An exact and approximate Schur complement method for time-harmonic optimal control problems

Owe Axelsson[1,3], Dalibor Lukáš[2], and Maya Neytcheva[3]

[1]The Czech Academy of Sciences, Institute of Geonics AS CR, Ostrava, Czech Republic
[2]VSB-Technical University of Ostrava, Czech Republic
[3]Department of Information Technology, Uppsala University, Uppsala, Sweden

September 29, 2023

Abstract

Time-harmonic control problems, constrained by a linear differential equation can be solved efficiently by utilising a Fourier time series expansion in the angular frequency variable. Then the optimal solution consists of a series of complex variable space discretization equations, which are uncoupled with respect to the different frequencies. Hence, it suffices to consider a single equation with the angular frequency as a parameter. We consider here methods to solve the so-arising linear system of equations, and describe, analyse and test the performance of two novel approaches, based on its exact and approximate Schur complement. The performance of both methods is tested and compared with another existing method.

Keywords: PDE-constrained optimization, Electromagnetics, complex linear systems, iterative methods, preconditioning

1 Introduction

Consider an optimal control problem, i.e., seek the solution (u, v) that minimizes the functional

$$\mathcal{J}(u,v) = \frac{1}{2} \int\limits_{\Omega\times[0,T]} \|u - u_d\|^2 \, d\boldsymbol{x} \, dt + \frac{\beta}{2} \int\limits_{\Omega\times[0,T]} \|v\|^2 \, d\boldsymbol{x} \, dt$$

subject to a time-dependent elliptic state equation, defined in a bounded domain $\Omega \subset \mathbb{R}^m, m = 2, 3$ and a time interval $[0, T]$. Here $u(\boldsymbol{x}, t)$ is the state variable, u_d is a given target (desired) state solution and v is the control variable, acting as a source function, to achieve this goal. The control cost depends on a (regularization) parameter β, which is positive and small.

As an example, the state equation can be the time-dependent heat equation

$$\mathfrak{L}u \equiv \frac{\partial u}{\partial t} + \nabla(k(\boldsymbol{x})\nabla u) = \mathfrak{g} + v \quad \text{in } \Omega \times [0,T], \tag{1}$$

where $u(\boldsymbol{x},t)$ is the temperature and k is the thermal conductivity, or the curl $-$ curl equation, arising in electromagnetic eddy-current problems,

$$\mathfrak{L}u \equiv \sigma\frac{\partial u}{\partial t} + \nabla \times (\frac{1}{\mu}\,\nabla \times u) + \epsilon\, u = \mathfrak{g} + v \quad \text{in } \Omega \times [0,T] \tag{2}$$

with proper boundary and initial conditions. In equation (2) u is the magnetic potential, the problem parameter μ is the permeability of the media, σ is the conductivity and ϵ is a regularization parameter.

Optimal control problems constrained by (1) are studied in [1, 2, 3] and constrained by (2) - in e.g., [4, 5, 6, 7, 8, 9].

One way to approach solving the optimal control problem is to incorporate the constraint equation in a Lagrangian functional with an adjoint function w, namely,

$$\mathcal{L}(u,v,w) = \mathcal{J}(u,v) + \int\limits_{\Omega\times[0,T]} (\mathfrak{L}u(\boldsymbol{x},t) - \mathfrak{g} - v)w(\boldsymbol{x},t)\; d\boldsymbol{x}\; dt.$$

We assume periodic boundary conditions $u(\boldsymbol{x},0) = u(\boldsymbol{x},T)$ and consider time-harmonic solutions of the form $u(\boldsymbol{x},t) = \widetilde{u}(\boldsymbol{x})e^{i\omega\, t/T}$ for a given frequency ω.

We adopt the 'discretize-then-optimize- framework. Using appropriate finite elements, after space discretization and utilizing the time-harmonic setting, we formulate the first order necessary optimality conditions, also referred to as the Karush-Kuhn-Tucker (KKT) conditions. In this case the KKT system is of a saddle point form with a three-by-three block matrix as follows,

$$\begin{bmatrix} A & 0 & B^T - i\omega A \\ 0 & \beta A & -A \\ B + i\omega A & -A & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} A\boldsymbol{u}_d \\ \mathbf{0} \\ \boldsymbol{g} \end{bmatrix}, \tag{3}$$

where typically A is a mass matrix and B is a stiffness matrix. For the considered problems A is symmetric and positive definite (spd) and B is symmetric and positive semi-definite (spsd).

When the heat equation is the state constraint, then $\boldsymbol{u}$ and $\boldsymbol{v}$ are discrete scalar functions and A and B are matrices of size n, equal to the number of spatial degrees of freedom. For the case of eddy current equations (2), $\boldsymbol{u}$ and $\boldsymbol{v}$ are discrete vector variables with two or three components per space discretization point and the matrices A and B are block matrices of size $2n$ or $3n$ in two and three dimensions, correspondingly.

Consider now the matrix in (3). More generally, we assume that B is real and $B + B^T$ is spsd. After elimination of the adjoint variable $\boldsymbol{w}$ we obtain the compressed system

$$\begin{bmatrix} A & \beta C^* \\ C & -A \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} A\boldsymbol{u}_d \\ \boldsymbol{g} \end{bmatrix},$$

with $C = B + i\omega A$. We scale the system further and introduce $\widetilde{\boldsymbol{v}} = -\sqrt{\beta}\boldsymbol{v}$ to get

$$\mathcal{A} \begin{bmatrix} \boldsymbol{u} \\ \widetilde{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} A & -\widetilde{C}^* \\ \widetilde{C} & A \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \widetilde{\boldsymbol{v}} \end{bmatrix} = \begin{bmatrix} A\boldsymbol{u}_d \\ \widetilde{\boldsymbol{g}} \end{bmatrix}, \tag{4}$$

where $\widetilde{C} = \sqrt{\beta}C = \widetilde{B} + i\widetilde{\omega}A$, $\widetilde{B} = \sqrt{\beta}B$, $\widetilde{\omega} = \sqrt{\beta}\omega$ and $\widetilde{\boldsymbol{g}} = \sqrt{\beta}\boldsymbol{g}$.

In this work we focus on the solution of (4) via its Schur complement. Needless to mention, we assume that the problem sizes are large, which motivates the usage of preconditioned iterative methods.

We note that the Schur complement of $\mathcal{A}$,

$$S = A + \widetilde{C}A^{-1}\widetilde{C}^* = (1 + \widetilde{\omega}^2)A + \widetilde{B}A^{-1}\widetilde{B}^T$$

is spd, like A, thus $\mathcal{A}$ is regular.

For the iterative solution of (4) we deal with the following three methods.

(i) Method **M1**, considered here for comparison reasons, based on the PRESB (PREconditioning for matrices with Square Blocks) method, see, e.g., [7].

(ii) Method **M2**, based on the exact factorization of the Schur complement matrix S in complex-valued factors.

(iii) Method **M3**, based on an approximate factorization of the Schur complement matrix in real-valued factors.

The first two methods involve solution of inner systems with complex matrices. Two efficient methods to solve systems with complex matrices are presented in Section 2. The implementation of the methods **M1**, **M2** and **M3** is described in Section 3 and the spectral analysis of the corresponding preconditioned systems is given in Section 4. A comparison of the computational complexity of the methods is found in Section 5 and numerical illustrations of their efficiency are given in Section 6. We present some conclusions in Section 7.

2 Solution of general complex matrix systems

As described in the later sections, in Methods **M1** and **M2** complex-valued systems arise. In this section we present two preconditioning methods to solve complex algebraic linear systems of the form $(A + iB)(\boldsymbol{x} + i\boldsymbol{y}) = \boldsymbol{f} + i\boldsymbol{g}$. Here A and B denote generic matrices that are spd and spsd, correspondingly. In the sequel I denotes an identity matrix of appropriate size.

2.1 The complex-to-real (C-to-R) method

Complex systems can be solved by the use of the complex-to-real (C-to-R) framework, rewriting the complex system in a two-by-two block matrix form with real matrices, that is, $(A + iB)(\boldsymbol{x} + i\boldsymbol{y}) = \boldsymbol{f} + i\boldsymbol{g}$ is solved via

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix}. \tag{5}$$

As a side note, (5) is not a unique way to write a complex system in a real form. Depending on the properties of A and B we could equivalently use $\begin{bmatrix} B & -A \\ A & B \end{bmatrix} \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{x} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix}$.

