SMAI-JCM

SMAI Journal of Computational Mathematics

Accelerating non-local exchange in generalized optimized Schwarz methods

Xavier Claeys & Roxane Delville Atchekzai

Volume 11 (2025), p. 517-532.

https://doi.org/10.5802/smai-jcm.133

The SMAI Journal of Computational Mathematics is a member of the Centre Mersenne for Open Scientific Publishing

http://www.centre-mersenne.org/

Submissions at https://smai-jcm.centre-mersenne.org/ojs/submission

e-ISSN: 2426-8399

SMAI Journal of Computational Mathematics
Vol. 11, 517-532 (2025)

Accelerating non-local exchange in generalized optimized Schwarz methods

Xavier Claeys [1]
Roxane Delville Atchekzai [2]

[1] POems, CNRS, Inria, ENSTA, Institut Polytechnique de Paris, 91120 Palaiseau, France
E-mail address: xavier.claeys@ensta-paris.fr
[2] CEA, CESTA, 33114 Le Barp, France
E-mail address: roxane.delvilleatchekzai@cea.fr.

Abstract. The generalized optimised Schwarz method proposed in [Claeys & Parolin, 2022] is a variant of the Després algorithm for solving harmonic wave problems where transmission condition are enforced by means of a non-local exchange operator. We introduce and analyse an acceleration technique that significantly reduces the cost of applying this exchange operator without deteriorating the precision and convergence speed of the overall domain decomposition algorithm.

2020 Mathematics Subject Classification. 65N55, 65F10, 65N22.

1. Introduction

The present article is concerned with the efficient numerical solution of time harmonic scalar wave equation by means of a non-overlapping domain decomposition method. The Després algorithm [7] also dubbed Optimized Schwarz Method (OSM) is among the most popular algorithms for this type of computation [9]. In a recent series of contributions [2, 3, 4, 5], we introduced variants of OSM able to treat the presence of cross points in a systematic manner while maintaining geometric convergence of the overall DDM algorithms. This approach was proved a generalization of classical OSM in the sense that it coincides with it under appropriate circumstances. A full convergence framework was also provided for this new approach, including a precise quantification of convergence rates.

In classical OSM, wave equations are solved locally in each subdomain. The local solves are then coupled by means of a swapping operator Π_{loc} that exchanges ingoing/outgoing traces through each interface. In the generalized variant of OSM introduced in [5], a key innovation lies in a more sophisticated exchange operator Π that replaces the swapping operator. While Π_{loc} is local by nature, the new exchange operator Π is non-local because it a priori couples distant non-neighbouring subdomains (although $\Pi = \Pi_{\mathrm{loc}}$ in well identified circumstances).

Compared to the standard OSM, the generalized OSM leads to rapidly converging DDM algorithms, but requires dealing with a potentially non-local exchange operator instead of the initial swapping operator, which represents an extra non-negligible computational cost. To be more precise, while the operation $\boldsymbol{x} \to \Pi_{\mathrm{loc}}(\boldsymbol{x})$ is trivial and simply consists in a permutation of unknowns, the operation $\boldsymbol{x} \to \Pi(\boldsymbol{x})$ requires the solution to a global problem that has nevertheless the favorable property of being hermitian positive definite. The goal of the present article is to exhibit one strategy that allows to perform the exchange operation $\boldsymbol{x} \to \Pi(\boldsymbol{x})$ approximately but much faster. We will prove in addition that this approximation does not induce any error in the overall DDM algorithm.

The exchange operation $\boldsymbol{x} \to \Pi(\boldsymbol{x})$ requires solving a self-adjoint positive definite (SPD) linear system, for which we rely on a preconditioned conjugate gradient (PCG) solver. At each iteration n

https://doi.org/10.5802/smai-jcm.133

of the DDM algorithm, such a linear system $\mathrm{L}(\boldsymbol{x}^{(n)}) = \boldsymbol{b}^{(n)}$ has to be solved. Two remarks can be made that allow to substantially accelerate these linear solves. First of all, the linear operator L is independent of n. Besides, the right hand sides $\boldsymbol{b}^{(n)}$ vary from one step n to another, but they form a converging sequence because of the convergence of the overall DDM algorithm. In this context, our acceleration strategy consists in a simple recycling strategy combined with a brutal truncation of PCG (only a few iterations are needed). Here, by recycling, we mean a procedure that takes account of previous iterates $\boldsymbol{x}^{(n-1)}, \boldsymbol{x}^{(n-2)}, \dots$ to speed up the solution of the linear system $\mathrm{L}(\boldsymbol{x}^{(n)}) = \boldsymbol{b}^{(n)}$. The recycling procedure we consider is called "warm restarting" in [13, §3.1]: it is a simple reuse of the solution $\boldsymbol{x}^{(n-1)}$ as an initial guess at the next iterate n. Although more complex iterative recycling strategies can be found in the literature (see [13] for an overview), in the present contribution, we stick to the warm restarting because it is elementary enough to allow a convergence analysis. We shall examine how to take advantage of more sophisticated recycling procedures in a future work.

After a description of our method, we shall give numerical evidence of the performance of this approach. In the second part of this contribution, we give a theoretical justification of the efficiency through derivation of an explicit convergence estimate.

2. Scattering problem under study

We start by describing a typical wave propagation boundary value problem. The aim of the domain decomposition method we are discussing here is to solve this problem as efficiently as possible. In the sequel $\Omega \subset \mathbb{R}^d$ will refer to a polygonal/polyhedral bounded domain, and Ω_{e} will refer to the unbounded connected component of $\mathbb{R}^d \setminus \overline{\Omega}$. We wish to solve the boundary value problem

$$\begin{aligned}&\text{Find } u \in \mathrm{H}^1(\Omega) \text{ such that}\\ &\begin{cases} \Delta u + \kappa^2 u = 0 & \text{in } \Omega,\\ \partial_{\boldsymbol{n}} u - i\kappa u = f & \text{on } \partial\Omega_e,\\ \partial_{\boldsymbol{n}} u = 0 & \text{on } \partial\Omega \setminus \partial\Omega_e.\end{cases}\end{aligned} \tag{2.1}$$

where $f \in \mathrm{L}^2(\partial\Omega) := \{v : \Omega \to \mathbb{C},\ \|v\|^2_{\mathrm{L}^2(\partial\Omega)} := \int_{\partial\Omega} |v|^2 \mathrm{d}\sigma < +\infty\}$ is any square integrable function and $\partial_{\boldsymbol{n}} u := \boldsymbol{n} \cdot \nabla u$ with $\boldsymbol{n}$ the vector field normal to the boundary $\partial\Omega$ directed toward the exterior. The wave number is modelled as a real constant $\kappa > 0$. Following widespread notations, we have considered the Sobolev space $\mathrm{H}^1(\Omega) := \{v \in \mathrm{L}^2(\Omega),\ \nabla v \in \mathrm{L}^2(\Omega)^d\}$ equipped with $\|v\|^2_{\mathrm{H}^1(\Omega)} := \|\nabla v\|^2_{\mathrm{L}^2(\Omega)} + \kappa^2 \|v\|^2_{\mathrm{L}^2(\Omega)}$. Problem (2.1) can be put in the variational form: find $u \in \mathrm{H}^1(\Omega)$ such that $a(u, v) = \ell(v)\ \ \forall\, v \in \mathrm{H}^1(\Omega)$ where

$$\begin{aligned} a(u, v) &:= \int_\Omega \nabla u \nabla \overline{v} - \kappa^2 u \overline{v}\, \mathrm{d}\boldsymbol{x} - i\kappa \int_{\partial\Omega_e} u \overline{v}\, \mathrm{d}\sigma\\ \ell(v) &:= \int_{\partial\Omega_e} f \overline{v}\, \mathrm{d}\sigma. \end{aligned} \tag{2.2}$$

Next we consider a regular triangulation $\mathcal{T}_h(\Omega)$ of the computational domain $\overline{\Omega} = \cup_{\tau \in \mathcal{T}_h(\Omega)} \overline{\tau}$ and we denote $\mathrm{V}_h(\Omega) := \{v \in \mathscr{C}^0(\overline{\Omega}) : v|_\tau \in \mathbb{P}_k(\overline{\tau})\ \ \forall\ \tau \in \mathcal{T}_h(\Omega)\} \subset \mathrm{H}^1(\Omega)$ a space of $\mathbb{P}_k$-Lagrange finite element functions constructed on this mesh, where $\mathbb{P}_k(\overline{\tau}) := \{\text{polynomials of order} \leq k \text{ on } \overline{\tau}\}$. The associated discrete variational formulation then writes

$$\begin{aligned}&\text{Find } u_h \in \mathrm{V}_h(\Omega) \text{ such that}\\ &a(u_h, v_h) = \ell(v_h)\quad \forall\, v_h \in \mathrm{V}_h(\Omega).\end{aligned} \tag{2.3}$$

Problem (2.3) shall be assumed to admit a unique solution, which is simply equivalent to assuming that the corresponding matrix (for a given choice of shape functions) is invertible. The domain decomposition strategy that we subsequently discuss aims at computing this solution.

