Noname manuscript No.
(will be inserted by the editor)

Linear and nonlinear substructured Restricted Additive Schwarz iterations and preconditioning

F. Chaouqui · M.J. Gander · P.M. Kumbhar · T. Vanzan

Received: date / Accepted: date

Abstract Iterative substructuring Domain Decomposition (DD) methods have been extensively studied, and they are usually associated with nonoverlapping decompositions. It is less known that classical overlapping DD methods can also be formulated in substructured form, i.e. as iterative methods acting on variables defined exclusively on the interfaces of the overlapping domain decomposition. We call such formulations substructured domain decomposition methods. We introduce here a substructured version of Restricted Additive Schwarz (RAS) which we call SRAS. We show that RAS and SRAS are equivalent when used as iterative solvers, as they produce the same iterates, while they are substantially different when used as preconditioners for GMRES. We link the volume and substructured Krylov spaces and show that the iterates are different by deriving the least squares problems solved at each GMRES iteration. When used as iterative solvers, SRAS presents computational advantages over RAS, as it avoids computations with matrices and vectors at the volume level. When used as preconditioners, SRAS has the further advantage of allowing GMRES to store smaller vectors and perform orthogonalization in a lower dimensional space. We then consider nonlinear problems, and we introduce SRASPEN (Substructured Restricted Additive Schwarz Precondi-

F. Chaouqui
Temple University, USA
E-mail: faycal.chaouqui@temple.edu

M.J. Gander
Université de Genève, Switzerland
E-mail: martin.gander@unige.ch

P.M. Kumbhar
IANM, Karlsruhe Institute of Technology, Germany
E-mail: pratik.kumbhar@kit.edu

T. Vanzan
CSQI Chair, École Polytecnique Fédérale de Lausanne, Switzerland
E-mail: tommaso.vanzan@epfl.ch

tioned Exact Newton), where SRAS is used as a preconditioner for Newton's method. In contrast to the linear case, we prove that Newton's method applied to the preconditioned volume and substructured formulation produces the same iterates in the nonlinear case. Next, we introduce two-level versions of nonlinear SRAS and SRASPEN. Finally, we validate our theoretical results with numerical experiments.

Keywords Substructured domain decomposition methods · Lions' parallel Schwarz method · Restricted Additive Schwarz (RAS) · Linear and Nonlinear Preconditioning · GMRES.

1 Introduction

We consider a boundary value problem posed in a Lipschitz domain $\Omega \subset \mathbb{R}^d$, $d \in \{1, 2, 3\}$,

$$\begin{aligned} \mathcal{L}(u) &= f, \quad \text{in } \Omega, \\ u &= 0, \quad \text{on } \partial\Omega. \end{aligned} \tag{1}$$

We assume that (1) admits a unique solution in some Hilbert space $\mathcal{V}$. If the boundary value problem is linear, a discretization of (1) with N_v degrees of freedom leads to a linear system

$$A\mathbf{u} = \mathbf{f}, \tag{2}$$

where $A \in \mathbb{R}^{N_v \times N_v}$, $\mathbf{u} \in V(\cong \mathbb{R}^{N_v})$, and $\mathbf{f} \in V$. If the boundary value problem is nonlinear, we obtain a nonlinear system

$$F(\mathbf{u}) = 0, \tag{3}$$

where $F : V \to V$ is a nonlinear function and $\mathbf{u} \in V$. Several numerical methods have been proposed in the last decades for the efficient solution of such boundary value problems, e.g., multigrid methods [21,37] and domain decomposition (DD) methods [36,34]. We will focus on DD methods, which are usually divided into two distinct classes, that is overlapping methods, which include the AS (Additive Schwarz) and RAS (Restricted Additive Schwarz) methods [36,6], and nonoverlapping methods such as FETI (Finite Element Tearing and Interconnect) and Neumann-Neumann methods [14,32,26]. Concerning nonlinear problems, DD methods can be applied either as nonlinear iterative methods, that is by just solving nonlinear problems in each subdomain and then exchanging information between subdomains as in the linear case [29,30,2], or as preconditioners to solve the Jacobian linear system inside Newton's iteration. In the latter case, the term Newton-Krylov-DD is employed, where DD is replaced by the domain decomposition preconditioner used [23].

An alternative is to use a DD method as a preconditioner for Newton's method. Preconditioning a nonlinear system $F(\mathbf{u}) = 0$ means that we aim to replace the original nonlinear system with a new nonlinear system, still

having the same solution, but for which the nonlinearities are more balanced and Newton's method converges faster [3,17]. Seminal contributions in nonlinear preconditioning have been made by Cai and Keyes in [3,4], where they introduced ASPIN (Additive Schwarz Preconditioned Inexact Newton). The development of good preconditioners is not an easy task even in the linear case. One useful strategy is to study efficient iterative methods, and then to use the associated preconditioners in combination with Krylov methods [17]. The same logical path paved the way to the development of RASPEN (Restricted Additive Schwarz Preconditioned Exact Newton) in [13], which in short applies Newton's method to the fixed point equation defined by the nonlinear RAS iteration at convergence. Extensions of this idea to Dirichlet-Neumann are presented in [7]. In [22], the authors describe and analyze the scalability of the two-level variants of the aformentioned methods (ASPIN and RASPEN). In particular, they discuss several approaches of adding the coarse space correction, and a numerical comparison of all these methods is reported for different types of coarse spaces. All these methods are left preconditioners. Right preconditioners are usually based on the concept of nonlinear elimination, presented in [27], and they are very efficient as shown in [20,19,5,31]. Right nonlinear preconditioners based on FETI-DP (Finite Element Tearing and InterconnectingDual-Primal) and BDDC (Balancing Domain Decomposition by Constraint) have been shown to be very effective, see, e.g., [24,25]. While left preconditioners aim to transform the original nonlinear function into a better behaved one, right preconditioners aim to provide a better initial guess for the next outer Newton iteration.

Nonoverlapping methods are sometimes called substructuring methods (a term borrowed from Przemieniecki's work [33]), as in these methods the unknowns in the interior of the nonoverlapping domains are eliminated through static condensation so that one needs to solve a smaller system involving only the degrees of freedom on the interfaces between the nonoverlapping subdomains [36]. However, it is also possible to write an overlapping method, such as Lions' Parallel Schwarz Method (PSM) [28], which is equivalent to RAS [16], in substructured form, even though this approach is much less common in the literature. For a two subdomain decomposition, a substructuring procedure applied to the PSM is carried out in [15, Section 5], [18, Section 3.4] and [10]. In [10,11], the authors introduced a substructured formulation of the PSM at the continuous level for decompositions with many subdomains and cross-points, and further studied ad-hoc spectral and geometric two-level methods. In this particular framework, the substructured unknowns are now the degrees of freedom located on the portions of a subdomain boundary that lie in the interior of another subdomain; that is where the overlapping DD method takes the information to compute the new iterate. We emphasize that, at a given iteration n, any iterative DD method (overlapping or nonoverlapping) needs only a few values of $\mathbf{u}^n$ to compute the new approximation $\mathbf{u}^{n+1}$. The major part of $\mathbf{u}^n$ is useless.