As it has been shown in earlier works, the solution of (5) can be done in several ways, but very efficiently using the PRESB preconditioner (see, e.g., [10, 11]),

$$\mathcal{P} = \begin{bmatrix} A & -B \\ B & A + 2B \end{bmatrix}.$$

The matrix $\mathcal{P}$ possesses the factorization

$$\mathcal{P} = \begin{bmatrix} A & -B \\ B & A + 2B \end{bmatrix} = \begin{bmatrix} I & -I \\ 0 & I \end{bmatrix} \begin{bmatrix} A + B & 0 \\ B & A + B \end{bmatrix} \begin{bmatrix} I & I \\ 0 & I \end{bmatrix}$$

and we see that apart from some vector operations and one multiplication with the block B, the solution of a system with $\mathcal{P}$ involves two solutions with the matrix $A+B$. Although the computations are straightforward, for completeness we include the analysis of the spectrum of the generalized eigenvalue problem

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} A & -B \\ B & A + 2B \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

Computation shows that

$$(1 - \lambda) \begin{bmatrix} A & -B \\ B & A + 2B \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 2B \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

Clearly, for $\boldsymbol{y} = \mathcal{N}(B)$, $\mathcal{N}(B)$ being the null space of B, and any $\boldsymbol{x}$ we see that $\lambda = 1$. For $\lambda \neq 1$ it holds that

$$(1 - \lambda)\boldsymbol{y}^T(BA^{-1}B + A + 2B)\boldsymbol{y} = 2\boldsymbol{y}^T B\boldsymbol{y}.$$

Transforming the latter equality by multiplying by $I = A^{1/2}A^{-1/2}$ from left and right and denoting $\widehat{B} = A^{-1/2}BA^{-1/2}$ and $\widehat{\boldsymbol{y}} = A^{1/2}\boldsymbol{y}$, we obtain

$$(1 - \lambda)\widehat{\boldsymbol{y}}^T(\widehat{B}^2 + I + 2\widehat{B})\widehat{\boldsymbol{y}} = 2\widehat{\boldsymbol{y}}^T\widehat{B}\widehat{\boldsymbol{y}}, \quad \text{i.e.,} \quad 1 - \lambda = \frac{2\widehat{\boldsymbol{y}}^T\widehat{B}\widehat{\boldsymbol{y}}}{\widehat{\boldsymbol{y}}^T(\widehat{B} + I)^2\widehat{\boldsymbol{y}}} \leq \frac{1}{2},$$

that is, $\frac{1}{2} \leq \lambda \leq 1$. Hence, Krylov subspace methods, applied to $\mathcal{A}$, preconditioned by $\mathcal{P}$, converge rapidly and with a convergence speed that holds uniformly with respect to all parameters, including the discretization parameter. The method is applicable also for the case when $B \neq B^T$, provided that $\mathcal{N}(A) \cap \mathcal{N}(B) = \emptyset$, and also when B is complex, as discussed in Section 4.

2.2 A direct real-valued matrix preconditioning form

Consider a generic complex matrix $A + iB$, where A and B fulfill the assumptions made above. As a preconditioner, let consider the real matrix $A + B$ and the corresponding generalized eigenvalue problem

$$(A + iB)(\boldsymbol{x} + i\boldsymbol{y}) = (\lambda + i\mu)(A + B)(\boldsymbol{x} + i\boldsymbol{y}). \tag{6}$$

We multiply from both sides by $I = (A+B)^{1/2}(A+B)^{-1/2}$, introduce $\widehat{A} = (A+B)^{-1/2}A(A+B)^{-1/2}$, $\widehat{B} = (A + B)^{-1/2}B(A + B)^{-1/2}$, $\widehat{\boldsymbol{x}} = (A + B)^{\frac{1}{2}}\boldsymbol{x}$ and $\widehat{\boldsymbol{y}} = (A + B)^{\frac{1}{2}}\boldsymbol{y}$, and obtain the equivalent eigenvalue problem

$$(\widehat{A} + i\widehat{B})(\widehat{\boldsymbol{x}} + i\widehat{\boldsymbol{y}}) = (\lambda + i\mu)(\widehat{\boldsymbol{x}} + i\widehat{\boldsymbol{y}}).$$

Rewriting it in real block matrix form, we have

$$\begin{bmatrix} \widehat{A} & -\widehat{B} \\ \widehat{B} & \widehat{A} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{x}} \\ \widehat{\boldsymbol{y}} \end{bmatrix} = \begin{bmatrix} \lambda & -\mu \\ \mu & \lambda \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{x}} \\ \widehat{\boldsymbol{y}} \end{bmatrix}.$$

Let $\|\widehat{\boldsymbol{x}}\| = 1$ and $\|\widehat{\boldsymbol{y}}\| = 1$, Multiplying from the left the first equation by $\widehat{\boldsymbol{x}}^T$ and the second by $\widehat{\boldsymbol{y}}^T$, and summing up we obtain

$$\widehat{\boldsymbol{x}}^T\widehat{A}\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{y}}^T\widehat{A}\widehat{\boldsymbol{y}} = \lambda(\widehat{\boldsymbol{x}}^T\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{y}}^T\widehat{\boldsymbol{y}}) = 2\lambda. \tag{7}$$

By multiplying (6) by $-i$ and performing analogous operations, we obtain

$$\widehat{\boldsymbol{x}}^T\widehat{B}\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{y}}^T\widehat{B}\widehat{\boldsymbol{y}} = \mu(\widehat{\boldsymbol{x}}^T\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{y}}^T\widehat{\boldsymbol{y}}) = 2\mu. \tag{8}$$

From (7),(8), and utilizing the relation $\widehat{A} + \widehat{B} = I$, it follows that

$$\lambda + \mu = \frac{1}{2}\left(\widehat{\boldsymbol{x}}^T(\widehat{A} + \widehat{B})\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{y}}^T(\widehat{A} + \widehat{B})\widehat{\boldsymbol{y}}\right) = 1.$$

Hence, the eigenvalues are located on the line between $\lambda = 0, \mu = 1$ and $\lambda = 1, \mu = 0$ in the complex plane. Let $\lambda = s, \mu = 1 - s$ for the left part of the line and $\lambda = 1 - s, \mu = s$ for the right part of the line, $0 < s < \frac{1}{2}$. This dependence on a single variable implies that a generalized conjugate gradient method, such as GMRES, converges corresponding to a spectral condition number 2, as for the PRESB method. In this case there are two alternatives. The first one is to solve the original complex system $A + i\,B$, preconditioned by $A+B$. In this form the method requires complex arithmetic but only one scalar system with real matrix to be solved during each outer iteration. The second one is to rewrite the complex system in real form and precondition it by a block-diagonal matrix with $A + B$ as diagonal blocks. In this case the eigenvalues of the preconditioned system lie on two symmetric parts of the line, similar to the case of two symmetric ovals, described in [12](Section 5.4*), the number of iterations is doubled. Therefore, the computational complexity of the two methods is expected to be similar to that of PRESB.

To our knowledge, the idea to precondition $A + iB$ by $A + B$ is utilised in [19].

3 Description and implementation of methods M1, M2 and M3

We return to the setting arising in the optimal control problem. The system to solve is given in (4), namely,

$$\mathcal{A}\begin{bmatrix}\boldsymbol{u}\\ \widetilde{\boldsymbol{v}}\end{bmatrix}=\begin{bmatrix}A & -\widetilde{C}^*\\ \widetilde{C} & A\end{bmatrix}\begin{bmatrix}\boldsymbol{u}\\ \widetilde{\boldsymbol{v}}\end{bmatrix}=\begin{bmatrix}\boldsymbol{f}\\ \widetilde{\boldsymbol{g}}\end{bmatrix},$$

where $\widetilde{C}=\sqrt{\beta}C=\widetilde{B}+i\widetilde{\omega}A$, $\widetilde{B}=\sqrt{\beta}B$, $\widetilde{\omega}=\sqrt{\beta}\omega$, $\widetilde{\boldsymbol{v}}=-\sqrt{\beta}\boldsymbol{v}$, $\widetilde{\boldsymbol{g}}=\sqrt{\beta}\boldsymbol{g}$, $\boldsymbol{f}=A\boldsymbol{u}_d$ and $\boldsymbol{g}$ is related to the boundary conditions. In the rest of this section, for simplicity, we omit $\widetilde{}$ for the vectors $\boldsymbol{v}$ and $\boldsymbol{g}$.