3. Decomposition of the computational domain

In the perspective of domain decomposition, we need to introduce a geometric decompositon of the computational domain. The strategy we wish to consider belongs to the class of substructuring methods and thus requires a non-overlapping partition

$$\begin{aligned} &\overline{\Omega} = \overline{\Omega}_1 \cup \cdots \cup \overline{\Omega}_{\mathrm{J}} \quad \text{with } \; \Omega_j \cap \Omega_k = \emptyset \text{ for } j \neq k, \\ &\Sigma = \Gamma_1 \cup \cdots \cup \Gamma_{\mathrm{J}} \quad \text{where } \; \Gamma_j := \partial\Omega_j. \end{aligned} \tag{3.1}$$

Each $\Omega_j \subset \Omega$ will be a polyhedral set assumed to be exactly resolved by the triangulation i.e. $\overline{\Omega}_j = \cup_{\tau \in \mathcal{T}_h(\Omega_j)} \overline{\tau}$ where $\mathcal{T}_h(\Omega_j) := \{\tau \in \mathcal{T}_h(\Omega) : \tau \subset \Omega_j\}$. Following usual parlance, we shall call Σ the skeleton of the partition.

In practice the geometric decomposition above is obtained by means of a graph partitioner. Such a decomposition a priori involves cross-points i.e. points of adjacency of either three sub-domains or two sub-domains meeting and the exterior boundary. The set of cross-points is also refered to as wire basket in DDM related literature. A major advantage of the DDM strategy we will consider is its ability to handle cross-points properly.

We consider finite element spaces local to each subdomain $\mathrm{V}_h(\Omega_j) := \{v_h|_{\Omega_j} : v_h \in \mathrm{V}_h(\Omega)\}$, as well as finite element spaces on local boundaries $\mathrm{V}_h(\Gamma_j) := \{v_h|_{\Gamma_j} : v_h \in \mathrm{V}_h(\Omega)\}$. We shall also refer to finite element functions defined on the skeleton

$$\mathrm{V}_h(\Sigma) := \{v_h|_{\Sigma} : \; v_h \in \mathrm{V}_h(\Omega)\}. \tag{3.2}$$

We also need to consider volume based finite element functions that are only piecewise continuous, with possible jumps through interfaces. Such a space is naturally identified with a cartesian product. This leads to setting

$$\mathbb{V}_h(\Omega) := \mathrm{V}_h(\Omega_1) \times \cdots \times \mathrm{V}_h(\Omega_{\mathrm{J}}). \tag{3.3}$$

Remark 3.1. We draw the attention of our reader on the fact that $\mathbb{V}_h(\Omega)$ differs from $\mathrm{V}_h(\Omega)$. The elements of $\mathbb{V}_h(\Omega)$ may be understood as functions defined in each subdomain separately, that are continuous within each subdomain, and that may jump across interfaces. On the other hand, elements of $\mathrm{V}_h(\Omega)$ are functions defined and continuous all over the computational domain Ω that admit no jump through interfaces.

We are interested in domain decomposition where behaviour of functions at interfaces play a crucial role, so we also need to introduce a space of traces at local boundaries $\mathbb{V}_h(\Sigma)$ and the corresponding trace map $\mathrm{B} : \mathbb{V}_h(\Omega) \to \mathbb{V}_h(\Sigma)$ defined by

$$\begin{aligned} &\mathbb{V}_h(\Sigma) := \mathrm{V}_h(\Gamma_1) \times \cdots \times \mathrm{V}_h(\Gamma_{\mathrm{J}}) \\ &\mathrm{B}(v) := (v_1|_{\Gamma_1}, \ldots, v_{\mathrm{J}}|_{\Gamma_{\mathrm{J}}}). \end{aligned} \tag{3.4}$$

Finally we need to embbed the space of trace on the skeleton into the space of traces on local boundaries by means of a restriction operator $\mathrm{R} : \mathrm{V}_h(\Sigma) \to \mathbb{V}_h(\Sigma)$ defined subdomain-wise through

$$\mathrm{R}(v) := (v|_{\Gamma_1}, \ldots, v|_{\Gamma_{\mathrm{J}}}). \tag{3.5}$$

The geometric decomposition that we have introduced above induces a decomposition of the sesquilinear form (2.2) and leads to an elementary reformulation of the discrete problem (2.3).

Define $\mathrm{A} : \mathbb{V}_h(\Omega) \to \mathbb{V}_h(\Omega)^*$ and $\boldsymbol{l} \in \mathbb{V}_h(\Omega)^*$ by

$$\begin{aligned}\langle \mathrm{A}(\boldsymbol{u}), \boldsymbol{v}\rangle &:= \sum_{j=1,\dots,\mathrm{J}} \int_{\Omega_j} \nabla u_j \nabla v_j - \kappa^2 u_j v_j \, d\boldsymbol{x} - i\kappa \int_{\partial\Omega_e \cap \partial\Omega_j} u_j v_j \mathrm{d}\sigma \\ \langle \boldsymbol{l}, \boldsymbol{v}\rangle &:= \sum_{j=1,\dots,\mathrm{J}} \int_{\partial\Omega_e \cap \partial\Omega_j} f \, v_j \, \mathrm{d}\boldsymbol{x}\end{aligned} \tag{3.6}$$

for any $\boldsymbol{u} = (u_1, \dots, u_\mathrm{J}), \boldsymbol{v} = (v_1, \dots, v_\mathrm{J}) \in \mathbb{V}_h(\Omega)$. The operator A is block diagonal, with each block associated with a different subdomain. The initial discrete variational formulation can be rewritten as follows: a function $u_h \in \mathrm{V}_h(\Omega)$ solves (2.3) if and only if $\boldsymbol{u} = (u_h|_{\Omega_1}, \dots, u_h|_{\Omega_\mathrm{J}})$ solves

$$\begin{aligned}&\boldsymbol{u} \in \mathbb{X}_h(\Omega) := \{(v|_{\Omega_1}, \dots, v|_{\Omega_\mathrm{J}}), v \in \mathrm{V}_h(\Omega)\} \\ &\text{and } \langle \mathrm{A}(\boldsymbol{u}), \boldsymbol{v}\rangle = \langle \boldsymbol{l}, \boldsymbol{v}\rangle \quad \forall\, \boldsymbol{v} \in \mathbb{X}_h(\Omega).\end{aligned} \tag{3.7}$$

4. Exchange operator

To obtain a domain decomposition method, we need to further transform (3.7). Our final formulation will be posed in the function space $\mathbb{V}_h(\Sigma)$ attached to the skeleton, and we need to introduce a scalar product for this space i.e. an hermitian positive definite operator

$$\begin{aligned}&\mathrm{T} : \mathbb{V}_h(\Sigma) \to \mathbb{V}_h(\Sigma)^* \text{ such that} \\ &\mathrm{T} = \mathrm{T}^* \text{ and } \langle \mathrm{T}(\boldsymbol{v}), \overline{\boldsymbol{v}}\rangle > 0 \ \ \forall\, \boldsymbol{v} \in \mathbb{V}_h(\Sigma) \setminus \{0\}.\end{aligned} \tag{4.1}$$

The operator $\mathrm{T}^{-1} : \mathbb{V}_h(\Sigma)^* \to \mathbb{V}_h(\Sigma)$ induces a scalar product on $\mathbb{V}_h(\Sigma)^*$. This scalar product will be used to quantify convergence of our domain decomposition algorithm. In the subsequent analysis, the space $\mathbb{V}_h(\Sigma)$ (resp. $\mathbb{V}_h(\Sigma)^*$) will be equipped with the norm

$$\begin{aligned}\|\boldsymbol{v}\|_\mathrm{T}^2 &:= \langle \mathrm{T}(\boldsymbol{v}), \overline{\boldsymbol{v}}\rangle \\ \|\boldsymbol{q}\|_{\mathrm{T}^{-1}}^2 &:= \langle \mathrm{T}^{-1}(\boldsymbol{q}), \overline{\boldsymbol{q}}\rangle\end{aligned} \tag{4.2}$$

Based on the scalar products above, we are going to consider so-called exchange operators that are responsible for enforcing transmission conditions through interfaces and hence coupling between subdomains. We define the exchange operator $\Pi : \mathbb{V}_h(\Sigma)^* \to \mathbb{V}_h(\Sigma)^*$ by the identity

$$\Pi := 2\mathrm{TR}(\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^* - \mathrm{Id}. \tag{4.3}$$

Because $\mathrm{TR}(\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^*$ is a T^{-1}-orthogonal projector, it is clear that Π is unitary by construction i.e. $\|\Pi(\boldsymbol{p})\|_{\mathrm{T}^{-1}} = \|\boldsymbol{p}\|_{\mathrm{T}^{-1}}\ \forall\, \boldsymbol{p} \in \mathbb{V}_h(\Sigma)^*$. Besides it is a priori non-local in the sense that it may couple trace functions attached to distant subdomains.

In the domain decomposition method considered here, many choices are possible for the operator T. Any choice fullfilling (4.1) is valid. A discussion about these multiple possibilities is available in [5, §5]. Below we examine two extreme cases (a) and (b).

