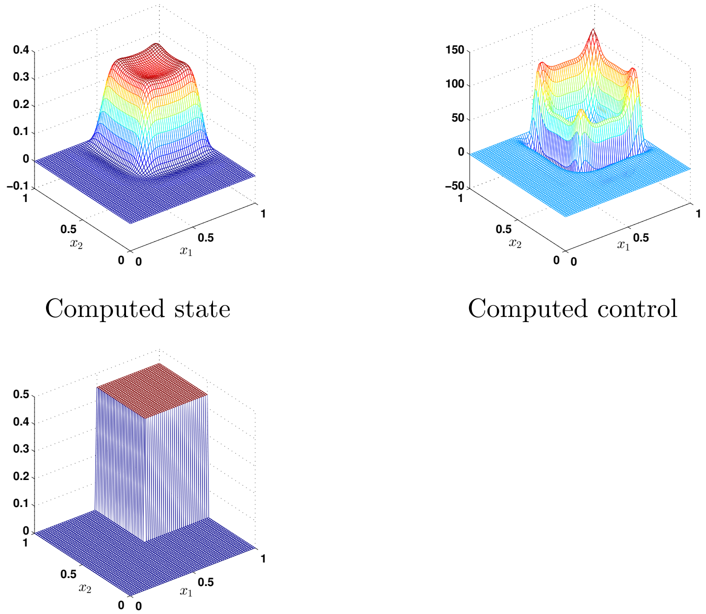
		$\beta = 10^{-2}$		$\beta = 10^{-4}$		$\beta = 10^{-6}$		$\beta = 10^{-8}$	
h	Δ_t	IT	CPU	IT	CPU	IT	CPU	IT	CPU
$\frac{1}{128}$	0.1	6	0.41	6	0.41	6	0.41	5	0.38
	0.05	6	0.68	6	0.69	6	0.66	5	0.65
	0.025	6	1.18	6	1.19	6	1.23	5	1.08
$\frac{1}{256}$	0.1	7	1.42	7	1.41	7	1.39	6	1.30
	0.05	7	2.74	7	2.70	6	2.45	6	2.44
	0.025	6	4.70	6	4.71	7	5.23	6	4.70
$\frac{1}{512}$	0.1	7	8.28	7	7.01	7	8.56	6	6.60
	0.05	7	15.57	7	13.91	7	17.35	6	14.17
	0.025	7	29.64	7	27.62	7	30.32	6	31.56

The difference in the solution obtained within the optimal control and the solution obtained via the time stepping scheme is illustrated in Table 4. The difference is computed $\frac{||u_{opt} - u_k||_2}{||u_{opt}||_2}$ with u_{opt} being the state computed by the optimal control method and u_k – computed by the time-stepping method.

Table 4. Problem 1: Difference between the optimal control solution u_{opt} and the solution from the time stepping method u_k for $t_k = T$

		$\beta = 10^{-2}$		$\beta = 10^{-4}$		$\beta = 10^{-6}$		$\beta = 10^{-8}$	
h	Δ_t	$N = 5$	$N = 10$	$N = 5$	$N = 10$	$N = 5$	$N = 10$	$N = 5$	$N = 10$
$\frac{1}{128}$	0.1	0.45	0.50	0.88	1.12	0.90	1.17	0.85	1.11
	0.05	0.12	0.15	0.24	0.38	0.26	0.45	0.25	0.45
	0.025	0.03	0.04	0.06	0.10	0.07	0.12	0.07	0.12

Figure 2 shows the obtained optimal control and state vectors at the final time $T (= 1)$, as well as the form of the desired state, approximated by a truncated Fourier expansion with $N = 5$. Figure 3 visualizes the solution from the time-stepping procedure for different timesteps.