3.1 Method M1

In Method **M1**, for the matrix $\mathcal{A}$ in (4), we consider the PRESB preconditioner

$$\mathcal{P}=\begin{bmatrix}A & -\widetilde{C}^*\\ \widetilde{C} & A+\widetilde{C}+\widetilde{C}^*\end{bmatrix}=\begin{bmatrix}A & -\widetilde{C}^*\\ \widetilde{C} & A+\widetilde{B}+\widetilde{B}^T\end{bmatrix},$$

which possesses the exact factorization

$$\mathcal{P}=\begin{bmatrix}I & -I\\ 0 & I\end{bmatrix}\begin{bmatrix}A+\widetilde{C} & 0\\ \widetilde{C} & A+\widetilde{C}^*\end{bmatrix}\begin{bmatrix}I & I\\ 0 & I\end{bmatrix}.$$

The latter expression shows that $\mathcal{P}$ is nonsingular and an action of $\mathcal{P}^{-1}$, besides a matrix-vector multiplication with $\widetilde{C}$, involves a solution with the complex matrices $A+\widetilde{B}+i\widetilde{\omega}A$ and $A+\widetilde{B}^T-i\widetilde{\omega}A$ for which the methods in Section 2 can be used. For completeness, the process of performing the action $\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix}=\mathcal{P}^{-1}\begin{bmatrix}\boldsymbol{p}\\ \boldsymbol{q}\end{bmatrix}$ is given in Algorithm 1.

Algorithm 1 Method **M1**

```
1: Let $F_1=A+\widetilde{B}+i\widetilde{\omega}A$, $F_2=A+\widetilde{B}^T-i\widetilde{\omega}A$, $\widetilde{B}=\sqrt{\beta}B$, $\widetilde{\omega}=\sqrt{\beta}\omega$
2: Solve $F_1\boldsymbol{h}=\boldsymbol{p}+\boldsymbol{q}$
3: Solve $F_2\boldsymbol{y}=\boldsymbol{q}-\widetilde{C}\boldsymbol{h}$
4: Compute $\boldsymbol{x}=\boldsymbol{h}-\boldsymbol{y}$
```

3.2 Method M2

We start with the following observation. Consider the matrix $\mathcal{A}$ in (4) and assume that the off-diagonal block C is real. By definition, its exact Schur complement S is equal to $S=A+CA^{-1}C^T$. Assume that $\widetilde{S}=(A+C)A^{-1}(A+C^T)$, suggested in [13], is a good quality approximation of S. Computation shows that for the following factorized preconditioner there holds

$$\widetilde{\mathcal{P}} = \begin{bmatrix} A & 0 \\ C & (A+C)A^{-1}(A+C^T) \end{bmatrix} \begin{bmatrix} I & A^{-1}C^T \\ 0 & I \end{bmatrix} = \begin{bmatrix} A & -C \\ C & A+C+C^T \end{bmatrix}. \tag{9}$$

Thus, $\widetilde{\mathcal{P}}$ coincides with the PRESB preconditioner. Solving straightforwardly systems with $\widetilde{\mathcal{P}}$ would mean that we have to solve with $A+C$ and $A+C^T$ and in addition to solve systems with A twice, while the complexity of PRESB is much lower.

In addition, in the case when the blocks C are complex and have the particular form $\widetilde{B}+i\widetilde{\omega}A$ we observe that S can be factorized in a matrix product form similarly as $\widetilde{S}$, enabling the possibility to solve systems with $\mathcal{A}$ directly, on the cost of solving with two blocks matrices of the size of A. To show the latter, we proceed as follows.

Recall that since B is symmetric, $S = (1+\widetilde{\omega}^2)A + \widetilde{B}A^{-1}\widetilde{B}$. Computation reveals that S possesses an exact complex-valued factorization of the form $S = H_1A^{-1}H_2$, where $H_1 = \sqrt{1+\widetilde{\omega}^2}A - i\widetilde{B}$ and $H_2 = \sqrt{1+\widetilde{\omega}^2}A + i\widetilde{B} = H_1^*$.

To define Method **M2** we use the exact block matrix factorization of $\mathcal{A}$,

$$\mathcal{A} = \begin{bmatrix} A & 0 \\ \widetilde{B}+i\widetilde{\omega}A & H_1 \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & A^{-1} \end{bmatrix} \begin{bmatrix} A & -(\widetilde{B}-i\widetilde{\omega}A) \\ 0 & H_2 \end{bmatrix}. \tag{10}$$

Then the exact inverse of $\mathcal{A}$ becomes

$$\mathcal{A}^{-1} = \begin{bmatrix} A^{-1} & A^{-1}(\widetilde{B}-i\widetilde{\omega}A)H_2^{-1} \\ 0 & H_2^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -AH_1^{-1}(\widetilde{B}+i\widetilde{\omega}A)A^{-1} & AH_1^{-1} \end{bmatrix}. \tag{11}$$

At a first glance the computation of the solution of the system $\mathcal{A}\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix}$ can be done by direct use of (11), thus, besides solutions with the matrices H_1 and H_2, it would also involve two solutions with A. We show now that solutions with A can be fully avoided. Consider first the matrix-vector multiplication arising from the second factor in (11):

$$\begin{bmatrix} I & 0 \\ -AH_1^{-1}(\widetilde{B}+i\widetilde{\omega}A)A^{-1} & AH_1^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ AH_1^{-1}(\boldsymbol{g} - (\widetilde{B}+i\widetilde{\omega}A)A^{-1}\boldsymbol{f}) \end{bmatrix}.$$

Note that since $H_1 = \sqrt{1+\widetilde{\omega}^2}A - i\widetilde{B}$ then

$$\begin{aligned} -AH_1^{-1}(\widetilde{B}+i\widetilde{\omega}A)A^{-1} &= -iAH_1^{-1}(-i\widetilde{B}+\sqrt{1+\widetilde{\omega}^2}A - (\sqrt{1+\widetilde{\omega}^2}-\widetilde{\omega})A)A^{-1} \\ &= -i(I - b_{\widetilde{\omega}}AH_1^{-1}), \end{aligned}$$

where $b_{\widetilde{\omega}} = \sqrt{1+\widetilde{\omega}^2} - \widetilde{\omega} = \frac{1}{\sqrt{1+\widetilde{\omega}^2}+\widetilde{\omega}} \le 1$. It follows that

$$\begin{aligned} \begin{bmatrix} I & 0 \\ -AH_1^{-1}(\widetilde{B}+i\widetilde{\omega}A)A^{-1} & AH_1^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{f} \\ -i\boldsymbol{f} + ib_{\widetilde{\omega}}AH_1^{-1}\boldsymbol{f} + AH_1^{-1}\boldsymbol{g} \end{bmatrix} \\ = \begin{bmatrix} \boldsymbol{f} \\ -i\boldsymbol{f} + iA\boldsymbol{h} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{f} \\ -i(\boldsymbol{f} - A\boldsymbol{h}) \end{bmatrix}, \end{aligned}$$

with

$$\boldsymbol{h} = H_1^{-1}(b_{\widetilde{\omega}}\boldsymbol{f} - i\boldsymbol{g}). \tag{12}$$

It follows from (11) that

$$\boldsymbol{v} = -iH_2^{-1}(\boldsymbol{f} - A\boldsymbol{h}). \tag{13}$$

To find the component $\boldsymbol{u}$, note now that

$$A^{-1}(\widetilde{B} - i\widetilde{\omega}A)H_2^{-1} = iA^{-1}(-i\widetilde{B} - \sqrt{1+\widetilde{\omega}^2}A + b_{\widetilde{\omega}}A)H_2^{-1} = -iA^{-1} + ib_{\widetilde{\omega}}H_2^{-1}.$$