(a) *Purely local exchange*
Denote $\mathrm{dof}(\Gamma_j)$ (resp. $\mathrm{dof}(\Sigma)$) those geometrical points that can be identified with degrees of freedom of the space $\mathrm{V}_h(\Gamma_j)$ (resp. $\mathrm{V}_h(\Sigma)$). Local to each subdomain boundary, define $\mathrm{T}_j : \mathrm{V}_h(\Gamma_j) \to \mathrm{V}_h(\Gamma_j)^*$ by the expression

$$\langle \mathrm{T}_j(u), v\rangle := \sum_{\boldsymbol{x} \in \mathrm{dof}(\Gamma_j)} u(\boldsymbol{x}) v(\boldsymbol{x})$$

In the standard basis of shape functions of $\mathrm{V}_h(\Gamma_j)$, this operator T_j is represented by the identity matrix. The most simple choice of operator T is then to take $\langle \mathrm{T}(\boldsymbol{u}), \boldsymbol{v}\rangle := \langle \mathrm{T}_1(u_1), v_1\rangle + \dots + \langle \mathrm{T}_\mathrm{J}(u_\mathrm{J}), v_\mathrm{J}\rangle$ for $\boldsymbol{u} = (u_1, \dots, u_\mathrm{J}), \boldsymbol{v} = (v_1, \dots, v_\mathrm{J}) \in \mathbb{V}_h(\Sigma)$. It was established in [3,

§9.1] that, with this particular choice of T, the operator Π becomes *purely local* i.e. it only couples unknowns that are geometrically close to each other and, if we denote $\mathfrak{I}(\boldsymbol{x}) := \{j \in \{1, \ldots, \mathrm{J}\},\ \boldsymbol{x} \in \Gamma_j\}$, we have the explicit formula

$$\langle \Pi \mathrm{T}(\boldsymbol{u}), \boldsymbol{v} \rangle = -\langle \mathrm{T}(\boldsymbol{u}), \boldsymbol{v} \rangle + \sum_{\boldsymbol{x} \in \mathrm{dof}(\Sigma)} \frac{1}{\mathrm{card}\, \mathfrak{I}(\boldsymbol{x})} \Bigg(\sum_{j \in \mathfrak{I}(\boldsymbol{x})} u_j(\boldsymbol{x}) \Bigg) \Bigg(\sum_{k \in \mathfrak{I}(\boldsymbol{x})} v_k(\boldsymbol{x}) \Bigg). \tag{4.4}$$

A close inspection of this formula reveals that, except at cross-points where it computes some kind of average, this operator Π simply swaps unknowns from both sides of each interface (see [3, §9.1]) so that performing $\boldsymbol{u} \mapsto \Pi(\boldsymbol{u})$ is costless and fast. In fact, the exchange operator given by (4.4) *is* the local swapping operator Π_{loc} mentionned in the introduction of the present article.

(b) *Fully non-local exchange*
Another possible choice of operator T discussed in [5, Example 5.5] consists in taking $\langle \mathrm{T}(\boldsymbol{u}), \boldsymbol{v} \rangle := \langle \mathrm{T}_1(u_1), v_1 \rangle + \cdots + \langle \mathrm{T}_\mathrm{J}(u_\mathrm{J}), v_\mathrm{J} \rangle$ for $\boldsymbol{u} = (u_1, \ldots, u_\mathrm{J}), \boldsymbol{v} = (v_1, \ldots, v_\mathrm{J}) \in \mathbb{V}_h(\Sigma)$, where each $\mathrm{T}_j : \mathrm{V}_h(\Gamma_j) \to \mathrm{V}_h(\Gamma_j)^*$ is defined so as to minimize the functional

$$\langle \mathrm{T}_j(v), v \rangle := \min\Big\{ \|\nabla \widetilde{v}\|^2_{\mathrm{L}^2(\Omega_j)} + \kappa^2 \|\widetilde{v}\|^2_{\mathrm{L}^2(\Omega_j)},\ \widetilde{v} \in \mathrm{V}_h(\Omega_j),\ \widetilde{v}|_{\Gamma_j} = v \Big\}$$

With this particular choice of T, the operator Π becomes *fully non-local*, and computing $\boldsymbol{q} \mapsto \Pi(\boldsymbol{q})$ requires to solve a symmetric positive definite problem over the whole computational domain Ω. With this choice, performing the operation $\boldsymbol{u} \mapsto \Pi(\boldsymbol{u})$ appears a priori costly.

In our domain decomposition method, the impedance T plays the role of a tuning parameter. Here, optimization of interface conditions takes place through the choice of T and this is why we dub this approach *Generalized Optimized Schwarz Method.* We have just described two extreme cases with Choice (a) that leads to an easy and fast implementation of Π, while Choice (b) leads to an operator Π that is non-local and a priori costly. Naively, it may be tempting to systematically opt for Choice (a). However we will see in the next section that Choice (a) is in fact not the most favorable from a domain decomposition stand point.

5. Skeleton formulation

Now we describe a formulation equivalent to (3.7) that serves as the master equation of our domain decomposition algorithm. It will be posed on the skeleton of the decomposition. In this skeleton equation, wave problems local to each subdomain are written by means of a so-called scattering operator $\mathrm{S} : \mathbb{V}_h(\Sigma)^* \to \mathbb{V}_h(\Sigma)^*$ defined by

$$\begin{aligned} \mathrm{S} &:= \mathrm{Id} + 2i\mathrm{TB}(\mathrm{A} - i\mathrm{B}^*\mathrm{TB})^{-1}\mathrm{B}^* \\ \boldsymbol{g} &:= 2i\mathrm{TB}(\mathrm{A} - i\mathrm{B}^*\mathrm{TB})^{-1}\boldsymbol{l} \end{aligned} \tag{5.1}$$

with $\boldsymbol{g} \in \mathbb{V}_h(\Sigma)^* = \mathrm{Range(T)}$. Since A and B are both subdomain-wise block-diagonal, when T is subdomain-wise block-diagonal, so is the scattering operator S. This makes such an operator adapted to parallelism. The following Proposition was established in [3, §6].

Proposition 5.1. *The function* $\boldsymbol{u} \in \mathbb{X}_h(\Omega)$ *solves* (3.7) *if and only if there exists* $\boldsymbol{q} \in \mathbb{V}_h(\Sigma)^*$ *satisfying* $\mathrm{B}^*\boldsymbol{q} = (\mathrm{A} - i\mathrm{B}^*\mathrm{TB})\boldsymbol{u} - \boldsymbol{l}$ *and the skeleton formulation*

$$(\mathrm{Id} + \Pi\mathrm{S})\boldsymbol{q} = \boldsymbol{g} \tag{5.2}$$

From the proposition above, we see that if the skeleton formulation (5.2) is solved, then the complete solution $\boldsymbol{u} = (\mathrm{A}-i\mathrm{B}^*\mathrm{T}\mathrm{B})^{-1}(\mathrm{B}^*\boldsymbol{q}+\boldsymbol{l})$ can be reconstructed, and this reconstruction step is fully parallel if the impedance operator T is block-diagonal.

A key feature of the skeleton formulation above is its strong coercivity with respect to the scalar product induced by T^{-1}, in spite of the a priori sign indefiniteness of (3.7). The following result was established in [3, Cor. 6.2].

Proposition 5.2. *The operator* $\mathrm{Id}+\Pi\mathrm{S} : \mathbb{V}_h(\Sigma)^* \to \mathbb{V}_h(\Sigma)^*$ *is a bijection that fulfills a coercivity bound in the scalar product induced by* T^{-1}. *For all* $\boldsymbol{q} \in \mathbb{V}_h(\Sigma)^*$ *we have*

$$\begin{aligned}&\Re e\{\langle(\mathrm{Id}+\Pi\mathrm{S})\boldsymbol{q}, \mathrm{T}^{-1}\overline{\boldsymbol{q}}\rangle\} \geq (\gamma_h^2/2)\|\boldsymbol{q}\|_{\mathrm{T}^{-1}}^2\\ &\textit{where } \gamma_h := \inf_{\boldsymbol{q}\in\mathbb{V}_h(\Sigma)^*\setminus\{0\}} \|(\mathrm{Id}+\Pi\mathrm{S})\boldsymbol{q}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}\|_{\mathrm{T}^{-1}}\end{aligned}$$

This result garantees a good behaviour of classical iterative solvers such as Richardson or GMRes when applied to the skeleton formulation (5.2). The coercivity estimate above leads to convergence bounds for such solvers, see e.g. [1].

In [5, Prop. 10.4], an explicit lower bound was provided for γ_h. This bound involves the inf-sup constant of the bilinear form $a(\cdot,\cdot)$ from (2.3) (which is h-uniformly lower bounded), and also the extremal eigenvalues of the impedance T with respect to the scalar product corresponding to Choice (b) of Section 4 above, see Proposition 10.4 in [5]. Here is what this implies for the two extreme cases of Section 4.

(a) If T is defined according to Choice (a), the constant γ_h deteriorates like $\mathcal{O}(h)$ as $h \to 0$ and the coercivity constant deteriorates like $\mathcal{O}(h^2)$, so the convergence of classical iterative solvers such as Richardson or GMRes follows this deterioration. This was discussed in detail in Example 11.4 and Section 14 of [5, Prop. 10.4]. Because of this phenomenon, in many cases, Choice (a) is simply not a viable option because iterative solvers do not converge in a reasonnable amount of time. For this reason, although it induces an explicit and fast exchange operator, Choice (a) is not satisfactory.