Computed state

Computed control

Desired state

Figure 2. Problem 1: Surface plots of the desired state u_d, the computed state u and the control v on a 64×64 grid and $\beta = 10^{-6}$

7.2 Eddy current electromagnetic problem with Maxwell's equations as constraint

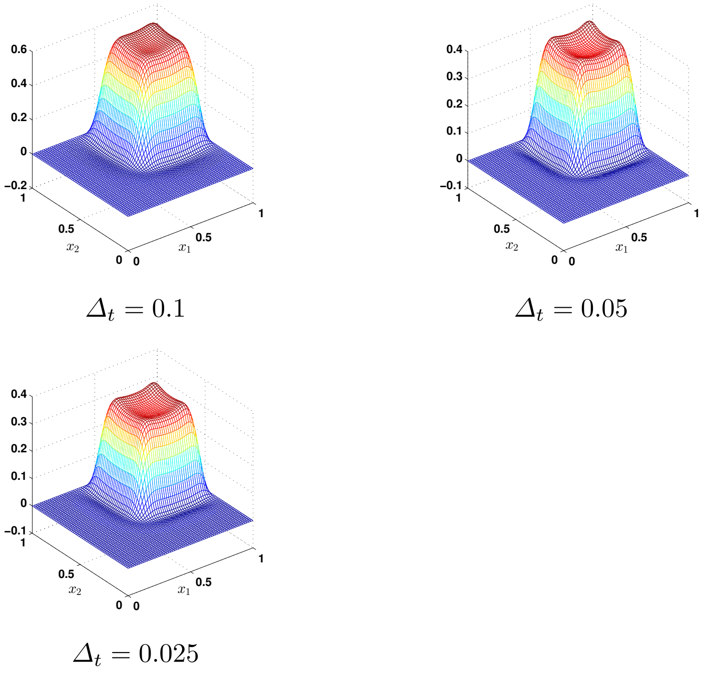
For completeness, we refer shortly also to a second test problem involving several regularization and problem parameters. The aim here is to demonstrate the applicability of our approach to more difficult problems.

Consider then the linear eddy current case of Maxwell's equations in a bounded domain Ω with Lipschitz boundary Γ. The problem reads: find a time-dependent magnetic vector potential y such that

$$\left\{ \begin{array}{rcll} \sigma \dfrac{\partial y}{\partial t} + \operatorname{curl}(\nu \operatorname{curl} y) & = & j & \text{in } \Omega \times (0,T), \\ y \times n & = & 0 & \text{on } \partial\Omega \times (0,T), \\ y & = & y_0 & \text{on } \partial\Omega \times \{0\}, \end{array} \right. \tag{7.2}$$

where σ, ν and j denote the electrical conductivity, magnetic reluctivity, and the external current density, respectively, and where n is the outward normal vector to $\partial\Omega$. Note that the conductivity can be zero, such as in air.

Due to the discontinuity of σ and to obtain uniqueness in the nonconducting regions, unless the solution is divergence free and a classical inf − sup stability relation holds, the state equation must be regularized. We do this here by adding a positive term εy, $\varepsilon > 0$ to the state equation. The regularized problem



$\Delta_t = 0.1$

$\Delta_t = 0.05$

$\Delta_t = 0.025$

Figure 3. Problem 1: Surface plots of the state obtained by different timesteps on a 64×64 grid and $\beta = 10^{-6}$

takes then the form,

$$\left\{\begin{array}{rcll} \sigma\frac{\partial y}{\partial t} + \mathrm{curl}(\nu \,\mathrm{curl}\, y) + \varepsilon y &=& j & \text{in } \Omega \times (0,T), \\ y \times n &=& 0 & \text{on } \partial\Omega \times (0,T), \\ y &=& y_0 & \text{in } \Omega \times \{0\}. \end{array}\right.$$

In the time-harmonic regime with angular frequency ω, $y(x,t) = Re\{\widehat{y}(x)e^{i\omega t}\}$, $j(x,t) = Re\{\widehat{j}(x)e^{i\omega t}\}$, it leads to finding the complex-valued amplitude $\widehat{y}$ satisfying

$$\left\{\begin{array}{rl} i\omega\sigma\widehat{y} + \mathrm{curl}(\nu \,\mathrm{curl}\, y) + \varepsilon\widehat{y} = \widehat{j} & \text{in } \Omega, \\ \widehat{y} \times n = 0 & \text{on } \partial\Omega. \end{array}\right.$$

The problem is formulated in sense of distributions: find $\widehat{y} \in H_0(\mathrm{curl}, \Omega)$

$$i\omega \int_\Omega \sigma\widehat{y} \cdot v dx + \int_\Omega (\nu \mathrm{curl}\, \widehat{y} \cdot \ \mathrm{curl}\, v + \varepsilon\widehat{y} \cdot v) dx = \int_\Omega \widehat{j} \cdot v\, dx$$

for all complex-valued test functions $v \in H_0(\mathrm{curl}; \Omega)$, where

$$H_0(\mathrm{curl}; \Omega) := \{v \in L^2(\Omega)^3 : \ \mathrm{curl}\, v \in L^2(\Omega)^3, v \times n = 0 \text{ on } \Gamma\}.$$