In this manuscript, we define a substructured version of RAS, that is we define an iterative scheme based on RAS which acts only over unknowns that

are located on the portions of a subdomain boundary that lie in the interior of another subdomain. We study in detail the effects that such a substructuring procedure has on RAS when the latter is applied either as an iterative solver or as a preconditioner to solve linear and nonlinear boundary value problems. Does the substructured iterative version converge faster than the volume one? Is the convergence of GMRES affected by substructuring? What about nonlinear problems when instead of preconditioned GMRES we rely on preconditioned Newton? We prove that substructuring does not influence the convergence of the iterative methods both in the linear and nonlinear case, by showing that at each iteration, the restriction on the interfaces of the volume iterates coincides with the iterates of the substructured iterative method. Nevertheless, we discuss in Section 3.1 and corroborate by numerical experiments that a substructured formulation presents computational advantages. The equivalence of iterates does not hold anymore when considering preconditioned GMRES. Specifically, our study shows that GMRES should be applied to the substructured system, since it is computationally less expensive, requiring to perform orthogonalization on a much smaller space, and thus needs also less memory. In contrast to the linear case, we prove that, surprisingly, the nonlinear preconditioners RASPEN and SRASPEN (Substructured RASPEN) for Newton produce the same iterates once these are restricted to the interfaces. However, SRASPEN has again more favorable properties when assembling and solving the Jacobian matrices at each Newton iteration. Finally, we also extend the work in [10,11] defining substructured two-level methods to the nonlinear case, where both smoother and coarse correction are defined directly on the interfaces between subdomains.

This paper is organized as follows: we introduce in Section 2 the mathematical setting with the domain, subdomains and operators defined on them. In Section 3, devoted to the linear case, we study the effects of substructuring on RAS and on GMRES applied to the preconditioned system. In Section 4, we extend our analysis to nonlinear boundary value problems. Section 5 contains two-level substructured methods for the nonlinear problems. Finally Section 6 presents numerical tests to corroborate the framework proposed.

2 Notation

Let us decompose the domain Ω into N nonoverlapping subdomains Ω_j, that is $\Omega = \bigcup_{j\in\mathcal{J}} \Omega_j$ with $\mathcal{J} := \{1, 2, \ldots, N\}$. The nonoverlapping subdomains Ω_j are then enlarged to obtain subdomains Ω_j' which form an overlapping decomposition of Ω. For each subdomain Ω_j', we define V_j as the restriction of V to Ω_j', that is V_j collects the degrees of freedom on Ω_j'. Further, we introduce the classical restriction and prolongation operators $R_j : V \to V_j$, $P_j : V_j \to V$, and the restricted prolongation operators $\widetilde{P}_j : V_j \to V$. We assume that these operators satisfy

$$R_j P_j = I_{V_j}, \quad \text{and} \quad \sum_{j\in\mathcal{J}} \widetilde{P}_j R_j = I, \tag{4}$$

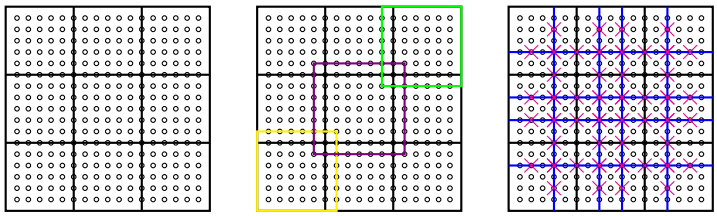

Fig. 1: The domain Ω is divided into nine nonoverlapping subdomains (left). The center panel shows how the diagonal nonoverlapping subdomains are enlarged to form overlapping subdomains. On the right, we denote the unknowns represented in $\overline{V}$ (blue line) and the unknowns of a coarse space of $\overline{V}$ (red crosses).

where I_{V_j} is the identity on V_j and I is the identity on V.

We now define the substructured skeleton. In the following, we use the notation introduced in [9]. For any $j \in \mathcal{J}$, we define the set of neighboring indices $N_j := \{\ell \in \mathcal{J} \ : \ \Omega'_j \cap \partial\Omega'_\ell \neq \emptyset\}$. Given a $j \in \mathcal{J}$, we introduce the substructure of Ω'_j defined as $S_j := \bigcup_{\ell \in N_j} \left(\partial\Omega'_\ell \cap \Omega'_j\right)$, that is the union of all the portions of $\partial\Omega'_\ell$ with $\ell \in N_j$. The substructure of the whole domain Ω is defined as $S := \bigcup_{j\in\mathcal{J}} \overline{S_j}$. A graphical representation of S is given in Fig. 1 for a decomposition of a square into nine subdomains. We now introduce the space $\overline{V}$ as the trace space of V onto the substructure S. Associated to $\overline{V}$, we consider the restriction operator $\overline{R} : V \to \overline{V}$ and a prolongation operator $\overline{P} : \overline{V} \to V$. The restriction operator $\overline{R}$ takes an element $v \in V$ and restricts it to the skeleton S. The prolongation operator $\overline{P}$ extends an element $v \in \overline{V}$ to the global space V. In our numerical experiments, $\overline{P}$ extends an element $v_S \in \overline{V}$ by zero in $\Omega \backslash S$. However, we can consider several different prolongation operators. How this extension is done is not crucial as we will use $\overline{P}$ inside a domain decomposition algorithm, and thus only the values on the skeleton S will play a role. Hence, as of now, we will need only one assumption on the restriction and prolongation operator, namely

$$\overline{RP} = \overline{I}, \tag{5}$$

where $\overline{I}$ is the identity over $\overline{V}$.

3 The linear case

In this section, we focus on the linear problem $A\mathbf{u} = \mathbf{f}$. After defining a substructured variant of RAS called SRAS, we prove the equivalence between RAS and SRAS. Then, we study in detail how GMRES performs if applied to the volume preconditioned system or the substructured system.

3.1 Linear iterative methods

To introduce our analysis, we recall the classical definition of RAS to solve the linear system (2). RAS starts from an approximation $\mathbf{u}^0$ and computes for $n = 1, 2, \ldots,$

$$\mathbf{u}^n = \mathbf{u}^{n-1} + \sum_{j \in \mathcal{J}} \widetilde{P}_j A_j^{-1} R_j \left(\mathbf{f} - A\mathbf{u}^{n-1}\right), \tag{6}$$

where $A_j := R_j A P_j$, that is, we use exact local solvers. Let us now rewrite the iteration (6) in an equivalent form using the hypothesis in (4) and the definition of A_j,

$$\begin{aligned}
\mathbf{u}^n &= \sum_{j \in \mathcal{J}} \widetilde{P}_j R_j \mathbf{u}^{n-1} + \sum_{j \in \mathcal{J}} \widetilde{P}_j A_j^{-1} R_j \left(\mathbf{f} - A\mathbf{u}^{n-1}\right) \\
&= \sum_{j \in \mathcal{J}} \widetilde{P}_j A_j^{-1} \left(A_j R_j \mathbf{u}^{n-1} + R_j \left(\mathbf{f} - A\mathbf{u}^{n-1}\right)\right) \\
&= \sum_{j \in \mathcal{J}} \widetilde{P}_j A_j^{-1} R_j \left(\mathbf{f} - A\left(I - P_j R_j\right)\mathbf{u}^{n-1}\right) \\
&=: G^{\mathrm{RAS}}(\mathbf{u}^{n-1}).
\end{aligned} \tag{7}$$

We emphasize that $\left(P_j R_j - I\right)\mathbf{u}^{n-1}$ contains non-zero elements only outside subdomain Ω_j', and in particular the terms $A\left(P_j R_j - I\right)\mathbf{u}^{n-1}$ represent precisely the boundary conditions for Ω_j' given the old approximation $\mathbf{u}^{n-1}$. This observation suggests that RAS, like most domain decomposition methods, can be written in a substructured form. Indeed, despite iteration (7) being written in volume form, involving the entire vector $\mathbf{u}^{n-1}$, only very few elements of $\mathbf{u}^{n-1}$ are needed to compute the new approximation $\mathbf{u}^n$. A substructured method iterates only on those values of $\mathbf{u}$ which are really needed at each iteration, avoiding thus superfluous operations on the whole volume vector $\mathbf{u}$ (e.g. the volume residual computation $\mathbf{f} - A\mathbf{u}^{n-1}$ and the summation with the old iterate $\mathbf{u}^{n-1}$ in the RAS method (6)). For further details about a substructured formulation of the parallel Schwarz method at the continuous level, we refer to [18] for the two subdomain case, and [10,11] for a general decomposition into several subdomains with crosspoints.