Hence, it follows from (11) that

$$\begin{aligned}\boldsymbol{u} &= \begin{bmatrix} A^{-1} & A^{-1}(\widetilde{B} - i\widetilde{\omega}A)H_2^{-1}\end{bmatrix}\begin{bmatrix} A\boldsymbol{h} + \boldsymbol{f} - A\boldsymbol{h} \\ -i(\boldsymbol{f} - A\boldsymbol{h})\end{bmatrix} \\ &= \boldsymbol{h} + A^{-1}(\boldsymbol{f} - A\boldsymbol{h}) - A^{-1}(\boldsymbol{f} - A\boldsymbol{h}) + b_{\widetilde{\omega}}H_2^{-1}(\boldsymbol{f} - A\boldsymbol{h}),\end{aligned}$$

that is,

$$\boldsymbol{u} = \boldsymbol{h} + b_{\widetilde{\omega}}H_2^{-1}(\boldsymbol{f} - A\boldsymbol{h}) = \boldsymbol{h} + ib_{\widetilde{\omega}}\boldsymbol{v}.$$

The latter relations show that, besides a matrix-vector multiplication with A and some complex vector additions, the computation of the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ requires one solution with H_1 to compute the vector $\boldsymbol{h}$ and one solution with H_2 to compute $\boldsymbol{v}$. Both systems involve complex matrices, which can be solved directly or by an iterative method using one of the methods in Section 2. In Algorithm 2 we summarize the computations, needed to obtain the action of $\mathcal{A}^{-1}$ on a vector $\begin{bmatrix}\boldsymbol{f} \\ \boldsymbol{g}\end{bmatrix}$.

Algorithm 2 Method **M2**

```
1: Let $H_1 = \sqrt{1+\widetilde{\omega}^2}A - i\widetilde{B}$, $H_2 = H_1^* = \sqrt{1+\widetilde{\omega}^2}A + i\widetilde{B}$, $b_{\widetilde{\omega}} = \frac{1}{\sqrt{1+\widetilde{\omega}^2}+\widetilde{\omega}}$
2: Solve $H_1\boldsymbol{h} = b_{\widetilde{\omega}}\boldsymbol{f} - i\boldsymbol{g}$
3: Solve $H_2\boldsymbol{v} = -i(\boldsymbol{f} - A\boldsymbol{h})$
4: Compute $\boldsymbol{u} = \boldsymbol{h} + ib_{\widetilde{\omega}}\boldsymbol{v}$
```

Clearly, if we solve exactly systems with H_1 and H_2, we have a direct solution method for $\mathcal{A}$. In practice these are solved to some limited tolerance. It could then be efficient to embed the method in a defect-correction framework, applied at least once.

3.3 Method M3

To reduce the complexity due to solving systems with complex matrices, here we use the following approximation of S in real-valued factors, $G_1A^{-1}G_2 = (\sqrt{1+\widetilde{\omega}^2}A + \widetilde{B})A^{-1}(\sqrt{1+\widetilde{\omega}^2}A + \widetilde{B}^T) = (1+\widetilde{\omega}^2)A + \widetilde{B}A^{-1}\widetilde{B}^T + \sqrt{1+\widetilde{\omega}^2}(\widetilde{B} + \widetilde{B}^T)$. The corresponding approximate factorization of $\mathcal{A}$ becomes

$$\mathcal{A} \approx \mathcal{B} = \begin{bmatrix} A & 0 \\ C & G_1\end{bmatrix}\begin{bmatrix} A^{-1} & 0 \\ 0 & A^{-1}\end{bmatrix}\begin{bmatrix} A & -C^* \\ 0 & G_2\end{bmatrix}.$$

Hence,

$$\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix} = \mathcal{B}^{-1}\begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{g}\end{bmatrix} = \begin{bmatrix}A^{-1} & A^{-1}C^*G_2^{-1}\\ 0 & G_2^{-1}\end{bmatrix}\begin{bmatrix}I & 0\\ -AG_1^{-1}CA^{-1} & AG_1^{-1}\end{bmatrix}\begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{g}\end{bmatrix}.$$

Here,

$$\begin{aligned}-AG_1^{-1}CA^{-1} &= -A(\sqrt{1+\widetilde{\omega}^2}A+\widetilde{B})^{-1}(\widetilde{B}+i\widetilde{\omega}A)A^{-1}\\ &= -A(\sqrt{1+\widetilde{\omega}^2}A+\widetilde{B})^{-1}(\sqrt{1+\widetilde{\omega}^2}A+\widetilde{B}-c_{\widetilde{\omega}}A)A^{-1}\\ &= -I+c_{\widetilde{\omega}}AG_1^{-1},\end{aligned}$$

where $c_{\widetilde{\omega}} = \sqrt{1+\widetilde{\omega}^2} - i\widetilde{\omega}$. Thus, it holds that

$$-AG_1^{-1}CA^{-1}\boldsymbol{f} + AG_1^{-1}\boldsymbol{g} = -\boldsymbol{f} + AG_1^{-1}(c_{\widetilde{\omega}}\boldsymbol{f}+\boldsymbol{g}) = -(\boldsymbol{f}-A\boldsymbol{h}),$$

with $\boldsymbol{h} = G_1^{-1}(c_{\widetilde{\omega}}\boldsymbol{f}+\boldsymbol{g}))$. Hence, $\boldsymbol{v} = -G_2^{-1}(\boldsymbol{f}-A\boldsymbol{h})$. Further,

$$\boldsymbol{u} = \begin{bmatrix}A^{-1} & A^{-1}C^*G_2^{-1}\end{bmatrix}\begin{bmatrix}A\boldsymbol{h}+\boldsymbol{f}-A\boldsymbol{h}\\ -(\boldsymbol{f}-A\boldsymbol{h})\end{bmatrix}.$$

Note, that $A^{-1}C^*G_2^{-1} = A^{-1}(B^T - i\widetilde{\omega}A)G_2^{-1} = A^{-1}(\sqrt{1+\widetilde{\omega}^2}A + B^T - c_{\widetilde{\omega}}^*A)G_2^{-1} = A^{-1} - c_{\widetilde{\omega}}^*G_2^{-1}$, where $c_{\widetilde{\omega}}^* = \sqrt{1+\widetilde{\omega}^2} + i\widetilde{\omega}$. Accordingly, $\boldsymbol{u} = \boldsymbol{h} + c_{\widetilde{\omega}}^*G_2^{-1}(\boldsymbol{f}-A\boldsymbol{h}) = \boldsymbol{h} - c_{\widetilde{\omega}}^*\boldsymbol{v}$.

We sum up the computational sequence in Algorithm 3.

Algorithm 3 Method **M3**

```
1: Let $G_1 = \sqrt{1+\widetilde{\omega}^2}A+\widetilde{B}$, $G_2 = G_1^T = \sqrt{1+\widetilde{\omega}^2}A+\widetilde{B}^T$, $c_{\widetilde{\omega}}^* = \sqrt{1+\widetilde{\omega}^2}+i\widetilde{\omega}$
2: Solve $G_1\boldsymbol{h} = c_{\widetilde{\omega}}\boldsymbol{f}+\boldsymbol{g}$
3: Solve $G_2\boldsymbol{v} = A\boldsymbol{h}-\boldsymbol{f}$
4: Compute $\boldsymbol{u} = \boldsymbol{h} - c_{\widetilde{\omega}}^*\boldsymbol{v}$
```

Thus, in this method we solve two inner systems with real matrices.

4 Spectral analysis

As a general remark, when we consider a matrix of the form $\begin{bmatrix}A & B\\ C & D\end{bmatrix}$ with square blocks of size n and a preconditioner of the form $\begin{bmatrix}A & B\\ P & Q\end{bmatrix}$ then the generalized eigenvalue problem

$$\begin{bmatrix}A & B\\ C & D\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} = \lambda\begin{bmatrix}A & B\\ P & Q\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix}$$

has at least n eigenvalues, equal to 1.