Assuming $\widehat{j} \in L^2(\Omega)^3$, the linear form is bounded. Assuming further that $\sigma, \nu \in L^\infty(\Omega)$, $\sigma(x) \geq 0$, and $\nu(x) \geq \nu_0 > 0$ the bilinear form is bounded and elliptic, therefore, the problem is uniquely solvable and the solution depends continuously on the data.

Using conforming finite element approximations requires continuity of the traces. For this purpose we use Nèdèlec finite elements, see [25, 26]. We arrive at the linear system

$$(i\omega M + K)z = b, \tag{7.3}$$

where $M_{ij} := \int_\Omega \sigma\varphi_j \cdot \varphi_i$, $K_{ij} := \int_\Omega \nu \text{ curl } \varphi_j \cdot \text{ curl}\varphi_i + \varepsilon\varphi_j \cdot \varphi_i$, $b_i := \int_\Omega \widehat{j} \cdot \varphi_i$ with $\varphi_i(x)$, $i, j = 1, \dots, n$, being the lowest order Nèdèlec basis functions. To avoid complex arithmetics we can rewrite (7.3) in real-valued block matrix form

$$\begin{bmatrix} K & -\omega M \\ \omega M & K \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \xi \\ \eta \end{bmatrix},$$

where $z = x+iy$ and $b = \xi+i\eta$. We recognize the same type of block matrix that arise in the optimal control problem, constrained by a time-harmonic equation.

For the time-harmonic problem the aim is to compute a periodic steady state solution (y, v) that satisfies (7.2) but not necessarily the initial condition $y = y_0$. Including instead the periodicity condition, $y(0) = y(2T)$, the state equation takes the form

$$\begin{cases} \sigma\frac{\partial y}{\partial t} + \text{curl}(\nu \text{ curl } y) + \varepsilon y &= v & \text{in } \Omega \times (0, T), \\ y \times n &= 0 & \text{on } \partial\Omega \times (0, T), \\ y(0) &= y(2T) & \text{in } \Omega. \end{cases}$$

We denote the adjoint variable with w. Similarly, the condition $w(2T) = 0$ is replaced by the periodicity condition, $w(0) = w(2T)$. We consider then a time-harmonic desired state

$$y_d(x, t) = y_d^c(\mathbf{x})\cos(wt) + y_d^s(\mathbf{x})\sin(\omega t).$$

Due to the linearity of the problem, the state y, the Lagrange multiplier, i.e. co-state w and the control u are time-harmonic as well with the same frequency. Considering the case where the control and observation regions are nonzero only on a subspace. As shown in [10] this leads to a four-by-four block matrix system with two-by-two block of the same from as before, where both are preconditioned by use of the PRESB method. The arising system is solved by a coupled inner-outer iteration method. Even though one now gets coupled inner-outer iterations, which multiply up, this is a viable approach since, as follows from Proposition 1, the arising condition numbers for both the outer and inner iterations are bounded by a not large number $1 + \alpha$, $0 < \alpha \leq 1$, for different values of α, as in Proposition 1. Hence, there will be few iterations. Furthermore, as is seen by numerical tests, in practice it suffices to solve the inner systems to a fairly rough relative accuracy, say 10^{-2} to get the smallest or nearly the smallest number of outer iterations, so, there will be very few iterations. As before, the major parallelism comes from the parallel solution of the problem for different frequencies in addition to the parallel implementation possible for the inner solution method used.

The robustness of the method has been reported in [10] for different values of the reluctivity and two meshsizes. In particular, the method is very robust with respective to the frequency ω.

Acknowledgements

The work of the first author is supported by the projects LD15105 Ultrascale computing in geosciences and LQ1602 IT4Innovations excellence in science supported by the Ministry of Education, Youth and Sports of the Czech Republic. The work of the third author is funded by the China Scholarship Council (File No. 201606180086) and by the National Natural Science Foundation of China (Grant No. 11771193). His work is performed during his visit at Uppsala University, Sweden.