(b) If T is defined according to Choice (b), the constant γ_h is h-uniformly lower bounded and the convergence of classical iterative solvers is robust with respect to h. This was also discussed in Example 11.7 and Section 14 of [5, Prop. 10.4]. This motivates focusing on choices like (b) where applying the exchange operator is costly, but this is compensated by a good behaviour of linear solvers.

In a Krylov solver, the matrix-vector operation $\boldsymbol{v} \mapsto (\mathrm{Id}+\Pi\mathrm{S})\boldsymbol{v}$ is a crucial step that has enormous impact on the computational cost of the overall solution procedure. In a situation where the impedance operator T is block-diagonal, the matrix-vector product $\boldsymbol{v} \mapsto \mathrm{S}\boldsymbol{v}$ is embarrassingly parallel as the scattering operator is itself block-diagonal.

As a consequence, the cost of the operation $\boldsymbol{v} \mapsto \Pi(\boldsymbol{v})$ is critical. This reduces to performing $\boldsymbol{v} \mapsto (\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}\boldsymbol{v}$ which, if performed in a naive way (like with a direct Cholesky solver), might be costly. Indeed solving a linear system attached to $\mathrm{R}^*\mathrm{T}\mathrm{R}$ is required each time a matrix-vector product is required inside the global Krylov solver.

The main goal of the present article is to show how this core step, that is at the center of our DDM strategy, can be optimized. We wish to exhibit how each such linear solve can benefit from the previous solves by means of a simple recycling strategy in such a way that the convergence speed of the overall DDM algorithm is not deteriorated.

For the sake of concreteness, we consider a Richardson solver, see e.g. Example 4.1 in [12], as a model iterative solver for the skeleton formulation (5.2). Starting from a trivial initial guess $\boldsymbol{q}^{(0)} = 0$

and choosing $\alpha \in (0,1)$ as relaxation parameter, Richardson's iteration takes the form

$$\boldsymbol{q}^{(n+1)} = (1-\alpha)\boldsymbol{q}^{(n)} - \alpha\Pi\mathrm{S}\boldsymbol{q}^{(n)} + \alpha\boldsymbol{g}. \tag{5.3}$$

Taking account of the expression of the exchange operator given by (4.3), this iteration can be decomposed as follows

Exact Richardson iteration

$$\begin{aligned} &\boldsymbol{p}^{(n+1)} = (\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}\mathrm{R}^*\mathrm{S}\boldsymbol{q}^{(n)} \\ &\boldsymbol{q}^{(n+1)} = ((1-\alpha)\mathrm{Id} + \alpha\mathrm{S})\boldsymbol{q}^{(n)} - 2\alpha\mathrm{T}\mathrm{R}(\boldsymbol{p}^{(n+1)}) + \alpha\boldsymbol{g} \end{aligned} \tag{5.4}$$

We dub this algorithm "exact" because all steps in this procedure are assumed to be conducted without any error. In particular, it is assumed that no approximation is made when evaluating the action of $(\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}$ in the first line above.

6. Approximation of the exchange operator

In a large distributed memory environment, the linear solve of $\boldsymbol{q} \mapsto (\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}\boldsymbol{q}$ cannot be achieved exactly but should rely instead on an approximate PCG solve that consists in Galerkin projections onto a Krylov space that depends itself on the right hand side, see e.g. [10, §2.3].

Consider a linear map $\mathrm{P} : \mathrm{V}_h(\Sigma)^* \to \mathrm{V}_h(\Sigma)$ that, in the subsequent analysis, shall play the role of a preconditioner. For a given integer $k \geq 1$ and an initial guess $\boldsymbol{x}_0 \in \mathrm{V}_h(\Sigma)$, define the k-th order Krylov space

$$\mathscr{K}_k(\boldsymbol{x}_0, \boldsymbol{b}) := \mathrm{vect}\{\boldsymbol{r}_0, (\mathrm{P}\mathrm{R}^*\mathrm{T}\mathrm{R})\boldsymbol{r}_0, \dots, (\mathrm{P}\mathrm{R}^*\mathrm{T}\mathrm{R})^{k-1}\boldsymbol{r}_0\} \qquad \text{where } \boldsymbol{r}_0 := \mathrm{P}\boldsymbol{b} - (\mathrm{P}\mathrm{R}^*\mathrm{T}\mathrm{R})\boldsymbol{x}_0. \tag{6.1}$$

Define $\mathrm{PCG}_k : \mathrm{V}_h(\Sigma) \times \mathrm{V}_h(\Sigma)^* \to \mathrm{V}_h(\Sigma)$ as the map that takes a right-hand side $\boldsymbol{b} \in \mathrm{V}_h(\Sigma)^*$ and an initial guess $\boldsymbol{x}_0 \in \mathrm{V}_h(\Sigma)$, and returns the result of k steps of a PCG algorithm preconditionned with P, see [12, §9.2]. Following the interpretation of Krylov methods in terms of projections, see [12, Chap. 5 & 6] and [10, Chap. 2], it is characterised as the unique solution to the following minimization problem

$$\begin{aligned} &\mathrm{PCG}_k(\boldsymbol{x}_0, \boldsymbol{b}) \in \boldsymbol{x}_0 + \mathscr{K}_k(\boldsymbol{x}_0, \boldsymbol{b}) \quad \text{and} \\ &\|(\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}\boldsymbol{b} - \mathrm{PCG}_k(\boldsymbol{x}_0, \boldsymbol{b})\|_{\mathrm{R}^*\mathrm{T}\mathrm{R}} = \min_{\eta \in \boldsymbol{x}_0 + \mathscr{K}_k(\boldsymbol{x}_0, \boldsymbol{b})} \|(\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}\boldsymbol{b} - \eta\|_{\mathrm{R}^*\mathrm{T}\mathrm{R}}. \end{aligned} \tag{6.2}$$

Let us see how the preconditioned conjugate gradient map PCG_k can be inserted into Richardson's iteration (5.4). A naive approach would systematically take $\boldsymbol{x}_0 = 0$ as initial guess which leads to considering the relation $\boldsymbol{p}^{(n+1)} = \mathrm{PCG}_k(0, \mathrm{R}^*\mathrm{S}\boldsymbol{q}^{(n)})$ for the first equation in (5.4). However, because the overall DDM algorithm is supposed to converge, the right-hand sides $\mathrm{R}^*\mathrm{S}\boldsymbol{q}^{(n)}$ should themselve form a converging sequence. As a consequence $\mathrm{R}^*\mathrm{S}\boldsymbol{q}^{(n-1)}$ should remain close to $\mathrm{R}^*\mathrm{S}\boldsymbol{q}^{(n)}$, so that taking $\boldsymbol{x}_0 = \boldsymbol{p}^{(n)}$ appears as a natural recycling strategy. The modified DDM strategy then takes the form

Approximate Richardson iteration

$$\begin{aligned} &\widetilde{\boldsymbol{p}}^{(n+1)} = \mathrm{PCG}_k(\widetilde{\boldsymbol{p}}^{(n)}, \mathrm{R}^*\mathrm{S}\widetilde{\boldsymbol{q}}^{(n)}) \\ &\widetilde{\boldsymbol{q}}^{(n+1)} = ((1-\alpha)\mathrm{Id} + \alpha\mathrm{S})\widetilde{\boldsymbol{q}}^{(n)} - 2\alpha\mathrm{T}\mathrm{R}(\widetilde{\boldsymbol{p}}^{(n+1)}) + \alpha\boldsymbol{g} \end{aligned} \tag{6.3}$$

The parameter k that represents the dimension of the Krylov space is assumed fixed and independent of n. Because the map PCG_k is not linear, we underline that the iterative procedure above is itself non-linear.

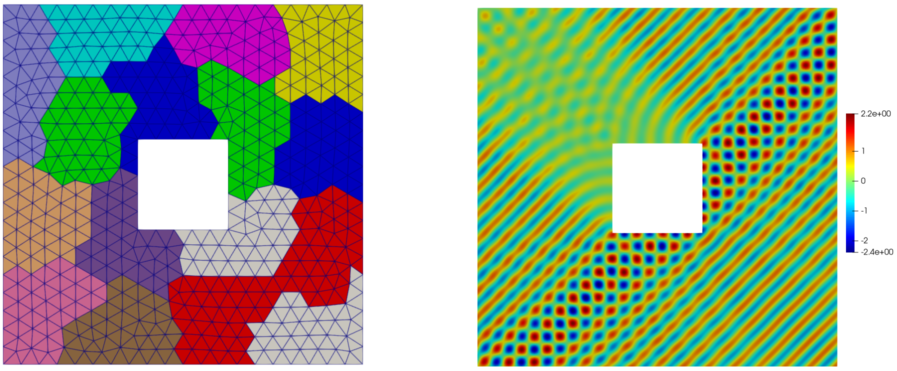

Figure 7.1. Left: Computational domain. Right: Real part of the reference solution.

7. Numerical experiment

We now present a numerical experiment illustrating the strategy described above. We will consider Problem (2.1) set in $\mathbb{R}^d = \mathbb{R}^2$ in the computational domain $\Omega = (-1,+1)^2\setminus[-0.25,+0.25]^2$ represented in Figure 7.1. We take $\lambda = 1/10$ hence a wavenumber $\kappa = 2\pi/\lambda \simeq 62.83$, and the source function $f(\boldsymbol{x}) = \exp(\imath\kappa\,\boldsymbol{d}\cdot\boldsymbol{x})$ with $\boldsymbol{d} = (-1/\sqrt{2},+1/\sqrt{2},0)$. The problem is discretized with $\mathbb{P}_1$-Lagrange finite elements based on a mesh generated with GMSH and partitioned with METIS in 16 subdomains. The computations were run sequentially with the C++ library DDMTOOL on a laptop with Intel® Core™ i7-1185G7 processor with 62.5 Gb of RAM.