4.1 Method M1

Let λ be an eigenvalue of $\mathcal{P}^{-1}\mathcal{A}$. There holds that

$$(1-\lambda)\mathcal{P}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} = (\mathcal{P}-\mathcal{A})\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} = \begin{bmatrix}0 & 0\\ 0 & \widetilde{B}+\widetilde{B}^T\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix}.$$

Analogously to PRESB in the C-to-R setting, if $\boldsymbol{y}\in\mathcal{N}(B+B^T)$ and for any $\boldsymbol{x}$, then $\lambda=1$. For $\lambda\neq 1$ we must have $A\boldsymbol{x}=C^*\boldsymbol{y}$ and $(1-\lambda)\boldsymbol{y}^T(CA^{-1}C^*+A+C+C^*)\boldsymbol{y}=\boldsymbol{y}^T(\widetilde{B}+\widetilde{B}^T)\boldsymbol{y}$. The latter is equivalent to

$$(1-\lambda)\boldsymbol{y}^T((1+\widetilde{\omega}^2)A+\widetilde{B}A^{-1}\widetilde{B}^T+\widetilde{B}+\widetilde{B}^T)\boldsymbol{y}=\boldsymbol{y}^T(\widetilde{B}+\widetilde{B}^T)\boldsymbol{y}.$$

So, λ is real and $0<\lambda\leq 1$. By the assumption that A is spd and transforming the last equality by $A^{-1/2}$ from both sides we get

$$(1-\lambda)\boldsymbol{y}^T((1+\widetilde{\omega}^2)I+\widehat{B}\widehat{B}^T+\widehat{B}+\widehat{B}^T)\boldsymbol{y}=\boldsymbol{y}^T(\widehat{B}+\widehat{B}^T)\boldsymbol{y}, \tag{14}$$

where $\widehat{B}=A^{-1/2}\widetilde{B}A^{-1/2}$. Let

$$\alpha=\frac{\boldsymbol{y}^T(\widehat{B}+\widehat{B}^T)\boldsymbol{y}}{\boldsymbol{y}^T((1+\widetilde{\omega}^2)I+\widehat{B}\widehat{B}^T+\widehat{B}+\widehat{B}^T)\boldsymbol{y}}\leq\frac{1}{2+\frac{\widetilde{\omega}^2}{1+\|\widehat{B}\widehat{B}^T\|^2}}.$$

The latter holds since $\widehat{B}+\widehat{B}^T\leq I+\widehat{B}\widehat{B}^T$. Hence, $0<\alpha<\frac{1}{2}$ and it follows that

$$1-\lambda\leq\alpha,\quad\text{or,}\quad\frac{1}{2}<1-\alpha\leq\lambda\leq 1.$$

We note, that an increase in the fraction above leads to a decay in α. Therefore, the eigenvalues λ cluster at unity when α is small. Further, note that as a function of β, since $\|\widehat{B}+\widehat{B}^T\|\approx|O(\sqrt{\beta})|$, α is small for small values of β.

Method **M1** involves solutions with complex systems of the form

$$(A+\widetilde{B}+i\widetilde{\omega}A)(\boldsymbol{x}+i\boldsymbol{y})=\boldsymbol{f}+i\boldsymbol{g}.$$

It can be handled via the C-to-R method and rewritten in a real form as

$$\begin{bmatrix}A+\widetilde{B} & -\widetilde{\omega}A\\ \widetilde{\omega}A & A+\widetilde{B}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix}=\begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{g}\end{bmatrix}.$$

As already mentioned, the solution of it can be done efficiently using some Krylov subspace method, preconditioned by the PRESB preconditioner

$$\mathcal{P}=\begin{bmatrix}A+\widetilde{B} & -\widetilde{\omega}A\\ \widetilde{\omega}A & (1+2\widetilde{\omega})A+\widetilde{B}\end{bmatrix}.$$

The generalized eigenvalue problem in this case has the form

$$\begin{bmatrix} A+\widetilde{B} & -\widetilde{\omega}A \\ \widetilde{\omega}A & A+\widetilde{B} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} A+\widetilde{B} & -\widetilde{\omega}A \\ \widetilde{\omega}A & (1+2\widetilde{\omega})A+\widetilde{B} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

Following the same routine as in Section 2.2 we see again that we have $\lambda = 1$ for $\boldsymbol{y} = \boldsymbol{0}$ and any $\boldsymbol{x}$. For $\lambda \neq 1$ it holds that $(1-\lambda)\boldsymbol{y}^T(\widetilde{\omega}^2 A(A+\widetilde{B})^{-1}A + (A+\widetilde{B}) + 2\widetilde{\omega}A)\boldsymbol{y} = 2\widetilde{\omega}\boldsymbol{y}^T A\boldsymbol{y}$. Transforming the latter equality by multiplying by $(A+\widetilde{B})^{-1/2}$ from left and right and denoting $\widehat{A} = (A+\widetilde{B})^{-1/2}A(A+\widetilde{B})^{-1/2}$ we obtain $(1-\lambda)\boldsymbol{y}^T(\widetilde{\omega}^2\widehat{A}^2 + I + 2\widetilde{\omega}\widehat{A})\boldsymbol{y} = 2\widetilde{\omega}\boldsymbol{y}^T\widehat{A}\boldsymbol{y}$, i.e.,

$$1-\lambda = \frac{2\widetilde{\omega}\boldsymbol{y}^T\widehat{A}\boldsymbol{y}}{\boldsymbol{y}^T(\widetilde{\omega}\widehat{A}+I)^2\boldsymbol{y}} \leq \frac{1}{2},$$

this is, $\frac{1}{2} \leq \lambda \leq 1$. For small or large $\widetilde{\omega}$, the eigenvalues cluster at unity. Note that $\widetilde{\omega} = \sqrt{\omega\beta}$ and β is small in practice, thus, $\widetilde{\omega}$ is usually small. The convergence rate of the preconditioned Krylov subspace methods remains fast and independent of all problem, method and discretization parameters.

4.2 Method M2

In general, **M2** is a direct method, provided that we solve exactly systems with the matrices H_1 and H_2. As already stated, when systems with H_1 and H_2 are not solved exactly, a defect-correction step is suggested, outlined in Section 4.3.

4.3 Method M3

In Method **M3** we precondition $\mathcal{A}$ by $\mathcal{B}$, where

$$\mathcal{B} = \begin{bmatrix} A & 0 \\ C & G_1 \end{bmatrix} \begin{bmatrix} I & -A^{-1}C^* \\ 0 & A^{-1}G_2 \end{bmatrix} = \begin{bmatrix} A & -C^* \\ C & G_1A^{-1}G_2 - CA^{-1}C^* \end{bmatrix}.$$

Recall that $CA^{-1}C^* = \widetilde{\omega}^2 A + \widehat{B}A^{-1}\widehat{B}^T$ and

$$G_1A^{-1}G_2 = (1+\widetilde{\omega}^2)A + \widehat{B}A^{-1}\widehat{B}^T + \sqrt{1+\widetilde{\omega}^2}(\widehat{B}+\widehat{B}^T).$$

Accordingly, $G_1A^{-1}G_2 - CA^{-1}C^* = A + \sqrt{1+\widetilde{\omega}^2}(\widehat{B}+\widehat{B}^T)$. The corresponding generalized eigenvalue problem becomes

$$(1-\lambda)\mathcal{B}\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = (\mathcal{B}-\mathcal{A})\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{1+\widetilde{\omega}^2}(\widehat{B}+\widehat{B}^T) \end{bmatrix}\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

We see that $\lambda = 1$ for $\boldsymbol{y} \in \mathcal{N}(\widehat{B}+\widehat{B}^T)$ and any $\boldsymbol{x}$. For $\lambda \neq 1$, then $A\boldsymbol{x} = C^*\boldsymbol{y}$ and

$$(1-\lambda)\left((1+\widetilde{\omega}^2)A + \widehat{B}A^{-1}\widehat{B}^T + \sqrt{1+\widetilde{\omega}^2}(\widehat{B}+\widehat{B}^T)\right)\boldsymbol{y} = \sqrt{1+\widetilde{\omega}^2}(\widehat{B}+\widehat{B}^T)\boldsymbol{y}.$$

Consequently, λ is positive and

$$(\frac{1}{\lambda} - 1)((1+\widetilde{\omega}^2)A + \widehat{B}A^{-1}\widehat{B}^T)\boldsymbol{y} = \sqrt{1+\widetilde{\omega}^2}(\widehat{B} + \widehat{B}^T)\boldsymbol{y}.$$

Denote $\widehat{\alpha} = \dfrac{\boldsymbol{y}^T\sqrt{1+\widetilde{\omega}^2}(\widehat{B}+\widehat{B}^T)\boldsymbol{y}}{\boldsymbol{y}^T((1+\widetilde{\omega}^2)A + \widehat{B}A^{-1}\widehat{B}^T)\boldsymbol{y}}$. Clearly, $0 \leq \widehat{\alpha} \leq 1$. It now follows straightforwardly, that

$$\frac{1}{\lambda} - 1 \leq \widehat{\alpha}, \quad \text{that is,} \quad \frac{1}{1+\widehat{\alpha}} \leq \lambda \leq 1.$$

We see that, similarly to the result of the analysis of Method **M1**, $\widehat{\alpha}$ gets smaller for small values of β.