References

[1] R. Alexander. Diagonally implicit Runge-Kutta methods for stiff ODEs. *SIAM Journal on Numerical Analysis*, **14**(6):1006–1021, 1977. https://doi.org/10.1137/0714068.

[2] V. Alexandrov, O. Esquivel-Flores, S. Ivanovska and A. Karaivanova. On the preconditioned quasi-Monte Carlo algorithm for matrix computations. In I. Lirkov, S. D. Margenov and J. Waśniewski(Eds.), *International Conference on Large-Scale Scientific Computing*, volume 9374, pp. 163–171, Cham, 2015. Springer. https://doi.org/10.1007/978-3-319-26520-9_17.

[3] O. Axelsson. On mesh independence and Newton type methods. *Applications of Mathematics*, **38**(4):249–265, 1993.

[4] O. Axelsson. *Iterative Solution Methods.* Cambridge University Press, Cambridge, 1994. https://doi.org/10.1017/CBO9780511624100.

[5] O. Axelsson and V.A. Barker. *Finite Element Solution of Boundary Value Problems: Theory and Computation.* Society for Industrial and Applied Mathematics, 2001. https://doi.org/10.1137/1.9780898719253.

[6] O. Axelsson, S. Farouq and M. Neytcheva. Comparison of preconditioned Krylov subspace iteration methods for PDE-constrained optimization problems. Poisson and convection-diffusion control. *Numerical Algorithms*, **73**(3):631–663, 2016. https://doi.org/10.1007/s11075-016-0111-1.

[7] O. Axelsson, S. Farouq and M. Neytcheva. Comparison of preconditioned Krylov subspace iteration methods for PDE-constrained optimization problems. Stokes control. *Numerical Algorithms*, **74**(1):19–37, 2017. https://doi.org/10.1007/s11075-016-0136-5.

[8] O. Axelsson, S. Farouq and M. Neytcheva. A preconditioner for optimal control problems, constrained by Stokes equation with a time-harmonic control. *Journal of Computational and Applied Mathematics*, **310**(C):5–18, 2017. https://doi.org/10.1016/j.cam.2016.05.029.

[9] O. Axelsson and W. Layton. A two-level method for the discretization of nonlinear boundary value problems. *SIAM Journal on Numerical Analysis*, **33**(6):2359–2374, 1996. https://doi.org/10.1137/S0036142993247104.

[10] O. Axelsson and D. Lukáš. Preconditioning methods for eddy current optimally controlled time-harmonic electromagnetic problems. *Journal of Numerical Mathematics*, 2018. https://doi.org/10.1515/jnma-2017-0064.

[11] O. Axelsson and M. Neytcheva. Algebraic multilevel iteration method for Stieltjes matrices. *Numerical Linear Algebra with Applications*, **1**(3):213–236, 1994. https://doi.org/10.1002/nla.1680010302.

[12] O. Axelsson, M. Neytcheva and A. Ström. An efficient preconditioning method for state box-constrained optimal control problems. Technical report, Uppsala University, Department of Information Technology, 2017. Available from Internet: `http://www.it.uu.se/research/publications/reports/2017-004/`.

[13] O. Axelsson and P.S. Vassilevski. Algebraic multilevel preconditioning methods. I. *Numerische Mathematik*, **56**(2-3):157–177, 1989. https://doi.org/10.1007/BF01409783.

[14] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. BMFEMe, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, D. A. May, L. Curfman McInnes, K. Rupp, B. F. Smith, S. Zampini, H. Zhang and H. Zhang. PETSc Web page, 2017. Available from Internet: `http://www.mcs.anl.gov/petsc`.

[15] T. V. Kolev et al. MFEM: Modular finite element methods. Available from Internet: `http://mfem.org/`.

[16] M. Heroux, R. Bartlett, V. Howle, R Hoekstra, J. Hu, T. Kolda, R.Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring and A. Williams. An overview of Trilinos. Technical Report SAND2003-2927, Sandia National Laboratories, 2003.

[17] R. Herzog and E.W. Sachs. Preconditioned conjugate gradient method for optimal control problems with control and state constraints. *SIAM Journal on Matrix Analysis and Applications*, **31**(5):2291–2317, 2010. https://doi.org/10.1137/090779127.