All the numerical experiments of the present section have been conducted with the same fixed mesh that contains 175794 nodes (which is also the dimension of $\mathrm{V}_h(\Omega)$) and 349588 triangles. The maximum mesh element size equals $h = 0.005$ so this discretization represents approximately $\lambda/h = 20$ points per wavelength. We compute a reference solution of (5.2) by means of a direct solver using UMPFPACK. This reference solution will be denoted $\boldsymbol{q}^{(\infty)}$ subsequently. Its real part is plotted in the right hand side of Figure 7.1. We construct the impedance operator $\mathrm{T} = \mathrm{diag}(\mathrm{T}_1,\dots,\mathrm{T}_\mathrm{J}) : \mathbb{V}_h(\Sigma) \to \mathbb{V}_h(\Sigma)^*$, with each $\mathrm{T}_j : \mathrm{V}_h(\Gamma_j) \to \mathrm{V}_h(\Gamma_j)^*$ defined locally, following the strategy advocated in [11, Chap. 8], [6], [4, §4.2]. In the present case, this boils down to selecting a subset of each subdomain $\widetilde{\Omega}_j \subset \Omega_j$ consisting in 5 layers of elements neighbouring $\Gamma_j = \partial\Omega_j$ (in particular $\Gamma_j \subset \partial\widetilde{\Omega}_j$) and defining each $\mathrm{T}_j : \mathrm{V}_h(\Gamma_j) \to \mathrm{V}_h(\Gamma_j)^*$ as the unique hermitian positive definite linear map satisfying the minimization property

$$\langle \mathrm{T}_j(v), v\rangle := \min\Big\{ \|\widetilde{v}\|^2_{\mathrm{L}^2(\tilde{\Omega}_j)} + \kappa^2\|\widetilde{v}\|^2_{\mathrm{L}^2(\tilde{\Omega}_j)} + \kappa\|\widetilde{v}\|^2_{\mathrm{L}^2(\partial\tilde{\Omega}_j\setminus\Gamma_j)},\ \widetilde{v}\in\mathrm{V}_h(\widetilde{\Omega}_j),\ \widetilde{v}|_{\Gamma_j} = v\Big\}.$$

The actual evaluation of this impedance operator rests on a (sparse) Cholesky factorization performed by means of UMFPACK locally in each $\widetilde{\Omega}_j$. As for the preconditioner P for the linear solve associated to the operation $\boldsymbol{b} \mapsto (\mathrm{R}^*\mathrm{T}\mathrm{R})^{-1}\boldsymbol{b}$, we choose the single level Neumann–Neumann preconditioner, see [8, §7.8.1] or [14, §6.2].

In a first experiment we run a variant of Algorithm (6.3) with $\alpha = 1/2$, where the initial guess of PCG is chosen trivial i.e. we set $\widetilde{\boldsymbol{p}}^{(n+1)} = \mathrm{PCG}_k(0,\boldsymbol{b})$ where $\boldsymbol{b} = \mathrm{R}^*\mathrm{S}\,\widetilde{\boldsymbol{q}}^{(n)}$, and the PCG algorithm is executed until the relative residual error $\|\boldsymbol{b} - (\mathrm{R}^*\mathrm{T}\mathrm{R})\,\mathrm{PCG}_k(0,\boldsymbol{b})\|_\mathrm{P}/\|\boldsymbol{b}\|_\mathrm{P} \leq 1e-20$ is reached. In Figure 7.2, we plot the norm of the error $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}}$ of Algorithm (6.3) versus the iteration number n. On the left picture of Figure 7.2, we take as many iterations of PCG as needed. For this plot, as a matter of fact, 14 iterations of PCG take place at each iteration of the global

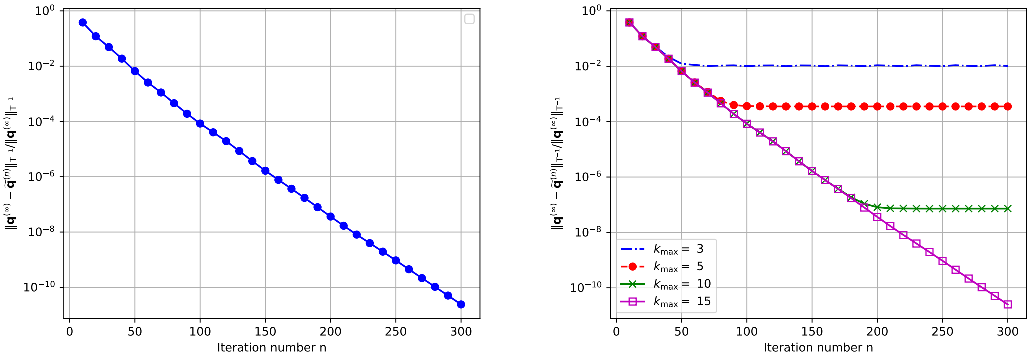

Figure 7.2. Relative error $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}}$ versus iteration number n in Richardson's Algorithm. No initial guess of PCG i.e. no recycling strategy. Left: no limitation on PCG iteration count i.e. $k_{\mathrm{MAX}} = \infty$. Right: imposing in addition $k \leq k_{\mathrm{MAX}}$.

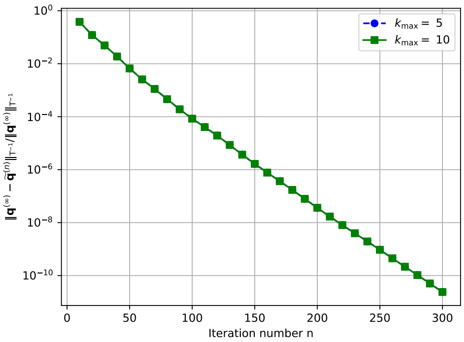

Figure 7.3. Relative error $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}}$ versus iteration number n in Richardson's Algorithm with a recycled initial guess for PCG and $k \leq k_{\mathrm{MAX}} = 5, 10$.

Richardson algorithm. On the right hand side of Figure 7.2, we plot the same graph except that this time the number of iterations of PCG is limited $k \leq k_{\mathrm{MAX}}$ for several values of k_{MAX}. We see that when the number of PCG iterations is limited, the error $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}}$ of the global Richardson algorithm decays normally until it reaches a certain critical value where it stalls. This plateau phenomenon appears here for $k_{\mathrm{MAX}} < 14$.

Next we launch the same computation, again limiting the number of PCG iterations $k \leq k_{\mathrm{MAX}}$. This time though, we choose the initial guess from the previous iterate as described before i.e. $\widetilde{\boldsymbol{p}}^{(n+1)} = \mathrm{PCG}_k(\boldsymbol{x}_0, \boldsymbol{b})$ with $\boldsymbol{b} = \mathrm{R}^*\mathrm{S}\,\widetilde{\boldsymbol{q}}^{(n)}$ and $\boldsymbol{x}_0 = \widetilde{\boldsymbol{p}}^{(n)}$. We do this for the two values $k_{\mathrm{MAX}} = 5$ and $k_{\mathrm{MAX}} = 10$ and plot the corresponding error $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}}$ versus n. This time the error keeps on decaying without reaching any plateau. The plot of Figure 7.3 looks identical to the one in the left hand side of Figure 7.2. This suggests that, when combining a truncation of PCG with the simple recycling strategy described above, the error decay of the global Richardson algorithm does not experience any deterioration. Keeping this strategy consisting in both recycling and truncating PCG, in the next table, we give the number of iterations n required to reach a relative tolerance $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}} < 1e - 10$.

Table 7.1.

k_{MAX}	20	15	10	5	3	2	1
#iter	281	281	281	281	282	286	616

This clearly indicates that, when using a recycling strategy, only a few PCG iterations are needed to maintain the same convergence speed for the overall Richardson algorithm. This represents a clear computational gain since the cost of each iteration of (6.3) depends directly on k_{MAX}.

The previous result shows that, although the operator Π is non-local, in practice its action may be evaluated with a cost corresponding to a small number of matrix-vector products from the operator T. In the next section, we provide theoretical analysis supporting this conclusion.

8. **Convergence analysis**

We exhibited an efficient heuristic to approximate the action of the non-local exchange operator i.e. the first line in (5.4). It consists in combining a brutal truncation of PCG with a basic recycling scheme. We provided numerical evidence supporting the relevance of this strategy.