5 Comparison of the computational complexity of the methods

Method **M1** involves two solutions with the complex matrices F_1 and F_2 at each outer iteration, but its convergence is fast, typically about 6-8 iterations for a relative stopping tolerance of order 10^{-6} to 10^{-8}. The two inner systems are complex. Each of the two methods from Section 2 can be used, involving then four, respectively, two solutions with a real-valued matrix. For the second method it has been shown in Section 2 that the eigenvalues of the preconditioned matrix are complex, leading typically to a double computational complexity as compared with solving systems with real matrices.

Method **M2** does not need any outer iteration if the two arising systems with the matrices H_1 and H_2, which are complex, are solved exactly or iteratively to a very fine stopping tolerance. Nonetheless, in practice it can be advisable to solve these systems by iteration to some practical accuracy, which can save computational effort. One must then also use an outer iteration method, which can be a simple defect-correction method. Namely, after the first iteration, where we compute a solution $\begin{bmatrix}\boldsymbol{u}_0\\ \boldsymbol{v}_0\end{bmatrix}$ of $\mathcal{A}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix} = \begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{g}\end{bmatrix}$, we solve

$$\mathcal{A}\left(\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix} - \begin{bmatrix}\boldsymbol{u}_0\\ \boldsymbol{v}_0\end{bmatrix}\right) = \begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{g}\end{bmatrix} - \mathcal{A}\begin{bmatrix}\boldsymbol{u}_0\\ \boldsymbol{v}_0\end{bmatrix},$$

again to a practical accuracy. Depending on the chosen stopping tolerance, say, as a square root of the desired accuracy, it can suffice with just a single such defect-correction step.

Method **M3** requires solution with the two real-valued matrices G_1 and G_2 in each outer iteration, which can save computational effort. The rate of convergence of Methods **M1** and **M3** is similar.

It is possible in all three methods to adjust the tolerance to see how much inner iterations can be saved without substantial increase in the number of outer iterations.

A further saving of computational effort can be achieved if one uses a coarse discretization for the inner iterations. This is similar to the use of a combination of coarse and fine meshes when solving nonlinear problems, see [14] and in the context of non-smooth Newton method for optimal control problems in [11]. It is also possible to apply a domain-decomposition method, where a fine mesh is used for each subdomain but a coarser mesh can be utilized for the coupling of the subdomains, see e.g., [15].

6 Numerical results

The performance of the preconditioning methods **M1**, **M2** and **M3** is illustrated numerically on the optimal control problem from Section 1 with a constraint, given by discrete time-harmonic eddy current problem as described in Problem 1.

Problem 1. *Consider the distributed optimal control problem with the following time-periodic PDE constraint,*

$$\begin{aligned} &\sigma\frac{\partial u}{\partial t} + \nabla\times(\frac{1}{\mu}\,\nabla\times u) + \epsilon u = \tau(\boldsymbol{x})v && in \quad \Omega\times[0,T],\\ &u\times n = 0 && in \quad \partial\Omega\times[0,T],\\ &u(\boldsymbol{x},0) = u(\boldsymbol{x},T). \end{aligned} \tag{15}$$

As already mentioned, u is the magnetic potential field, the problem parameter μ is the permeability of the media and σ is the conductivity. The parameter ϵ is positive and small, and is added for regularization purposes to mitigate the large null space of the original curl-curl problem.

As discussed, e.g., in [4, 5, 7], in the time-harmonic setting, $u = Re(\widehat{u}e^{i\omega t})$ and $v = Re(\widehat{v}e^{i\omega t})$. Here ω is the angular frequency. With that assumption the state equation becomes

$$\begin{aligned} &i\omega\sigma\widehat{u} + \nabla\times(\tfrac{1}{\mu}\,\nabla\times\widehat{u}) + \epsilon\widehat{u} = \tau(\boldsymbol{x})\widehat{v} && in \quad \Omega\times[0,T]\\ &\widehat{u}\times n = 0 && in \quad \partial\Omega\times[0,T], \end{aligned} \tag{16}$$

where $\widehat{u}$ and $\widehat{v}$ are complex functions.

For the experiments we choose $\Omega = (0,1)^3$, the parameters as $\sigma = \mu = 1$, the source term $\tau = 1$, and the regularization parameter $\epsilon = 10^{-6}$. We choose also $u_d = 1$ and $g = 0$.

The space discretization is done using lowest-order Nédélec-I finite elements [16]. The numerical experiments are performed on a laptop equipped with 12-core processor Intel Core i7-8750H, 2.2 GHz and 32 GB memory. All the computations are performed in an in-house C++ code using the external library COLAMD [17].

Setting of the experiments

We test the performance of the three methods solving the compressed and scaled system (4) for various values of the time-frequency ω, the regularization parameter β, the discontinuous coefficients μ in the stiffness (permeability) matrix as well as for several levels of discretization. The discretization levels $l = 0, 1, 2, 3$, reported in the tables, correspond to the number of spacial degrees of freedom being 8632, 75180, 626408, 5112080, respectively.

The linear systems in Methods **M1** and **M3** are solved using an inner-outer iterative solution framework. To cope with the inner solutions of the blocks in the preconditioners, which act as variable preconditioning, as an outer solution method we use the flexible GMRES (FGMRES) method, [20], with relative stopping tolerance for the preconditioned residual 10^{-8}.

Layout **M1**: The arising inner systems with the matrices $A+\widetilde{C}$ and $A+\widetilde{C}^*$ are solved by the methods, described in Section 2, either in C-to-R form using PRESB as a precontidioner (Section 2.1) or directly the complex system with a real preconditioner (Section 2.2).

In the C-to-R case, the inner systems with the complex-valued blocks F_1 and F_2, after converting them in two-by-two block form, are solved by PRESB-preconditioned FGMRES as outlined in Section 3.2, with stopping tolerance 10^{-2}. For the solution of the most inner systems with matrices $(1+2\widetilde{\omega})A+\widetilde{B}$ and $(1+2\widetilde{\omega})A+\widetilde{B}^T$ we use a direct method on level 0 and from level 1 on we use conjugate gradient iterations preconditioned by one V-cycle of a geometric multigrid using three symmetric sweeps of the Arnold-Falk-Winther patch smoother, cf. [18], with stopping criterion 10^{-2}. We refer to this method as **C-to-R**.

When the systems $A+\widetilde{C}$ and $A+\widetilde{C}^*$ are solved by the method from Section 2.2, the method is referred to as **Real**. In this case we solve the systems as complex with a real preconditioner. Only in this case the outer iterative solver is the Generalized Conjugate Residual (GCR) method, cf, e.g., [21], (Algorithm 3.1 in [22]).

Layout **M3**: As shown in Algorithm 3, each outer iteration of **M3** requires two solutions of systems with real and spd matrices G_1 and G_2 with complex right-hand sides. The latter in real arithmetic leads to solutions of four real spd systems. These are solved by the CG method, preconditioned again with one V-cycle of the geometric multigrid, up to the relative tolerance 10^{-2}.

Layout **M2**: **M2** does not require an outer solver, cf. Algorithm 2. We include tests when H_1 and H_2 are solved iteratively by the PRESB-preconditioned FGMRES with a stopping tolerance 10^{-8}. The inner systems are solved as the most inner systems in **M1**, also with a stopping tolerance of 10^{-2}.

Performance results

The performance of **M1** is presented in Tables 1 and 2. This preconditioner with inner **M1:C-to-R** solver has already been tested on a similar example and reported in [7]. Here, in Table 1, in addition to the results for the **C-to-R**-PRESB preconditioner, we present also results using the real-valued preconditioner from Section 2.2 (**Real**) for levels 0, 1, and 2. The iteration counts are in a form 'X(Y)', where 'X' is the number of outer FGMRES iterations and 'Y' is the average PRESB-preconditioned FGMRES. We do not report the most inner average MG-preconditioned CG iterations as they are nearly always 1.