[18] M. Hintermüller and M. Hinze. Moreau-Yosida regularization in state constrained elliptic control problems: error estimates and parameter adjustment. *SIAM Journal on Numerical Analysis*, **47**(3):1666–1683, 2009. https://doi.org/10.1137/080718735.

[19] G. Karypis and V. Kumar. ParMETIS-parallel graph partitioning and fill-reducing matrix ordering, 2013. Available from Internet: `http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview`.

[20] T. V. Kolev and P.S. Vassilevski. Parallel auxiliary space AMG solver for $H(curl)$ problems. *Journal of Computational Mathematics*, **27**(5):604–623, 2009. https://doi.org/10.4208/jcm.2009.27.5.013.

[21] M. Kollmann, M. Kolmbauer, U. Langer, M. Wolfmayr and W. Zulehner. A robust finite element solver for a multiharmonic parabolic optimal control problem. *Computers & Mathematics with Applications*, **65**(3):469–486, 2013. https://doi.org/10.1016/j.camwa.2012.06.012.

[22] M. Kolmbauer and U. Langer. A robust preconditioned MINRES solver for distributed time-periodic eddy current optimal control problems. *SIAM Journal on Scientific Computing.*, **34**(6):B785–B809, 2012. https://doi.org/10.1137/110842533.

[23] U. Langer and M. Wolfmayr. Multiharmonic finite element analysis of a time-periodic parabolic optimal control problem. *Journal of Numerical Mathematics*, **21**(4):265–300, 2013. https://doi.org/10.1515/jnum-2013-0011.

[24] Lawrence Livermore National Laboratory. *hypre: High Performance Preconditioners*. Available from Internet: `http://www.llnl.gov/CASC/hypre`.

[25] J.C. Nèdèlec. Mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, **35**(3):315–341, 1980. https://doi.org/10.1007/BF01396415.

[26] J.C. Nèdèlec. A new family of mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, **50**(1):57–81, 1986. https://doi.org/10.1007/BF01389668.

[27] Y. Notay. An aggregation-based algebraic multigrid method. *Electronic Transactions on Numerical Analysis*, **37**:123–146, 2010.

[28] C.C. Paige and M.A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, **12**(4):617–629, 1975. https://doi.org/10.1137/0712047.

[29] J.W. Pearson, M. Stoll and A.J. Wathen. Preconditioners for state-constrained optimal control problems with Moreau-Yosida penalty function. *Numerical Linear Algebra with Applications*, **21**(1):81–97, 2014. https://doi.org/10.1002/nla.1863.

[30] A. Redlund and G. Cheng. Implementation and performance studies of a three-phase model solver. Technical report, Uppsala University, Department of Information Technology, 2013. Available from Internet: `http://www.it.uu.se/edu/course/homepage/projektTDB/ht13/project13/`.

[31] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific Computing*, **14**(2):461–469, 1993. https://doi.org/10.1137/0914028.

[32] Y. Saad. *Iterative methods for sparse linear systems.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003. https://doi.org/10.1137/1.9780898718003.

[33] P.J. van der Houwen and B.P. Sommeijer. Analysis of parallel diagonally implicit iteration of Runge-Kutta methods. *Applied Numerical Mathematics*, **11**(1-3):169–188, 1993. https://doi.org/10.1016/0168-9274(93)90047-U.

[34] P.S. Vassilevski. *Multilevel Block Factorization Preconditioners.* Springer-Verlag, New York, 2008.

[35] R. Čiegis, V. Starikovičius and S. Margenov. On parallel numerical algorithms for fractional diffusion problems. In J. Carretero, J. Garsia Blas and S. Margenov(Eds.), *Proceedings of the Third International Workshop on Sustainable Ultrascale Computing Systems (NESUS 2016).* Universidad Carlos III de Madrid, 12 2016.

[36] R. Čiegis, V. Starikovičius, S. Margenov and R.Kriauzienė. Parallel solvers for fractional power diffusion problems. *Concurrency and Computation: Practice and Experience*, **29**(24), 2017. https://doi.org/10.1002/cpe.4216.

[37] J.-C. Xu. Two-grid discretization techniques for linear and nonlinear PDEs. *SIAM Journal on Numerical Analysis.*, **33**(5):1759–1777, 1996. https://doi.org/10.1137/S0036142992232949.