In Section 2, we introduced the substructured space $\overline{V}$ geometrically, but we can also provide an algebraic characterization using the RAS operators R_j and P_j. We consider

$$\mathcal{K} := \left\{k \in \{1, \ldots, N_v\} : \exists j \in \{1, \ldots, N\} \text{ such that } R_j A(\mathbf{e}_k - P_j R_j \mathbf{e}_k) \neq 0\right\},$$

that is, $\mathcal{K}$ is the set of indices such that the canonical vectors $\mathbf{e}_k$ represent a Dirichlet boundary condition at least for a subdomain, and its complement $\mathcal{K}^c := \{1, \ldots, N_v\} \backslash \mathcal{K}$. The cardinality of $\mathcal{K}$ is $|\mathcal{K}| =: \overline{N}$. We can thus introduce

$$\widehat{V} := \left\{\mathbf{v} \in \mathbb{R}^{N_v} : \text{ if } j \notin \mathcal{K} \text{ then } v_j = 0\right\} = \operatorname{span}\left\{\mathbf{e}_k\right\}_{k \in \mathcal{K}} \subset \mathbb{R}^{N_v}.$$

Finally $\overline{R}$ is the Boolean restriction operator, mapping a vector of $\mathbb{R}^{N_v}$ onto a vector of $\mathbb{R}^{\overline{N}}$, keeping only the indices in $\mathcal{K}$. Hence, $\overline{V} := \mathrm{Im}\overline{R}(\cong \mathbb{R}^{\overline{N}})$ and $\overline{P} = \overline{R}^{\top}$.

To define SRAS, we need one more assumption on the restriction and prolongation operators, namely

$$\overline{R}M^{-1}A = \overline{R}M^{-1}A\overline{P}\overline{R}, \tag{8}$$

where M^{-1} is the preconditioner for RAS, formally defined as

$$M^{-1} := \sum_{j\in\mathcal{J}} \widetilde{P}_j A_j^{-1} R_j. \tag{9}$$

Heuristically, this assumption means that the operator $\overline{PR}$ preserves all the information needed by G^{RAS} (defined in (7)) to compute correctly the values of the new iterate on the skeleton S. Indeed a direct calculation shows that (8) is equivalent to the condition

$$\overline{R}G^{\mathrm{RAS}}(\mathbf{u}) = \overline{R}G^{\mathrm{RAS}}(\overline{PR}\mathbf{u}).$$

Given a substructured approximation $\mathbf{v}^0 \in \overline{V}$, for $n = 1, 2, \ldots$, we define SRAS as

$$\mathbf{v}^n = G^{\mathrm{SRAS}}(\mathbf{v}^{n-1}), \quad \text{where} \quad G^{\mathrm{SRAS}}(\mathbf{v}) := \overline{R}G^{\mathrm{RAS}}(\overline{P}\mathbf{v}). \tag{10}$$

RAS and SRAS are tightly linked, but when are they equivalent? Clearly, we must impose some conditions on $\overline{P}$ and $\overline{R}$. The next theorem shows that assumption (8) is in fact sufficient for equivalence.

Theorem 1 (Equivalence between RAS and SRAS) *Assume that the operators $\overline{R}$ and $\overline{P}$ satisfy* (8)*. Given an initial guess $\mathbf{u}^0 \in V$ and its substructured restriction $\mathbf{v}^0 := \overline{R}\mathbf{u}^0 \in \overline{V}$, define the sequences $\{\mathbf{u}^n\}$ and $\{\mathbf{v}^n\}$ such that*

$$\mathbf{u}^n = G^{RAS}(\mathbf{u}^{n-1}), \quad \mathbf{v}^n = G^{SRAS}(\mathbf{v}^{n-1}).$$

Then, $\overline{R}\mathbf{u}^n = \mathbf{v}^n$ for every iteration $n \geq 1$.

Proof We prove this statement for $n = 1$ by a direct calculation. Taking the restriction of $\mathbf{u}^1$ we have

$$\overline{R}\mathbf{u}^1 = \overline{R}G^{\mathrm{RAS}}(\mathbf{u}^0) = \overline{R}G^{\mathrm{RAS}}(\overline{PR}\mathbf{u}^0) = \overline{R}G^{\mathrm{RAS}}(\overline{P}\mathbf{v}^0) = G^{\mathrm{SRAS}}(\mathbf{v}^0) = \mathbf{v}^1,$$

where we used assumption (8), and the definition of $\mathbf{v}^0$ and G^{SRAS}. For a general n, the proof is obtained by induction.

Remark 1 (Implementation of SRAS) In (10), we have introduced the substructured operator G^{SRAS} directly through the volume operator G^{RAS}. This definition is very useful from the theoretical point of view as it permits to link the volume and substructured methods and facilitates the theoretical analysis. However, we stress that one should not implement G^{SRAS} by directly calling

the volume routine G^{RAS} onto a vector $\overline{P}\mathbf{v}$. Doing so, one would lose all the computational advantages as computations on the volume vector $\overline{P}v$ would be performed. There are two strategies to implement a fully substructured SRAS method. The first one is to implement a routine that, for each $j = 1, \ldots, N$, extracts from $\mathbf{v}$ those values which lie on the boundary of Ω_j' and rescale them appropriately as boundary conditions for the local subdomain solve. A second possibility arises from (10) and (7), by observing that

$$\begin{aligned}\mathbf{v}^n &= \overline{R}G^{\mathrm{RAS}}(\overline{P}\mathbf{v}^n) = \overline{R}\sum_{j=1}^{N}\widetilde{P}_j A_j^{-1} R_j(\mathbf{f} - A(I - P_j R_j)\overline{P}\mathbf{v}^{n-1} \\ &= \mathbf{b} + \overline{R}\sum_{j=1}^{N}\widetilde{P}_j A_j^{-1} R_j(-A(I - P_j R_j)\overline{P}\mathbf{v}^{n-1},\end{aligned} \tag{11}$$

where $\mathbf{b} := \overline{R}\sum_{j=1}^{N}\widetilde{P}_j A_j^{-1} R_j f$. One can then pre-assemble the matrices $\overline{R}_j := R_j(-A(I - P_j R_j)\overline{P} \in \mathbb{R}^{M_j,\overline{N}}$, where M_j is the number of degrees of freedom in Ω_j'. The matrices $\overline{R}_j$ are very sparse and their goal is just to extract the values needed for the j-th subdomain solve from $\mathbf{v}^{n-1}$. Similarly $\overline{P}_j := \overline{R}\widetilde{P}_j \in \mathbb{R}^{\overline{N},M_j}$ weights a subdomain solution with the partition of unity and maps it to the substructured vector. We finally obtain the equivalent iterative method

$$\mathbf{v}^n = \mathbf{b} + \sum_{j=1}^{N}\overline{P}_j A_j^{-1}\overline{R}_j\mathbf{v}^{n-1}, \tag{12}$$

where no computations are performed with matrices or vectors at the volume level, except for the subdomain solves.