In Tables 3 and 4 we report results of the new method **M2**. The presented outer iteration counts are the sum of the outer iterations for H_1 and H_2. In Table 3 we include results for both approaches to solve the complex systems, **C-to-R** and **Real**.

Table 1: **M1**: Iteration counts for various values of β, ω and l, $\mu = 1$

β	l	ω									
		10^{-2}		10^{0}		10^{2}		10^{4}		10^{6}	
		C-to-R	**Real**	**C-to-R**	**Real**	**C-to-R**	**Real**	**C-to-R**	**Real**	**C-to-R**	**Real**
10^{-10}	0	6(1)	6(1)	6(1)	6(1)	6(1)	6(1)	6(2)	6(1)	4(1)	4(3)
	1	9(1)	9(1)	9(1)	9(1)	9(1)	9(1)	9(2)	9(2)	4(2)	4(3)
	2	10(1)	10(1)	10(1)	10(1)	10(1)	10(1)	10(2)	10(2)	6(3)	6(5)
10^{-8}	0	10(1)	10(1)	10(1)	10(1)	10(1)	11(1)	9(3)	9(5)	4(2)	4(3)
	1	11(1)	11(1)	11(1)	11(1)	11(1)	11(1)	10(3)	10(5)	4(1)	5(4)
	2	11(1)	11(1)	11(1)	11(1)	11(1)	11(1)	10(3)	10(6)	6(3)	5(6)
10^{-6}	0	10(1)	10(1)	10(1)	11(1)	11(2)	11(2)	7(3)	8(10)	4(2)	4(3)
	1	10(1)	10(1)	11(1)	11(1)	11(2)	12(2)	7(3)	8(12)	4(2)	5(4)
	2	11(1)	11(1)	11(1)	11(1)	11(2)	12(2)	7(3)	8(12)	6(3)	6(6)
10^{-2}	0	9(1)	9(1)	9(2)	10(2)	8(3)	8(11)	6(4)	8(12)	4(2)	4(3)
	1	9(2)	9(2)	9(4)	10(4)	8(3)	9(13)	6(4)	8(15)	4(2)	5(4)
	2	9(2)	10(2)	9(4)	10(4)	8(6)	8(21)	7(3)	8(15)	6(3)	6(6)

Table 2: **M1**: Iteration counts for varying β, μ and l, $\omega = 1$ (only C-to-R used)

β	l	$1/\mu$				
		10^{-8}	10^{-4}	10^{0}	10^{4}	10^{8}
10^{-10}	0	1(1)	2(1)	6(1)	9(1)	3(1)
	1	1(1)	2(1)	9(1)	9(2)	5(2)
	2	1(1)	2(1)	10(1)	9(2)	7(2)
10^{-8}	0	2(1)	2(1)	10(1)	5(1)	3(1)
	1	2(1)	2(1)	11(1)	6(2)	5(2)
	2	2(1)	3(1)	11(1)	7(2)	7(2)
10^{-6}	0	2(1)	3(1)	10(1)	5(1)	4(1)
	1	2(1)	3(1)	11(1)	5(2)	5(2)
	2	2(1)	4(1)	11(1)	7(2)	7(3)
10^{-4}	0	3(1)	4(1)	11(1)	5(1)	5(1)
	1	3(1)	5(1)	12(1)	6(2)	6(2)
	2	3(1)	7(1)	12(2)	7(3)	7(3)
10^{-2}	0	2(2)	6(2)	9(2)	3(2)	3(2)
	1	2(2)	9(2)	9(4)	4(4)	4(4)
	2	2(2)	10(2)	9(4)	5(4)	5(4)

Tables 5 and 6 contain the performance results of the new method **M3**. Here we use only the **C-to-R** framework.

Table 7 shows timing and memory usage resilts for the different methods. As the

Table 3: **M2**: Iteration counts for varying β, ω and l, $\mu = 1$

β	l	ω									
		10^{-2}		10^{0}		10^{2}		10^{4}		10^{6}	
		C-to-R	**Real**	**C-to-R**	**Real**	**C-to-R**	**Real**	**C-to-R**	**Real**	**C-to-R**	**Real**
10^{-10}	0	13(1)	12(1)	13(1)	12(1)	13(1)	12(1)	12(1)	12(1)	7(1)	8(1)
	1	19(1)	19(1)	19(1)	19(1)	19(1)	19(1)	18(1)	19(1)	9(1)	10(1)
	2	21(1)	32(1)	21(1)	32(1)	21(1)	32(1)	20(1)	31(1)	13(1)	14(1)
	3	21(1)	28(1)	21(1)	28(1)	21(1)	28(1)	21(1)	28(1)	19(1)	22(1)
10^{-8}	0	21(1)	27(1)	21(1)	27(1)	21(1)	27(1)	19(1)	22(1)	7(1)	8(1)
	1	22(1)	36(1)	22(1)	36(1)	22(1)	36(1)	21(1)	34(1)	9(1)	10(1)
	2	23(1)	40(1)	23(1)	40(1)	23(1)	40(1)	21(1)	38(1)	13(1)	14(1)
	3	23(1)	41(1)	23(1)	41(1)	23(1)	41(1)	22(1)	40(1)	19(1)	22(1)
10^{-6}	0	21(1)	39(1)	21(1)	39(1)	21(1)	39(1)	19(1)	26(1)	7(1)	8(1)
	1	22(1)	41(1)	22(1)	41(1)	22(1)	41(1)	21(1)	34(1)	9(1)	10(1)
	2	23(1)	42(1)	23(1)	42(1)	23(1)	42(1)	21(1)	38(1)	13(1)	14(1)
	3	25(2)	43(1)	25(2)	43(1)	25(2)	43(1)	22(1)	40(1)	19(1)	22(1)
10^{-4}	0	23(1)	34(1)	23(1)	34(1)	23(1)	36(1)	19(1)	26(1)	7(1)	8(1)
	1	26(2)	38(1)	26(2)	38(1)	25(1)	38(1)	21(1)	34(1)	13(1)	10(1)
	2	23(2)	40(1)	23(2)	40(1)	23(2)	40(1)	19(1)	38(1)	7(1)	14(1)
	3	25(2)	42(1)	25(2)	42(1)	23(2)	44(1)	22(1)	40(1)	19(1)	22(1)
10^{-2}	0	19(1)	20(1)	19(1)	20(1)	23(1)	34(1)	19(1)	26(1)	7(1)	8(1)
	1	20(2)	30(1)	20(2)	30(1)	24(1)	38(1)	21(1)	34(1)	9(1)	10(1)
	2	21(2)	34(1)	21(2)	34(1)	25(2)	40(1)	21(1)	38(1)	13(1)	14(1)
	3	21(2)	38(1)	21(2)	37(1)	24(2)	43(1)	22(1)	40(1)	19(2)	22(1)

number of inner iterations may vary, we present the minimum and maximum time required to assembly all necessary matrices and to solve the resulting linear system of equations with matrix $\mathcal{A}$. From the timing results in Table 7 we see that the **M2** slightly outperforms **M1** and **M3**.