We seek now to obtain a theoretical explanation for the performance of this approach. Instead of trying to systematically derive the sharpest estimates, at certain points of our analysis we will take upper bounds that are larger than strictly required which, hopefully, will help simplify the calculus. To begin with, we re-arrange the approximate Richardson iteration (6.3),

$$\begin{aligned}
&\widetilde{\boldsymbol{p}}^{(n+1)} - \widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} = \mathrm{PCG}_k(\widetilde{\boldsymbol{p}}^{(n)}, \mathrm{R}^*\mathrm{S}\widetilde{\boldsymbol{q}}^{(n)}) - (\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^*\mathrm{S}\widetilde{\boldsymbol{q}}^{(n)}\\
&\widetilde{\boldsymbol{q}}^{(n+1)} = ((1-\alpha)\mathrm{Id} - \alpha\Pi\mathrm{S})\widetilde{\boldsymbol{q}}^{(n)} - 2\alpha\mathrm{TR}(\widetilde{\boldsymbol{p}}^{(n+1)} - \widetilde{\boldsymbol{p}}_{\infty}^{(n+1)}) + \alpha\boldsymbol{g}\\
&\text{where}\quad \widetilde{\boldsymbol{p}}_{\infty}^{(n)} := (\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^*\mathrm{S}\widetilde{\boldsymbol{q}}^{(n-1)}.
\end{aligned} \tag{8.1}$$

Focusing on the first line of (8.1), we try to estimate the decay of the left hand side, making use of the classical convergence estimate for the conjugate gradient, see e.g. Corollary 5.6.7 in [10] that, in our notations, yields the following inequality

$$\begin{aligned}
&\|(\mathrm{R}^*\mathrm{TR})^{-1}\boldsymbol{b} - \mathrm{PCG}_k(\boldsymbol{x}_0,\boldsymbol{b})\|_{\mathrm{R}^*\mathrm{TR}} \leq \epsilon_k \|(\mathrm{R}^*\mathrm{TR})^{-1}\boldsymbol{b} - \boldsymbol{x}_0\|_{\mathrm{R}^*\mathrm{TR}}\\
&\text{with}\ \ \epsilon_k = 2\left(\frac{\sqrt{\mathrm{cond}(\mathrm{PR}^*\mathrm{TR})}-1}{\sqrt{\mathrm{cond}(\mathrm{PR}^*\mathrm{TR})}+1}\right)^k
\end{aligned} \tag{8.2}$$

and cond(L) refers to the spectral condition number i.e. $\mathrm{cond}(\mathrm{L}) = \sup\mathfrak{S}(\mathrm{L})/\inf\mathfrak{S}(\mathrm{L})$ where $\mathfrak{S}(\mathrm{L})$ is the spectrum of a linear map $\mathrm{L} : \mathrm{V}_h(\Sigma) \to \mathrm{V}_h(\Sigma)$. For the moment, we assume that k is chosen sufficiently large hence ϵ_k as small as required. We shall come back and discuss later on the choice of the parameter k. Injecting Estimate (8.2) into (8.1) leads to following inequality

$$\begin{aligned}
\|\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} - \widetilde{\boldsymbol{p}}^{(n+1)}\|_{\mathrm{R}^*\mathrm{TR}} &\leq \epsilon_k\|\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} - \widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}\\
&\leq \epsilon_k\|\widetilde{\boldsymbol{p}}_{\infty}^{(n)} - \widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}} + \epsilon_k\|\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} - \widetilde{\boldsymbol{p}}_{\infty}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}.
\end{aligned}$$

By the very definition of the auxiliary variable in (8.1), we have $\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} - \widetilde{\boldsymbol{p}}_{\infty}^{(n)} = (\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^*\mathrm{S}(\boldsymbol{w})$ where $\boldsymbol{w} = \widetilde{\boldsymbol{q}}^{(n)} - \widetilde{\boldsymbol{q}}^{(n-1)}$. On the other hand, since $\mathrm{TR}(\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^*$ is a T^{-1}-orthogonal projection, and S is a contraction with respect to $\|\ \|_{\mathrm{T}^{-1}}$ according to Lemma 5.2 in [3], we have $\|\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} - \widetilde{\boldsymbol{p}}_{\infty}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}} = \|\mathrm{TR}(\mathrm{R}^*\mathrm{TR})^{-1}\mathrm{R}^*\mathrm{S}(\boldsymbol{w})\|_{\mathrm{T}^{-1}} \leq \|\boldsymbol{w}\|_{\mathrm{T}^{-1}} = \|\widetilde{\boldsymbol{q}}^{(n)} - \widetilde{\boldsymbol{q}}^{(n-1)}\|_{\mathrm{T}^{-1}}$. From this we obtain

$$\|\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)} - \widetilde{\boldsymbol{p}}^{(n+1)}\|_{\mathrm{R}^*\mathrm{TR}} \leq \epsilon_k\|\widetilde{\boldsymbol{p}}_{\infty}^{(n)} - \widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}} + \epsilon_k\|\widetilde{\boldsymbol{q}}^{(n)} - \widetilde{\boldsymbol{q}}^{(n-1)}\|_{\mathrm{T}^{-1}}. \tag{8.3}$$

Coming back to the approximate Richardson iteration (8.1), we now focus on the second line. We take the difference of two successive iterates, which yields

$$\widetilde{\boldsymbol{q}}^{(n+1)}-\widetilde{\boldsymbol{q}}^{(n)}=((1-\alpha)\mathrm{Id}-\alpha\Pi\mathrm{S})(\widetilde{\boldsymbol{q}}^{(n)}-\widetilde{\boldsymbol{q}}^{(n-1)})-2\alpha\mathrm{TR}(\widetilde{\boldsymbol{p}}^{(n+1)}-\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)})+2\alpha\mathrm{TR}(\widetilde{\boldsymbol{p}}^{(n)}-\widetilde{\boldsymbol{p}}_{\infty}^{(n)}) \quad (8.4)$$

Next we bound the norm of the left-hand side above, taking account of Inequality (8.3), and introducing the continuity modulus of $(1-\alpha)\mathrm{Id}-\alpha\Pi\mathrm{S}$ with respect to the norm induced by T^{-1}. This yields

$$\begin{aligned}&\|\widetilde{\boldsymbol{q}}^{(n+1)}-\widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}\\&\leq\rho\|\widetilde{\boldsymbol{q}}^{(n)}-\widetilde{\boldsymbol{q}}^{(n-1)}\|_{\mathrm{T}^{-1}}+2\alpha\|\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)}-\widetilde{\boldsymbol{p}}^{(n+1)}\|_{\mathrm{R}^*\mathrm{TR}}+2\alpha\|\widetilde{\boldsymbol{p}}_{\infty}^{(n)}-\widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}\\&\leq(\rho+2\alpha\epsilon_k)\|\widetilde{\boldsymbol{q}}^{(n)}-\widetilde{\boldsymbol{q}}^{(n-1)}\|_{\mathrm{T}^{-1}}+2\alpha(1+\epsilon_k)\|\widetilde{\boldsymbol{p}}_{\infty}^{(n)}-\widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}\\&\text{where}\quad\rho:=\sup_{\boldsymbol{q}\in\mathbb{V}_h(\Sigma)^*\setminus\{0\}}\frac{\|((1-\alpha)\mathrm{Id}-\alpha\Pi\mathrm{S})\boldsymbol{q}\|_{\mathrm{T}^{-1}}}{\|\boldsymbol{q}\|_{\mathrm{T}^{-1}}}.\end{aligned}\quad(8.5)$$

We recall that, according to [5, Thm. 9.2], the operator $(1-\alpha)\mathrm{Id}-\alpha\Pi\mathrm{S}$ is a strict contraction with the following estimate for its spectral radius

$$\rho\leq1-\alpha(1-\alpha)\gamma_h^2. \quad (8.6)$$

Now we can gather (8.3) and (8.5) to form a system of 2×2 recursive inequalities. We slightly rescale the inequalities and apply the majorizations $\epsilon_k\leq2\epsilon_k$ and $\alpha\leq1$ to make the analysis more comfortable. Set

$$\mathfrak{R}:=\left[\begin{array}{cc}\rho+2\epsilon_k & 2\sqrt{\epsilon_k(1+\epsilon_k)}\\ 2\sqrt{\epsilon_k(1+\epsilon_k)} & 2\epsilon_k\end{array}\right],\quad\mathfrak{e}^{(n)}:=\left[\begin{array}{l}\|\widetilde{\boldsymbol{q}}^{(n)}-\widetilde{\boldsymbol{q}}^{(n-1)}\|_{\mathrm{T}^{-1}}\\ \|\widetilde{\boldsymbol{p}}_{\infty}^{(n)}-\widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}\sqrt{1+1/\epsilon_k}\end{array}\right]. \quad (8.7)$$

Given two vectors of positive coefficients $u,v\in\mathbb{R}_+^2$ with $u=(u_1,u_2)$ and $v=(v_1,v_2)$, we shall write $u\leq v\iff u_j\leq v_j$ for $j=1,2$. With this notation we obtain $\mathfrak{e}^{(n+1)}\leq\mathfrak{R}\cdot\mathfrak{e}^{(n)}$. Iterating over n, since all coefficients are positive, we obtain $\mathfrak{e}^{(n)}\leq\mathfrak{R}^n\cdot\mathfrak{e}^{(0)}$ with $\mathfrak{e}^{(0)}=(\|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}},0)^\top$. For $\boldsymbol{x}=(x_1,x_2)\in\mathbb{R}^2$, denoting $|\boldsymbol{x}|:=(|x_1|^2+|x_2|^2)^{1/2}$ and the associated matrix norm $|\mathfrak{R}|=\sup_{\boldsymbol{x}\in\mathbb{R}^2\setminus\{0\}}|\mathfrak{R}\boldsymbol{x}|/|\boldsymbol{x}|$.