3.2 Linear preconditioners for GMRES

It is well known that any stationary iterative method should be used in practice as a preconditioner for a Krylov method, since the Krylov method finds in general a much better residual polynomial with certain optimality properties, compared to the residual polynomial of the stationary iteration, see e.g. [8]. The preconditioner associated to RAS is M^{-1} and is defined in (9). The preconditioned volume system then reads

$$M^{-1}A\mathbf{u} = M^{-1}\mathbf{f}. \tag{13}$$

To discover the preconditioner associated with SRAS, we consider the fixed point limit of (10),

$$\begin{aligned}\mathbf{v} = G^{\mathrm{SRAS}}(\mathbf{v}) &= \overline{R}G^{\mathrm{RAS}}(\overline{P}\mathbf{v}) = \overline{R}\left(\overline{P}\mathbf{v} + \sum_{j\in\mathcal{J}}\widetilde{P}_j A_j^{-1} R_j(\mathbf{f} - A\overline{P}\mathbf{v})\right) \\ &= \mathbf{v} + \overline{R}\sum_{j\in\mathcal{J}}\widetilde{P}_j A_j^{-1} R_j\mathbf{f} - \overline{R}\sum_{j\in\mathcal{J}}\widetilde{P}_j A_j^{-1} R_j A\overline{P}\mathbf{v} \\ &= \mathbf{v} + \overline{R}M^{-1}\mathbf{f} - \overline{R}M^{-1}A\overline{P}\mathbf{v},\end{aligned} \tag{14}$$

where in the second line we used the identity $\overline{RP} = \overline{I}$. We thus consider the preconditioned substructured system

$$\overline{R}M^{-1}A\overline{P}\mathbf{v} = \overline{R}M^{-1}\mathbf{f}. \tag{15}$$

Observe that as $A = M - N$, (15) can be written as $(\overline{I} - \overline{G})\mathbf{v} = \mathbf{b}$, where $\overline{G} := \overline{R}M^{-1}N\overline{P}$ and $\mathbf{b} := \overline{R}M^{-1}f$, thus recovering the classical form of the substructured PSM, see [18,10,11].

It is then natural to ask how a Krylov method like GMRES performs if applied to (13), compared to (15). Let us consider an initial guess in volume $\mathbf{u}^0$, its restriction $\mathbf{v}^0 := \overline{R}\mathbf{u}^0$ and the initial residuals $\mathbf{r}^0 := M^{-1}(\mathbf{f} - A\mathbf{u}^0)$, $\overline{\mathbf{r}}^0 := \overline{R}M^{-1}(\mathbf{f} - A\overline{P}\mathbf{v}^0)$. Then GMRES applied to the preconditioned systems (13) and (15) looks for solutions in the affine Krylov spaces

$$\begin{aligned}\mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0) &:= \mathbf{u}^0 + \ \text{span}\left\{\mathbf{r}^0, M^{-1}A\mathbf{r}^0, \ldots, (M^{-1}A)^{k-1}\mathbf{r}^0\right\}\\ \mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \overline{\mathbf{r}}^0) &:= \mathbf{v}^0 + \ \text{span}\left\{\overline{\mathbf{r}}^0, \overline{R}M^{-1}A\overline{P}\overline{\mathbf{r}}^0, \ldots, (\overline{R}M^{-1}A\overline{P})^{k-1}\overline{\mathbf{r}}^0\right\},\end{aligned} \tag{16}$$

where $k \geq 1$. The two Krylov spaces are tightly linked, as Theorem 2 below will show. To prove it, we need the following Lemma.

Lemma 1 *If the restriction and prolongation operators $\overline{R}$ and $\overline{P}$ satisfy* (8) *then for $k \geq 1$,*

$$\overline{R}\left(M^{-1}A\right)^k = \left(\overline{R}M^{-1}A\overline{P}\right)^k \overline{R}. \tag{17}$$

Proof Multiplying equation (8) from the right by $M^{-1}A$ we get

$$\overline{R}M^{-1}AM^{-1}A = \overline{R}M^{-1}A\overline{P}\overline{R}M^{-1}A = \overline{R}M^{-1}A\overline{P}\overline{R}M^{-1}A\overline{P}\overline{R},$$

where in the second equality we have used once more (8). Using induction, one gets for every $k \geq 1$,

$$\overline{R}\left(M^{-1}A\right)^k = \left(\overline{R}M^{-1}A\overline{P}\right)^k \overline{R},$$

and this completes the proof.

Theorem 2 (Relation between RAS and SRAS Krylov subspaces) *Let us consider operators $\overline{R}$ and $\overline{P}$ satisfying* (8)*, an initial guess $\mathbf{u}^0 \in V$, its restriction $\mathbf{v}^0 := \overline{R}\mathbf{u}^0 \in \overline{V}$ and the residuals $\mathbf{r}^0 := M^{-1}(\mathbf{f} - A\mathbf{u}^0)$, $\overline{\mathbf{r}}^0 := \overline{R}M^{-1}(\mathbf{f} - A\overline{P}\mathbf{v}^0)$. Then for every $k \geq 1$, we have*

$$\mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \overline{\mathbf{r}}^0) = \overline{R}(\mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0)). \tag{18}$$

Proof First, due to (8) we have

$$\overline{R}\mathbf{r}^0 = \overline{R}M^{-1}(\mathbf{f} - A\mathbf{u}^0) = \overline{R}M^{-1}(\mathbf{f} - A\overline{P}\overline{R}\mathbf{u}^0) = \overline{R}M^{-1}(\mathbf{f} - A\overline{P}\mathbf{v}^0) = \overline{\mathbf{r}}^0.$$

Let us now show the first inclusion. If $\mathbf{v} \in \overline{R}\left(\mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0)\right)$, then $\mathbf{v} = \overline{R}\mathbf{u}^0 + \overline{R}\sum_{j=0}^{k-1}\gamma_j\left(M^{-1}A\right)^j\mathbf{r}^0$, for some coefficients γ_j. Using Lemma 1, we can rewrite $\mathbf{v}$ as

$$\begin{aligned}\mathbf{v} &= \mathbf{v}^0 + \sum_{j=0}^{k-1}\gamma_j\overline{R}\left(M^{-1}A\right)^j\mathbf{r}^0 = \mathbf{v}^0 + \sum_{j=0}^{k-1}\gamma_j\left(\overline{R}M^{-1}A\overline{P}\right)^j\overline{R}\mathbf{r}^0\\ &= \mathbf{v}^0 + \sum_{j=0}^{k-1}\gamma_j\left(\overline{R}M^{-1}A\overline{P}\right)^j\bar{\mathbf{r}}^0 \in \mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \bar{\mathbf{r}}^0),\end{aligned}$$

and thus $\overline{R}\left(\mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0)\right) \subset \mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \bar{\mathbf{r}}^0)$. Similarly if $\mathbf{w} \in \mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \bar{\mathbf{r}}^0)$ then

$$\begin{aligned}\mathbf{w} &= \mathbf{v}^0 + \sum_{j=0}^{k-1}\gamma_j\left(\overline{R}M^{-1}A\overline{P}\right)^k\bar{\mathbf{r}}^0 = \overline{R}\mathbf{u}^0 + \sum_{j=0}^{k-1}\gamma_j\left(\overline{R}M^{-1}A\overline{P}\right)^j\overline{R}\mathbf{r}^0\\ &= \overline{R}\mathbf{u}^0 + \sum_{j=0}^{k-1}\gamma_j\overline{R}\left(M^{-1}A\right)^j\mathbf{r}^0,\end{aligned}$$

thus $\mathbf{w} \in \overline{R}\left(\mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0)\right)$ and we achieve the desired relation (18).