7 Conclusion

As we see from the presented overall timing results, the novel methods **M2** and **M3** slightly outperform Method **M1**, since the average number of inner iterations is usually only 1 and always below 2, unlike in case of Method **M1**, where it grows up to 7. The growth of the average PCG iterations per the outermost FGMRES iteration is in case of Methods **M1** and **M3** actually caused by an increase of the inner FGMRES iterations. Thanks to the robustness and the efficiency of the chosen multigrid preconditioner the innermost PCG iterations hardly exceed 2. From the timing results we also observe that the PRESB

Table 4: **M2**: Sum of the iteration counts for H_1 and H_2, varying β, μ and l, $\omega = 1$

β	l	$1/\mu$				
		10^{-8}	10^{-4}	10^{0}	10^{4}	10^{8}
10^{-10}	0	2(1)	4(1)	13(1)	19(1)	5(1)
	1	2(1)	4(1)	19(1)	20(1)	14(1)
	2	2(1)	4(1)	21(1)	21(2)	16(2)
10^{-8}	0	2(1)	4(1)	21(1)	11(1)	5(1)
	1	2(1)	5(1)	22(1)	14(2)	14(2)
	2	2(1)	6(1)	23(1)	16(2)	16(2)
10^{-6}	0	3(1)	6(1)	21(1)	8(1)	7(1)
	1	3(1)	6(1)	22(1)	13(2)	14(2)
	2	3(1)	9(1)	23(1)	16(2)	15(2)
10^{-4}	0	3(1)	8(1)	23(1)	7(1)	7(1)
	1	3(1)	10(1)	23(1)	14(2)	13(2)
	2	3(1)	15(1)	26(2)	16(2)	15(2)
10^{-2}	0	3(1)	12(1)	19(1)	5(1)	6(1)
	1	3(1)	18(1)	20(1)	13(2)	12(2)
	2	3(1)	20(1)	21(2)	15(2)	15(2)

method of Section 2.1 outperforms the real-valued preconditioner of Section 2.2. The memory consumption is nearly equal for all combinations of solution methods.

References

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

[2] Z.-Z. O. Axelsson, G.-F. Zhang, Efficient iterative solvers for a complex valued two-by-two block linear system with application to parabolic optimal control problems. *Appl. Numer. Math.* 152 (2020), 422-445.

[3] Z.-Z. Liang, O. Axelsson, M. Neytcheva, A robust structured preconditioner for time-harmonic parabolic optimal control problems. *Numer. Algorithms* 79 (2018), 575–596.

[4] M. Kolmbauer, The Multiharmonic Finite Element and Boundary Element Method for Simulation and Control of Eddy Current Problems. Ph.D. thesis, Johannes Kepler Universität, Linz, Austria, 2012.

Table 5: **M3**: Iteration counts for varying β, ω and l, $\mu = 1$

β	l	ω				
		10^{-2}	10^{0}	10^{2}	10^{4}	10^{6}
10^{-10}	0	5(1)	5(1)	5(1)	6(1)	4(1)
	1	8(1)	8(1)	8(1)	8(1)	6(1)
	2	9(1)	9(1)	9(1)	10(1)	8(1)
	3	10(1)	10(1)	10(1)	10(1)	11(1)
10^{-8}	0	9(1)	9(1)	9(1)	10(1)	5(1)
	1	10(1)	10(1)	10(1)	11(1)	6(1)
	2	10(1)	10(1)	10(1)	11(1)	9(1)
	3	11(1)	11(1)	11(1)	11(1)	12(1)
10^{-6}	0	10(1)	10(1)	10(1)	12(1)	5(1)
	1	11(1)	11(1)	11(1)	12(1)	7(1)
	2	11(1)	11(1)	11(1)	12(1)	10(1)
	3	12(2)	12(2)	12(2)	13(1)	14(1)
10^{-4}	0	12(1)	12(1)	12(1)	13(1)	6(1)
	1	12(1)	12(1)	12(1)	13(1)	8(1)
	2	12(2)	12(2)	13(2)	14(1)	12(1)
	3	12(2)	12(2)	12(2)	14(1)	15(1)
10^{-2}	0	10(1)	10(1)	13(1)	14(1)	6(1)
	1	10(1)	10(1)	13(1)	15(1)	8(1)
	2	10(2)	10(2)	14(2)	15(1)	13(1)
	3	11(2)	11(2)	13(2)	15(2)	16(2)

[5] M. Kolmbauer and U. Langer, A robust preconditioned MINRES solver for distributed time-periodic eddy current optimal control problems. *SIAM J. Sci. Comput.* 34 (2012), B785–B809.

[6] F. Bachinger, U. Langer, and J. Schöberl, Eflcient solvers for nonlinear time-periodic eddy current problems. *Comput. Vis. Sci.* 9 (2006), 197–207.

[7] O. Axelsson, D. Lukáš, Preconditioning methods for eddy-current optimally controlled time-harmonic electromagnetic problems. *J. Numer. Math.* 27 (2019), 1–21.

[8] I. Yousept, Optimal control of quasilinear H(curl)-elliptic partial differential equations in magnetostatic field problems. *SIAM J. Control Optim.* 51 (2013), 3624–3651.

[9] I. Yousept, Optimal control of non-smooth hyperbolic evolution Maxwell equations in type-II superconductivity. *SIAM J. Control Optim.* 55 (2017), 2305–2332.

Table 6: **M3**: Iteration counts for varying β, μ and l, $\omega = 1$

β	l	$1/\mu$				
		10^{-8}	10^{-4}	10^{0}	10^{4}	10^{8}
10^{-10}	0	1(1)	1(1)	5(1)	10(1)	3(1)
	1	1(1)	1(1)	8(1)	10(1)	5(2)
	2	1(1)	1(1)	9(1)	10(2)	5(2)
10^{-8}	0	1(1)	1(1)	9(1)	6(1)	3(1)
	1	1(1)	1(1)	10(1)	7(2)	5(2)
	2	1(1)	2(1)	10(1)	7(2)	6(2)
10^{-6}	0	1(1)	2(1)	10(1)	5(1)	3(1)
	1	1(1)	2(1)	11(1)	5(2)	5(2)
	2	1(1)	3(1)	11(1)	6(2)	5(2)
10^{-4}	0	1(1)	3(1)	12(1)	4(1)	3(1)
	1	1(1)	4(1)	12(1)	5(2)	5(2)
	2	1(1)	6(1)	12(2)	6(2)	5(2)
10^{-2}	0	2(1)	6(1)	10(1)	3(1)	3(1)
	1	2(1)	8(1)	10(1)	5(2)	5(2)
	2	2(1)	10(1)	10(2)	5(2)	5(2)

[10] O. Axelsson, M. Neytcheva, B. Ahmad, A comparison of iterative methods to solve complex valued linear algebraic systems. *Numer. Algorithms* 66 (2014), 811–841.

[11] O. Axelsson, M. Neytcheva, A. Ström, An efficient preconditioning method for state box-constrained optimal control problems. *J. Numer. Math.* 26 (2018), 185–207.

[12] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, 1994.

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

[14] O. Axelsson, W. Layton, A two–level method for the discretization of nonlinear boundary value problems. *SIAM J. Numer. Anal.* 33 (1996), 2359–2374.

[15] O. Axelsson and I. Gustafsson, Preconditioning and two-level multigrid methods of arbitrary degree of approximation. *Math. Comput.* 40 (1983), 219–242.

[16] J. C. Nédélec, Mixed finite elements in R3. *Numer. Math.* 35 (1980), 315–341.

Table 7: Timing and memory usage

Level of refinement	Time range (s)			Memory usage
	Assembly	Solution		
		C-to-R	**Real**	
M1				
0	21	2-14	2-38	70 MB
1	39	98-483	149-1997	420 MB
2	274	1560-6116	1545-18494	3.2 GB
3	1998	8612-41661	—	25.3 GB
M2				
0	16	1-5	1-4	68 MB
1	51	67-214	30-182	403 MB
2	245	697-2593	395-1640	3.0 GB
3	2004	7963-15931	4619-9359	24.4 GB
M3				
0	20	1-6	n.a.	68 MB
1	52	97-253	n.a.	398 MB
2	328	1225-2956	n.a.	3.0 GB
3	2005	8426-16498	n.a.	24.1 GB

[17] T.A. Davis, J.R. Gilbert, S.I. Larimore, E.G. Ng, Algorithm 836: COLAMD, a column approximate minimum degree ordering algorithm. *ACM Trans. Math. Software* 30 (2004), 377–380.

[18] D. N. Arnold, R. S. Falk, R. Winther, Multigrid in H(div) and H(curl). *Numer. Math.* 85 (2000), 197–217.

[19] J. Schöberl, Netgen/NGSolve - a high performance multiphysics finite element software, `https://ngsolve.org`

[20] Y. Saad, A Flexible Inner-Outer Preconditioned GMRES Algorithm, *SIAM Journal on Scientific Computing*, 14 (1993), 461–469.

[21] S. C. Eisenstat, H. C. Elman, M. H. Schultz, Variational iterative methods for nonsymmetric systems of linear equations, *SIAM Journal on Numerical Analysis*, 20 (1983), 345-–357.

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