Lemma 8.1. *Under the condition that* $|\mathfrak{R}|<1$, *the sequence* $\widetilde{\boldsymbol{q}}^{(n)}$ *defined by* (6.3) *converges toward* $\boldsymbol{q}^{(\infty)}:=(\mathrm{Id}+\Pi\mathrm{S})^{-1}\boldsymbol{g}$ *the solution to* (5.2) *and we have* $\|\widetilde{\boldsymbol{q}}^{(n)}-\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}}\leq\|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}}|\mathfrak{R}|^{n+1}/(1-|\mathfrak{R}|)$ *for all* $n\geq0$.

Proof. First of all observe that $\|\widetilde{\boldsymbol{q}}^{(n)}-\widetilde{\boldsymbol{q}}^{(n-1)}\|_{\mathrm{T}^{-1}}\leq|\mathfrak{e}^{(n)}|\leq|\mathfrak{R}|^n\cdot|\mathfrak{e}^{(0)}|$. and $|\mathfrak{e}^{(0)}|=\|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}}$. Now pick arbitrary integers n,m with $m>n$. Under the assumption that $|\mathfrak{R}|<1$, we have the following estimate

$$\begin{aligned}\|\widetilde{\boldsymbol{q}}^{(m)}-\widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}&\leq\sum_{\nu=n+1}^{m}\|\widetilde{\boldsymbol{q}}^{(\nu)}-\widetilde{\boldsymbol{q}}^{(\nu-1)}\|_{\mathrm{T}^{-1}}\leq\|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}}\sum_{\nu=n+1}^{m}|\mathfrak{R}|^{\nu}\\&\leq\|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}}|\mathfrak{R}|^{n+1}/(1-|\mathfrak{R}|).\end{aligned}\quad(8.8)$$

This proves that the sequence $\widetilde{\boldsymbol{q}}^{(n)}$ is of Cauchy type in the norm $\|\cdot\|_{\mathrm{T}^{-1}}$ and admits a limit that we denote $\widetilde{\boldsymbol{q}}^{(\infty)}$. Letting $m\to\infty$ in (8.8) yields $\|\widetilde{\boldsymbol{q}}^{(n)}-\widetilde{\boldsymbol{q}}^{(\infty)}\|_{\mathrm{T}^{-1}}\leq\|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}}|\mathfrak{R}|^{n+1}/(1-|\mathfrak{R}|)$, so there only remains to prove that $\widetilde{\boldsymbol{q}}^{(\infty)}=\boldsymbol{q}^{(\infty)}$. Observe now that we also have

$$\|\widetilde{\boldsymbol{p}}_{\infty}^{(n)}-\widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}\leq\frac{|\mathfrak{e}^{(n)}|}{\sqrt{1+1/\epsilon_k}}\leq\frac{|\mathfrak{e}^{(0)}|\,|\mathfrak{R}|^n}{\sqrt{1+1/\epsilon_k}}$$

which proves that $\|\widetilde{\boldsymbol{p}}_{\infty}^{(n)}-\widetilde{\boldsymbol{p}}^{(n)}\|_{\mathrm{R}^*\mathrm{TR}}\to0$. Next, coming back to (6.3), recall that we have the relation $\widetilde{\boldsymbol{q}}^{(n+1)}=((1-\alpha)\mathrm{Id}-\alpha\Pi\mathrm{S})\widetilde{\boldsymbol{q}}^{(n)}-2\alpha\mathrm{TR}(\widetilde{\boldsymbol{p}}^{(n+1)}-\widetilde{\boldsymbol{p}}_{\infty}^{(n+1)})+\alpha\boldsymbol{g}$. Taking $n\to\infty$ in the previous relation, we obtain that $\widetilde{\boldsymbol{q}}^{(\infty)}$ satisfies the equation $\widetilde{\boldsymbol{q}}^{(\infty)}=((1-\alpha)\mathrm{Id}-\alpha\Pi\mathrm{S})\widetilde{\boldsymbol{q}}^{(\infty)}+\alpha\boldsymbol{g}$. After re-arrangement, this leads to $\widetilde{\boldsymbol{q}}^{(\infty)}=(\mathrm{Id}+\Pi\mathrm{S})^{-1}\boldsymbol{g}=\boldsymbol{q}^{(\infty)}$, which ends the proof. ∎

A remarkable conclusion that can be drawn from the previous lemma is that, with recycling, truncation of the PCG algorithm for the computation of the exchange operator Π does not induce any consistency error in the global DDM algorithm. Such is not the case when recycling is not used, as was shown through numerical experiments in the previous section.

Let us quantify more precisely the convergence rate. Because $\mathfrak{R}$ is symmetric, its norm $|\mathfrak{R}|$ equals its spectral radius. As a consequence, to bound the convergence rate provided by the previous lemma, we need to estimate the largest eigenvalue of $\mathfrak{R}$. This can be done explicitely by examining its characteristic polynomial,

$$\begin{aligned}\det(\lambda \mathrm{Id} - \mathfrak{R}) &= (\rho + 2\epsilon_k - \lambda)(2\epsilon_k - \lambda) - 4\epsilon_k(1+\epsilon_k)\\ &= \lambda^2 - \lambda(\rho + 4\epsilon_k) - (2-\rho)2\epsilon_k.\end{aligned} \tag{8.9}$$

Estimate (8.6) implies in particular that $\rho < 1$, so we see that $(2-\rho)2\epsilon_k > 0$ and that the roots of the characteristic polynomial (8.9) have opposite signs. Hence $|\mathfrak{R}|$ agrees with the positive root. With the gross estimate $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$, we deduce

$$\begin{aligned}&|\mathfrak{R}| = \frac{1}{2}(\rho + 4\epsilon_k) + \frac{1}{2}\sqrt{(\rho + 4\epsilon_k)^2 + 8(2-\rho)\epsilon_k},\\ &|\mathfrak{R}| \leq \rho + 4\epsilon_k + 2\sqrt{\epsilon_k}.\end{aligned} \tag{8.10}$$

To simplify the above estimate observe that, if $\rho + 4\sqrt{\epsilon_k} < 1$, then $2\sqrt{\epsilon_k} < 1 \Rightarrow 4\epsilon_k < 2\sqrt{\epsilon_k}$ and in this case $|\mathfrak{R}| \leq \rho + 4\epsilon_k + 2\sqrt{\epsilon_k} < \rho + 4\sqrt{\epsilon_k} < 1$. Let us examine what does the condition $\rho + 4\sqrt{\epsilon_k} < 1$ means. According to Estimate (8.6) to ensure $\rho + 4\sqrt{\epsilon_k} < 1$, it is sufficient that

$$\epsilon_k = 2\left(\frac{\sqrt{\mathrm{cond}(\mathrm{PR}^*\mathrm{TR})} - 1}{\sqrt{\mathrm{cond}(\mathrm{PR}^*\mathrm{TR})} + 1}\right)^k < \left(\alpha(1-\alpha)\gamma_h^2/4\right)^2 \tag{8.11}$$

Because ϵ_k decays exponentially fast to 0 as $k \to \infty$, which reflects the spectral convergence of the (preconditioned) conjugate gradient, only a few PCG iterations are necessary for satisfying (8.11). This is particularly true when the preconditioner P is devised appropriately. In addition, we underline that γ_h is h-uniformly lower bounded provided that the operator T is properly chosen like e.g. in Section 7, see the discussion following Proposition 5.2. The next proposition summarizes the previous discussion on convergence criterion and convergence rate.

Proposition 8.2. *Assume the number k of PCG iterations constant and chosen sufficiently large to satisfy Condition* (8.11). *Then $\rho + 4\sqrt{\epsilon_k} < 1$, and the sequence $\widetilde{\boldsymbol{q}}^{(n)}$ defined by* (6.3) *converges toward $\boldsymbol{q}^{(\infty)}$ solution to* (5.2) *with the error estimate*

$$\|\widetilde{\boldsymbol{q}}^{(n)} - \boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}} \leq \frac{(\rho + 4\sqrt{\epsilon_k})^n}{1 - (\rho + 4\sqrt{\epsilon_k})}\, \|\widetilde{\boldsymbol{q}}^{(0)}\|_{\mathrm{T}^{-1}}.$$

This result provides a theoretical justification for Figure 7.3 and Table 7.1. Truncating PCG in the exchange operation has an effect on the convergence rate of the approximate Richardson Algorithm (6.3). This effect is quantified by $4\sqrt{\epsilon_k}$, so this perturbation decreases exponentially with k according to (8.11). This is why only a few iterations of PCG suffice to maintain convergence.

How efficient is this trick depends on the performance of the preconditioner P according to (8.11). Pushing the analysis further in this respect requires more information on P and has to be conducted on a case by case basis regarding the choice of this preconditioner, which is beyond the scope of the present article.

Appendix. further numerical experiments

In the present section we come back to the numerical setup considered in Section 7, and investigate the sensitivity of our acceleration procedure with respect to various parameters of the domain decomposition method. We consider the very same boundary value problem (2.1) as in Section 7 in the same domain $\Omega = (-1,1)^2 \setminus [-0.25, 0.25]^2$, with a wavelength $\lambda = 1/5$ corresponding to a wavenumber $\kappa \simeq 31.41$. The real part of the reference solution is represented in Figure A.1. We recall the significance of a few parameters:

- h is the meshwidth i.e. $h = \max\{\mathrm{diam}(\tau),\ \tau \in \mathcal{T}_h(\Omega)\}$.