Theorem 2 shows that the restriction to the substructure of the affine volume Krylov space of RAS coincides with the affine substructured Krylov space of SRAS. One could then wonder if the restrictions of the iterates of GMRES applied to the preconditioned volume system (13) coincide with the iterates of GMRES applied to the preconditioned substructured system (15). However, this does not turn out to be true. Nevertheless, we can further link the action of GMRES on these two preconditioned systems.

It is well known, (see e.g [35, Section 6.5.1]), that GMRES applied to (13) and (15) generates a sequence of iterates $\left\{\mathbf{u}^k\right\}_k$ and $\left\{\mathbf{v}^k\right\}_k$ such that

$$\mathbf{u}^k = \text{argmin}_{\widetilde{\mathbf{u}}^k \in \mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0)}\|M^{-1}\mathbf{f} - M^{-1}A\widetilde{\mathbf{u}}^k\|_2, \tag{19}$$

and

$$\mathbf{v}^k = \text{argmin}_{\widetilde{\mathbf{v}}^k \in \mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \bar{\mathbf{r}}^0)}\|\overline{R}M^{-1}\mathbf{f} - \overline{R}M^{-1}A\overline{P}\widetilde{\mathbf{v}}^k\|_2. \tag{20}$$

The iterates $\mathbf{u}^k$ and $\mathbf{v}^k$ can be characterized using orthogonal and Hessenberg matrices obtained with the Arnoldi iteration. In particular, the k-th iteration of Arnoldi provides orthogonal matrices Q_k, Q_{k+1} and a Hessenberg matrix H_k such that $M^{-1}AQ_k = Q_{k+1}H_k$, and the columns of Q_k form an orthonormal basis for the Krylov subspace $\mathcal{K}_k(M^{-1}A, \mathbf{r}^0)$. Using these matrices, one writes $\mathbf{u}^k$ as $\mathbf{u}^k = \mathbf{u}^0 + Q_k\mathbf{a}$, where $\mathbf{a} \in \mathbb{R}^k$ is the solution of the least squares problem

$$\mathbf{a} = \text{argmin}_{\widetilde{\mathbf{a}} \in \mathbb{R}^k}\|Q_k(\|\mathbf{r}^0\|_2\mathbf{e}_1 - H_k\widetilde{\mathbf{a}})\|_2 = \text{argmin}_{\widetilde{\mathbf{a}} \in \mathbb{R}^k}\|\|\mathbf{r}^0\|_2\mathbf{e}_1 - H_k\widetilde{\mathbf{a}}\|_2, \tag{21}$$

and $\mathbf{e}_1$ is the canonical vector of $\mathbb{R}^{k+1}$. Similarly, one characterizes the vector $\mathbf{v}^k$ as $\mathbf{v}^k = \mathbf{v}^0 + \overline{Q}_k\mathbf{y}$ such that

$$\mathbf{y} = \text{argmin}_{\widetilde{\mathbf{y}}\in\mathbb{R}^k} \|\|\bar{\mathbf{r}}^0\|_2\mathbf{e}_1 - \overline{H}_k\widetilde{\mathbf{y}}\|_2, \tag{22}$$

where $\overline{Q}_k, \overline{H}_k$ are the orthogonal and Hessenberg matrices obtained through the Arnoldi method applied to the matrix $\overline{R}M^{-1}A\overline{P}$.

The next theorem provides a link between the volume least square problem (19) and the substructured one (20).

Theorem 3 *Under the hypothesis of Theorem 2, the k-th iterate of GMRES applied to* (15) *is equal to* $\mathbf{v}^k = \mathbf{v}^0 + \overline{Q}_k\mathbf{y} = \mathbf{v}^0 + \overline{R}Q_k\mathbf{t}$*, where* $\mathbf{y}$ *satisfies* (22) *while*

$$\mathbf{t} := \textit{argmin}_{\widetilde{\mathbf{t}}\in\mathbb{R}^k} \|\overline{R}Q_{k+1}(\|\mathbf{r}^0\|_2\mathbf{e}_1 - H_k\widetilde{\mathbf{t}})\|_2. \tag{23}$$

Proof It is clear that $\mathbf{v}^k = \mathbf{v}^0 + \overline{Q}_k\mathbf{y} = \mathbf{v}^0 + \overline{R}Q_k\mathbf{t}$ as the first equality follows from standard GMRES literature (see e.g [35, Section 6.5.1]). The second equality follows from Theorem (2) as we have shown that $\mathbf{v}^0 + \mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \bar{\mathbf{r}}^0) = \overline{R}(\mathbf{u}^0 + \mathcal{K}_k(M^{-1}A, \mathbf{r}^0))$. Thus the columns of $\overline{R}Q_k$ form an orthonormal basis of $\mathcal{K}_k(\overline{R}M^{-1}A\overline{P}, \bar{\mathbf{r}}^0)$ and hence, $\mathbf{v}^k$ can be expressed as a linear combination of the columns of $\overline{R}Q_k$ with coefficients in the vector $\mathbf{t} \in \mathbb{R}^k$ plus $\mathbf{v}^0$. We are then left to show (23). We have

$$\begin{aligned}
\min_{\widetilde{\mathbf{v}}^k\in\mathbf{v}^0+\mathcal{K}_k(\overline{R}M^{-1}A\overline{P},\bar{\mathbf{r}}^0)} & \| \overline{R}M^{-1}\mathbf{f} - \overline{R}M^{-1}A\overline{P}\widetilde{\mathbf{v}}^k\|_2 \\
&= \min_{\widetilde{\mathbf{t}}\in\mathbb{R}^k} \| \overline{R}M^{-1}\mathbf{f} - \overline{R}M^{-1}A\overline{P}(\mathbf{v}^0 + \overline{Q}_k\widetilde{\mathbf{t}})\|_2 \\
&= \min_{\widetilde{\mathbf{t}}\in\mathbb{R}^k} \| \overline{R}M^{-1}\mathbf{f} - \overline{R}M^{-1}A\overline{P}\overline{R}\mathbf{u}^0 - \overline{R}M^{-1}A\overline{P}\overline{Q}_k\widetilde{\mathbf{t}})\|_2.
\end{aligned}$$

Using the relation $\text{Im}(\overline{R}Q_k) = \text{Im}(\overline{Q}_k)$, Lemma (1), the Arnoldi relation $M^{-1}AQ_k = Q_{k+1}H_k$ and that $\mathbf{r}^0$ coincides with the first column of Q_k except for a normalization constant, we conclude

$$\begin{aligned}
&\min_{\widetilde{\mathbf{t}}\in\mathbb{R}^k} \| \overline{R}M^{-1}\mathbf{f} - \overline{R}M^{-1}A\overline{P}\overline{R}\mathbf{u}^0 - \overline{R}M^{-1}A\overline{P}\overline{R}Q_k\widetilde{\mathbf{t}})\|_2 \\
&= \min_{\widetilde{\mathbf{t}}\in\mathbb{R}^k} \| \overline{R}\mathbf{r}^0 - \overline{R}M^{-1}AQ_k\widetilde{\mathbf{t}})\|_2 \\
&= \min_{\widetilde{\mathbf{t}}\in\mathbb{R}^k} \| \overline{R}Q_{k+1}(\|\mathbf{r}^0\|_2\mathbf{e}_1 - H_k\widetilde{\mathbf{t}})\|_2,
\end{aligned} \tag{24}$$

and this completes the proof.

Few comments are in order here. First, GMRES applied to (15) converges in maximum $\overline{N}$ iterations as the preconditioned matrix $\overline{R}M^{-1}A\overline{P}$ has size $\overline{N}\times\overline{N}$. Second, Theorem (2) states that $\overline{R}(\mathbf{u}^0 + \mathcal{K}_{\overline{N}}(M^{-1}A, \mathbf{r}^0))$ already contains the exact substructured solution, that is the exact substructured solution lies in the restriction of the volume Krylov space after $\overline{N}$ iterations. Theoretically, if one could get the exact substructured solution from $\overline{R}(\mathbf{u}_0 + \mathcal{K}_{\overline{N}}(M^{-1}A, \mathbf{r}^0))$, then $\overline{N}$ iterations of GMRES applied to (13), plus an harmonic extension of

the substructured data into the subdomains, would be sufficient to get the exact volume solution.

On the other hand, we can say a bit more analyzing the structure of $M^{-1}A$. Using the splitting $A = M - N$, we have $M^{-1}A = I - M^{-1}N$. A direct calculation states $\mathcal{K}_k(M^{-1}A, \mathbf{r}^0) = \mathcal{K}_k(M^{-1}N, \mathbf{r}^0)$ by using the relation

$$(M^{-1}A)^k = (I - M^{-1}N)^k = \sum_{j=0}^{k} \binom{k}{j} (-1)^j (M^{-1}N)^j \quad \forall k \geq 1,$$

that is the Krylov space generated by $M^{-1}A$ is equal to the Krylov space generated by the RAS iteration matrix for error equation. We denote this linear operator with G_0^{RAS} which is defined as in (7) with $\mathbf{f} = 0$. We now consider the orthogonal complement $\widehat{V}^{\perp} := (\mathrm{span}\,\{\mathbf{e}_k\}_{k\in\mathcal{K}})^{\perp} = \mathrm{span}\,\{\mathbf{e}_i\}_{i\in\mathcal{K}^c}$, and $\dim(\widehat{V}^{\perp}) = N_v - \overline{N}$. Since for every $\mathbf{v} \in \widehat{V}^{\perp}$, it holds that $R_j A(I - P_j R_j)\mathbf{v} = 0$, we can conclude that $\widehat{V}^{\perp} \subset \ker\left(G_0^{\mathrm{RAS}}\right)$.

Using the rank-nullity theorem, we obtain

$$\dim\big(\mathrm{Im}(G_0^{\mathrm{RAS}})\big) + \dim\big(\mathrm{Ker}\left(G_0^{\mathrm{RAS}}\right)\big) = N_v \implies \dim\big(\mathrm{Im}\left(G_0^{\mathrm{RAS}}\right)\big) \leq \overline{N},$$

hence GMRES applied to the preconditioned volume system encounters a lucky Arnoldi breakdown after at most $\overline{N} + 1$ iterations (in exact arithmetic). This rank argument can be used for the substructured preconditioned system as well. Indeed as $\overline{R}M^{-1}A\overline{P} = \overline{I} - \overline{R}M^{-1}N\overline{P}$, the substructured Krylov space is generated by the matrix $\overline{R}M^{-1}N\overline{P}$, whose rank is equal to the rank of $M^{-1}N$, that is the rank of G_0^{RAS}.

Heuristically, choosing a zero initial guess, $\mathbf{r}^0 := M^{-1}\mathbf{f}$ corresponds to a solution of subdomains problem with the correct right hand side, but with zero Dirichlet boundary conditions along the interfaces of each subdomain. Thus, GMRES applied to (13) needs only to find the correct boundary conditions for each subdomain, and this can be achieved in at most $\overline{N}$ iterations as Theorem 2 shows.

Finally, we remark that each GMRES iteration on (15) is computationally less expensive than a GMRES iteration on (13) as the orthogonalization of the Arnoldi method is carried out in a much smaller space. From the memory point of view, this implies that GMRES needs to store shorter vectors. Thus, a saturation of the memory is less likely, and restarted versions of GMRES may be avoided.

4 The nonlinear case

In this section, we study iterative and preconditioned domain decomposition methods to solve the nonlinear system (3).

357 4.1 Nonlinear iterative methods

358 RAS can be generalized to solve the nonlinear equation (3). To show this, we
359 introduce the solution operators G_j which are defined through

$$R_j F(P_j G_j(\mathbf{u}) + (I - P_j R_j)\mathbf{u}) = 0, \tag{25}$$

360 where the operators R_j and P_j are defined in Section 2. Nonlinear RAS for N
361 subdomains then reads

$$\mathbf{u}^n = \sum_{j\in\mathcal{J}} \widetilde{P}_j G_j(\mathbf{u}^{n-1}). \tag{26}$$

It is possible to show that (26) reduces to (7) if $F(\mathbf{u})$ is a linear function: assuming that $F(\mathbf{u}) = A\mathbf{u} - \mathbf{f}$, equation (25) becomes

$$\begin{aligned} &R_j F\left(P_j G_j\left(\mathbf{u}^{n-1}\right) + \left(I - P_j R_j\right)\mathbf{u}^{n-1}\right) = R_j\left(A\left(P_j G_j\left(\mathbf{u}^{n-1}\right) + \left(I - P_j R_j\right)\mathbf{u}^{n-1}\right) - \mathbf{f}\right) \\ &= A_j G_j\left(\mathbf{u}^{n-1}\right) + R_j\left(A\left(I - P_j R_j\right)\mathbf{u}^{n-1} - \mathbf{f}\right) = 0, \end{aligned}$$

362 which implies $G_j\left(\mathbf{u}^{n-1}\right) = A_j^{-1} R_j\left(\mathbf{f} - A\left(I - P_j R_j\right)\mathbf{u}^{n-1}\right)$, and thus (26)
363 reduces to (7).

364 Similarly to the linear case, we introduce the nonlinear SRAS. Defining

$$\overline{G}_j(\mathbf{v}^{n-1}) := \overline{R}\widetilde{P}_j G_j\left(\overline{P}\mathbf{v}^{n-1}\right), \tag{27}$$

365 we obtain the nonlinear substructured iteration

$$\mathbf{v}^n = \overline{R}\sum_{j\in\mathcal{J}} \widetilde{P}_j G_j(\overline{P}\mathbf{v}^{n-1}) = \sum_{j\in\mathcal{J}} \overline{G}_j(\mathbf{v}^{n-1}), \tag{28}$$

366 which is the nonlinear counterpart of (10).

367 The same calculations of Theorem 1 allow one to obtain an equivalence
368 result between nonlinear RAS and nonlinear SRAS.

369 **Theorem 4 (Equivalence between nonlinear RAS and SRAS)** *Assume*
370 *that the operators* $\overline{R}$ *and* $\overline{P}$ *satisfy* $\overline{R}\sum_{j\in\mathcal{J}} \widetilde{P}_j G_j(\mathbf{u}) = \overline{R}\sum_{j\in\mathcal{J}} \widetilde{P}_j G_j(\overline{P}\overline{R}\mathbf{u})$.
371 *Let us consider an initial guess* $\mathbf{u}^0 \in V$ *and its substructured restriction* $\mathbf{v}^0 :=$
372 $\overline{R}\mathbf{u}^0 \in \overline{V}$*, and define the sequences* $\{\mathbf{u}^n\}$, $\{\mathbf{v}^n\}$ *such that*

$$\mathbf{u}^n = \sum_{j\in\mathcal{J}} \widetilde{P}_j G_j(\mathbf{u}^{n-1}), \quad \mathbf{v}^n = \sum_{j\in\mathcal{J}} \overline{G}_j(\mathbf{v}^{n-1}).$$

373 *Then for every* $n \geq 1$, $\overline{R}\mathbf{u}^n = \mathbf{v}^n$.