- J is the number of subdomains i.e. $\overline{\Omega} = \overline{\Omega}_1 \cup \cdots \cup \overline{\Omega}_{\mathrm{J}}$.

- α is the relaxation parameter involved in the Richardson solver (5.4).

- k_{MAX} is the maximum number of PCG iterations performed (for each outer iteration n) in the first line of (6.3). It is a truncation parameter.

- #iter is the number of outer iterations n in (6.3) required for the relative error to satisfy $\|\boldsymbol{q}^{(\infty)} - \widetilde{\boldsymbol{q}}^{(n)}\|_{\mathrm{T}^{-1}}/\|\boldsymbol{q}^{(\infty)}\|_{\mathrm{T}^{-1}} < 1e - 10$.

Like for Table 7.1, we systematically examine the value of #iter as k_{MAX} ranges from 1 to 20, for different values of h, α, J. To evaluate #iter, we run Algorithm (6.3) that relies on both the recycling strategy described in Section 6, 7 and 8, and a truncation consisting in stopping PCG so that the dimension k of the Krylov space does not get larger than k_{MAX}. We consider a nominal configuration corresponding to the parameters $h \simeq 0.00552917, \dim \mathrm{V}_h(\Omega) = 175794, \alpha = 0.5, \mathrm{J} = 16$, and all the results we present are variations around this nominal configuration.

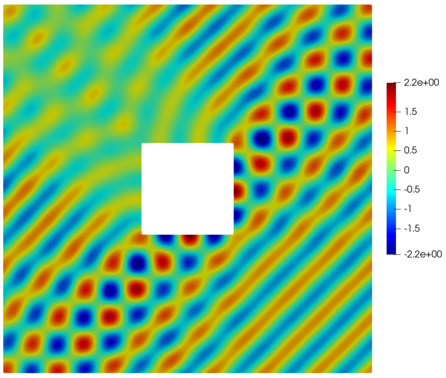

Figure A.1.

Influence of the mesh size. In Table A.1, we consider a fixed partition in $\mathrm{J} = 16$ subdomains, a fixed relaxation parameter $\alpha = 0.5$, and we examine the number of outer iterations #iter as k_{MAX} and h vary. On each line of Table A.1, #iter remains approximately constant except for $k_{\mathrm{MAX}} = 1$ where it substantially deteriorates.

Table A.1. #iter for varying h and k_{MAX}

		k_{MAX}						
h	$\dim \mathrm{V}_h(\Omega)$	20	15	10	5	3	2	1
0.0218929	11170	222	222	222	222	222	222	232
0.0111801	43947	216	216	216	216	216	217	297
0.00552917	175794	233	233	233	233	233	233	734
0.00282605	696832	292	292	292	292	293	296	820

Influence of Richardson's relaxation parameter. In Table A.2, we consider a fixed partition with $\mathrm{J} = 16$ subdomains, and a fixed mesh $h \simeq 0.00552917$ and $\dim \mathrm{V}_h(\Omega) = 175794$, and we examine the number of outer iterations #iter as k_{MAX} and α vary. On each line of Table A.2, again, we see that #iter remains approximately constant until $k_{\mathrm{MAX}} = 1$ or $k_{\mathrm{MAX}} = 2$ where it deteriorates. Interestingly, although seemingly anecdotal, for $\alpha = 0.9$, we see that #iter is smaller for $k_{\mathrm{MAX}} = 2$ than for $k_{\mathrm{MAX}} = 5$.

Table A.2. #iter for varying α and k_{MAX}

	k_{MAX}						
α	20	15	10	5	3	2	1
0.1	1061	1061	1061	1061	1062	1062	1264
0.25	434	434	434	434	435	435	686
0.5	233	233	233	233	233	233	734
0.75	199	199	199	199	204	239	968
0.9	304	304	304	304	259	262	1106

Influence of the number of subdomains. In Table A.3, we consider a fixed mesh $h \simeq 0.00552917$ and $\dim \mathrm{V}_h(\Omega) = 175794$, and a fixed value $\alpha = 0.5$ for the relaxation parameter of Richardson's solver, and we examine the number of outer iterations #iter as k_{MAX} and the number of subdomains J vary. On each line of Table A.3, again, we see that #iter remains approximately constant until $k_{\mathrm{MAX}} = 1$ or $k_{\mathrm{MAX}} = 2$ where it deteriorates.

Table A.3. #iter for varying J and k_{MAX}

	k_{MAX}						
J	20	15	10	5	3	2	1
4	197	197	197	197	198	202	425
8	210	210	210	210	211	215	439
16	233	233	233	233	233	233	734
32	278	278	278	278	278	280	535
64	342	342	342	343	343	344	948

Importance of the preconditioner. In Table A.4, we take $\alpha = 0.5$ and $\mathrm{J} = 16$, and we consider two different meshes ($h \simeq 0.0055$ and $h \simeq 0.0028$) and, depending on the value of k_{MAX}, we record #iter. This time however, we do *not* use a preconditioner and apply the conjugate gradient (CG) instead of the preconditioned conjugate gradient (PCG) on the first line of (6.3). We still use recycling to choose the initial guess. Comparing with the last two rows of Table A.1, this shows the benefit of preconditioning.

Table A.4. #iter for varying h and k_{MAX}, using CG instead of PCG to compute Π

		k_{MAX}						
h	$\dim \mathrm{V}_h(\Omega)$	20	15	10	5	3	2	1
0.00552917	175794	233	233	233	267	512	1299	>5000
0.00282605	696832	292	290	296	440	1289	3356	>5000

Conclusion. The previous tables all agree on the same trend: for a given mesh, a given value of α, J, the number of outer iterations #iter remains independent of k_{MAX} unless k_{MAX} is below a certain threshold value. In our numerical setup, with a single level Neumann–Neumann preconditioner used in PCG (in first line of (6.3)), this threshold value is $k_{\mathrm{MAX}} = 1$ or $k_{\mathrm{MAX}} = 2$. However this threshold is sensitive to the preconditioner used in PCG, which is indeed consistent with (8.11). It would be desirable to provide an explicit estimate, or some automatic procedure for determining it. We shall investigate this point in a future contribution.

References

[1] Bernhard Beckermann, Sergei A. Goreinov, and Evgeniĭ E. Tyrtyshnikov. Some remarks on the Elman estimate for GMRES. *SIAM J. Matrix Anal. Appl.*, 27(3):772–778, 2006.

[2] Xavier Claeys. Non-local variant of the Optimised Schwarz Method for arbitrary non-overlapping subdomain partitions. *ESAIM, Math. Model. Numer. Anal.*, 55(2):429–448, 2021.

[3] Xavier Claeys. Nonselfadjoint impedance in generalized optimized Schwarz methods. *IMA J. Numer. Anal.*, 43(5):3026–3054, 2023.

[4] Xavier Claeys, Francis Collino, and Emile Parolin. Nonlocal optimized Schwarz methods for time-harmonic electromagnetics. *Adv. Comput. Math.*, 48(6): article no. 72 (36 pages), 2022.

[5] Xavier Claeys and Emile Parolin. Robust treatment of cross-points in Optimized Schwarz Methods. *Numer. Math.*, 151(2):405–442, 2022.

[6] Francis Collino, Patrick Joly, and Emile Parolin. Non-local impedance operator for non-overlapping DDM for the Helmholtz equation. In *Domain decomposition methods in science and engineering XXVI*, volume 145 of *Lecture Notes in Computational Science and Engineering*, pages 725–733. Springer, 2022.

[7] Bruno Després. *Méthodes de décomposition de domaine pour les problèmes de propagation d'ondes en régime harmonique. Le théorème de Borg pour l'équation de Hill vectorielle.* PhD thesis, Université de Paris IX (Dauphine), France, 1991.

[8] Victorita Dolean, Pierre Jolivet, and Frédéric Nataf. *An introduction to domain decomposition methods. Algorithms, theory, and parallel implementation*, volume 144 of *Other Titles in Applied Mathematics.* Society for Industrial and Applied Mathematics, 2015.

[9] Martin J. Gander and Hui Zhang. A class of iterative solvers for the Helmholtz equation: factorizations, sweeping preconditioners, source transfer, single layer potentials, polarized traces, and optimized Schwarz methods. *SIAM Rev.*, 61(1):3–76, 2019.

[10] Jörg Liesen and Zdeněk Strakoš. *Krylov subspace methods.* Numerical Mathematics and Scientific Computation. Oxford University Press, 2013.

[11] Émile Parolin. *Non-overlapping domain decomposition methods with non-local transmissionoperators for harmonic wave propagation problems.* PhD thesis, Institut Polytechnique de Paris, France, 2020.

[12] Yousef Saad. *Iterative methods for sparse linear systems.* Society for Industrial and Applied Mathematics, 2003.

[13] Kirk M. Soodhalter, Eric de Sturler, and Misha E. Kilmer. A survey of subspace recycling iterative methods. *GAMM-Mitt.*, 43(4): article no. e202000016 (29 pages), 2020.

[14] Andrea Toselli and Olof Widlund. *Domain decomposition methods — algorithms and theory*, volume 34 of *Springer Series in Computational Mathematics.* Springer, 2005.