4.2 Nonlinear preconditioners for Newton's method

In [13], it was proposed to use the fixed point equation of nonlinear RAS as a preconditioner for Newton's method, in a spirit that goes back to [4,3]. This method has been called RASPEN (Restricted Additive Schwarz Preconditioned Exact Newton) and it consists in applying Newton's method to the fixed point equation of nonlinear RAS, that is,

$$\mathcal{F}(\mathbf{u}) = \mathbf{u} - \sum_{j\in\mathcal{J}} \widetilde{P}_j G_j(\mathbf{u}) = 0. \tag{29}$$

For a comprehensive discussion of this method, we refer to [13]. As done in (14) for the linear case, we now introduce a substructured variant of RASPEN and we call it SRASPEN (Substructured Restricted Additive Schwarz Preconditioned Exact Newton). SRASPEN is obtained by applying Newton's method to the fixed point equation of nonlinear SRAS, that is,

$$\overline{\mathcal{F}}(\mathbf{v}) := \mathbf{v} - \sum_{j\in\mathcal{J}} \overline{G}_j(\mathbf{v}) = 0.$$

One can verify that the above equation $\overline{\mathcal{F}}(\mathbf{v}) = 0$ can also be written as

$$\overline{\mathcal{F}}(\mathbf{v}) = \overline{RP}\mathbf{v} - \sum_{j\in\mathcal{J}} \overline{R}\widetilde{P}_j G_j(\overline{P}\mathbf{v}) = \overline{R}\mathcal{F}(\overline{P}\mathbf{v}) = 0. \tag{30}$$

This formulation of SRASPEN provides its relation with RASPEN and simplifies the task of computing the Jacobian of SRASPEN.

4.2.1 Computation of the Jacobian and implementation details

To apply Newton's method, we need to compute the Jacobian of SRASPEN. Let $J_{\mathcal{F}}(\mathbf{w})$ and $J_{\overline{\mathcal{F}}}(\mathbf{w})$ denote the action of the Jacobian of RASPEN and SRASPEN on a vector $\mathbf{w}$. Since these methods are closely related, indeed $\overline{\mathcal{F}}(\mathbf{v}) = \overline{R}\mathcal{F}(\overline{P}\mathbf{v})$, we can immediately compute the Jacobian of $\overline{\mathcal{F}}$ once we have the Jacobian of $\mathcal{F}$, using the chain rule, $J_{\overline{\mathcal{F}}}(\mathbf{v}) = \overline{R}J_{\mathcal{F}}(\overline{P}\mathbf{v})\overline{P}$. The Jacobian of $\mathcal{F}$ has been derived in [13] and we report here the main steps for the sake of completeness. Differentiating equation (29) with respect to $\mathbf{u}$ leads to

$$J_{\mathcal{F}}(\mathbf{u}) := \frac{d\mathcal{F}}{d\mathbf{u}}(\mathbf{u}) = I - \sum_{j\in\mathcal{J}} \widetilde{P}_j \frac{dG_j}{d\mathbf{u}}(\mathbf{u}). \tag{31}$$

Recall that the local inverse operators $G_j : V \to V_j$ are defined in equation (25) as the solutions of $R_j F(P_j G_j(\mathbf{u}) + (I - P_j R_j)\mathbf{u}) = 0$. Differentiating this relation yields

$$\frac{dG_j}{d\mathbf{u}}(\mathbf{u}) = R_j - \left(R_j J\left(\mathbf{u}^{(j)}\right) P_j\right)^{-1} R_j J\left(\mathbf{u}^{(j)}\right), \tag{32}$$

where $\mathbf{u}^{(j)} := P_j G_j(\mathbf{u}) + (I - P_j R_j)\mathbf{u}$ is the volume solution vector in subdomain j and J is the Jacobian of the original nonlinear function F. Combining the above equations (31)-(32) and defining $\widetilde{\mathbf{u}}^{(j)} := P_j G_j(\overline{P}\mathbf{v}) + (I - P_j R_j)\overline{P}\mathbf{v}$, we get

$$J_{\mathcal{F}}(\mathbf{u}) = \left(\sum_{j \in \mathcal{J}} \widetilde{P}_j \left(R_j J \left(\mathbf{u}^{(j)} \right) P_j \right)^{-1} R_j J \left(\mathbf{u}^{(j)} \right) \right), \tag{33}$$

and

$$J_{\overline{\mathcal{F}}}(\mathbf{v}) = \overline{R} \left(\sum_{j \in \mathcal{J}} \widetilde{P}_j \left(R_j J \left(\widetilde{\mathbf{u}}^{(j)} \right) P_j \right)^{-1} R_j J(\widetilde{\mathbf{u}}^{(j)}) \right) \overline{P}, \tag{34}$$

where we used the assumptions $\sum_{j \in \mathcal{J}} \widetilde{P}_j R_j = I$ and $\overline{RP} = \overline{I}$. We remark that to assemble $J_{\mathcal{F}}(\mathbf{u})$ or to compute its action on a given vector, one needs to calculate $J\left(\mathbf{u}^{(j)}\right)$, that is, evaluate the Jacobian of the original nonlinear function F on the subdomain solutions $\mathbf{u}^{(j)}$. The subdomain solutions $\mathbf{u}^{(j)}$ are obtained evaluating $\mathcal{F}(\mathbf{u})$, that is performing one step of RAS with initial guess equal to $\mathbf{u}$. A smart implementation can use the local Jacobian matrices $R_j J\left(\mathbf{u}^{(j)}\right) P_j$ that are already computed by the inner Newton solvers while solving the nonlinear problem on each subdomain, and hence no extra cost is required to assemble this term. Further, the matrices $R_j J\left(\mathbf{u}^{(j)}\right)$ are different from the local Jacobian matrices at very few columns corresponding to the degrees of freedom on the interfaces and thus it suffices to only modify those specific entries. In a non-optimized implementation, one can also directly evaluate the Jacobian of F on the subdomain solutions $\mathbf{u}^{(j)}$, without relying on already computed quantities. Concerning $J_{\overline{\mathcal{F}}}(\mathbf{v})$, we emphasize that $\widetilde{\mathbf{u}}^{(j)}$ is the volume subdomain solution obtained by substructured RAS starting from a substructured function $\mathbf{v}$. Thus, like $\mathbf{u}^{(j)}$, $\widetilde{\mathbf{u}}^{(j)}$ is readily available in Newton iteration after evaluating the function $\overline{\mathcal{F}}$.

From the computational point of view, SRASPEN has several advantages over RASPEN. From (33) and (34), we note that $J_{\overline{\mathcal{F}}}$ is a matrix of dimension $\overline{N} \times \overline{N}$ where $\overline{N}$ is the number of unknowns on S, and thus is a much smaller matrix than $J_{\mathcal{F}}$, whose size is $N_v \times N_v$, with N_v the number of unknowns in volume. On the one hand, if one prefers to assemble the Jacobian matrix, either because one wants to use a direct solver or because one wants to recycle the Jacobian for several iterations, then SRASPEN dramatically reduces the cost of the assembly of the Jacobian matrix. On the other hand, we remark that (33) and (34) have the same structure of the volume and substructured preconditioned matrices (13) and (15), by just identifying $M^{-1} = \sum_{j \in \mathcal{J}} \widetilde{P}_j \left(R_j J \left(\mathbf{u}^{(j)} \right) P_j \right)^{-1}$. Similarly to Remark 1 in the linear case, we can have a fully substructured formulation, by writing

$$J_{\overline{\mathcal{F}}}(\mathbf{v}) = \sum_{j \in \mathcal{J}} \overline{P}_j \left(R_j J \left(\widetilde{\mathbf{u}}^{(j)} \right) P_j \right)^{-1} \overline{R}_j,$$

where $\overline{P}_j := \overline{R}\widetilde{P}_j$ and $\overline{R}_j := R_j J(\widetilde{u}^j)\overline{P}$. It follows that if one prefers to use a Krylov method such as GMRES, then according to the discussion in Section

3.2, SRASPEN better exploits the properties of the underlying domain decomposition method, and saves computational time by permitting to perform the orthogonalization in a much smaller space. Further implementation details and a more extensive comparison are available in the numerical section 6.

4.2.2 Convergence analysis of RASPEN and SRASPEN

Theorem 4 gives an equivalence between nonlinear RAS and nonlinear SRAS. Are RASPEN and SRASPEN equivalent? Does Newton's method behave differently if applied to the volume or to the substructured fixed point equation, like it happens with GMRES (see Section 3.2)? In this section, we aim to answer these questions by discussing the convergence properties of the exact Newton's method applied to $\mathcal{F}$ and $\overline{\mathcal{F}}$.

Let us recall that, given two approximations $\mathbf{u}^0$ and $\mathbf{v}^0$, the exact Newton's method computes for $n \geq 1$,

$$\mathbf{u}^n = \mathbf{u}^{n-1} - \left(J_{\mathcal{F}}\left(\mathbf{u}^{n-1}\right)\right)^{-1} \mathcal{F}\left(\mathbf{u}^{n-1}\right) \quad \text{and} \quad \mathbf{v}^n = \mathbf{v}^{n-1} - \left(J_{\overline{\mathcal{F}}}\left(\mathbf{v}^{n-1}\right)\right)^{-1} \overline{\mathcal{F}}(\mathbf{v}^{n-1}),$$

where $J_{\mathcal{F}}\left(\mathbf{u}^{n-1}\right)$ and $J_{\overline{\mathcal{F}}}\left(\mathbf{v}^{n-1}\right)$ are the Jacobian matrices respectively of $\mathcal{F}$ and $\overline{\mathcal{F}}$ evaluated at $\mathbf{u}^{n-1}$ and $\mathbf{v}^{n-1}$. In this paragraph, we do not need a precise expression for $J_{\mathcal{F}}$ and $J_{\overline{\mathcal{F}}}$. However we recall that, the definition $\overline{\mathcal{F}}(\mathbf{v}) = \overline{R}\mathcal{F}(\overline{P}\mathbf{v})$ and the chain rule derivation provides us the relation $J_{\overline{\mathcal{F}}}(\mathbf{v}) = \overline{R}J_{\mathcal{F}}(\overline{P}\mathbf{v})\overline{P}$. If the operators $\overline{R}$ and $\overline{P}$ were square matrices, we would immediately obtain that RASPEN and SRASPEN are equivalent, due to the affine invariance theory for Newton's method [12]. However, in our case, $\overline{R}$ and $\overline{P}$ are rectangular matrices and they map between spaces of different dimensions. Nevertheless, in the following theorem, we show that RASPEN and SRASPEN provide the same iterates restricted to the interfaces under further assumptions on $\overline{R}$ and $\overline{P}$, which is a direct generalization of (8) to the nonlinear case.

Theorem 5 (Equivalence between RASPEN and SRASPEN) *Assume that the operators $\overline{R}$ and $\overline{P}$ satisfy*

$$\overline{R}\mathcal{F}(\mathbf{u}) = \overline{R}\mathcal{F}(\overline{P}\overline{R}\mathbf{u}) = \overline{\mathcal{F}}(\overline{R}\mathbf{u}). \tag{35}$$

Given an initial guess $\mathbf{u}^0 \in V$ and its substructured restriction $\mathbf{v}^0 := \overline{R}\mathbf{u}^0 \in \overline{V}$, define the sequences $\{\mathbf{u}^n\}$ and $\{\mathbf{v}^n\}$ such that

$$\mathbf{u}^n = \mathbf{u}^{n-1} - \left(J_{\mathcal{F}}\left(\mathbf{u}^{n-1}\right)\right)^{-1} \mathcal{F}\left(\mathbf{u}^{n-1}\right) \quad \text{and} \quad \mathbf{v}^n = \mathbf{v}^{n-1} - \left(J_{\overline{\mathcal{F}}}\left(\mathbf{v}^{n-1}\right)\right)^{-1} \overline{\mathcal{F}}\left(\mathbf{v}^{n-1}\right).$$

Then for every $n \geq 1$, $\overline{R}\mathbf{u}^n = \mathbf{v}^n$.

Proof We first prove the equality $\overline{R}\mathbf{u}^1 = \mathbf{v}^1$ by direct calculations. Taking the restriction of the RASPEN iteration, we obtain

$$\overline{R}\mathbf{u}^1 = \overline{R}\mathbf{u}^0 - \overline{R}\left(J_{\mathcal{F}}\left(\mathbf{u}^0\right)\right)^{-1} \mathcal{F}\left(\mathbf{u}^0\right) = \mathbf{v}^0 - \overline{R}\left(J_{\mathcal{F}}\left(\mathbf{u}^0\right)\right)^{-1} \mathcal{F}\left(\mathbf{u}^0\right). \tag{36}$$

Now, due to the definition of $\overline{\mathcal{F}}$ and of $\mathbf{v}^0$, and to the assumption (35), we have

$$\overline{\mathcal{F}}\left(\mathbf{v}^0\right) = \overline{R}\mathcal{F}\left(\overline{P}\mathbf{v}^0\right) = \overline{R}\mathcal{F}\left(\overline{PR}\mathbf{u}^0\right) = \overline{R}\mathcal{F}\left(\mathbf{u}^0\right). \tag{37}$$

Further, taking the Jacobian of assumption (35), we have $\overline{R}J_{\mathcal{F}}(\mathbf{u}^0) = J_{\overline{\mathcal{F}}}(\overline{R}\mathbf{u}^0)\overline{R}$, which simplifies by taking the inverse of the Jacobians to

$$\overline{R}\left(J_{\mathcal{F}}(\mathbf{u}^0)\right)^{-1} = \left(J_{\overline{\mathcal{F}}}(\overline{R}\mathbf{u}^0)\right)^{-1}\overline{R}. \tag{38}$$

Finally substituting relations (37) and (38) into (36) leads to

$$\begin{aligned}\overline{R}\mathbf{u}^1 &= \mathbf{v}^0 - \overline{R}\left(J_{\mathcal{F}}\left(\mathbf{u}^0\right)\right)^{-1}\mathcal{F}\left(\mathbf{u}^0\right) = \mathbf{v}^0 - \left(J_{\overline{\mathcal{F}}}(\overline{R}\mathbf{u}^0)\right)^{-1}\overline{R}\mathcal{F}\left(\mathbf{u}^0\right)\\ &= \mathbf{v}^0 - \left(J_{\overline{\mathcal{F}}}(\mathbf{v}^0)\right)^{-1}\overline{\mathcal{F}}\left(\mathbf{v}^0\right) = \mathbf{v}^1,\end{aligned}$$

and the general case is obtained by induction.

5 Two-level nonlinear methods

RAS and SRAS can be generalized to two-level iterative schemes. This has already been treated in detail for the linear case in [10,11]. In this section, we introduce two-level variants for nonlinear RAS and SRAS, and also for the associated RASPEN and SRASPEN.

5.1 Two-Level iterative methods

To define a two-level method, we introduce a coarse space $V_0 \subset V$, a restriction operator $R_0 : V \to V_0$ and an interpolation operator $P_0 : V_0 \to V$. The nonlinear system F can be projected onto the coarse space V_0, defining the coarse nonlinear function $F_0\left(\mathbf{u}_0\right) := R_0 F\left(P_0\mathbf{u}_0\right)$, for every $\mathbf{u}_0 \in V_0$. Due to this definition, it follows immediately that $J_{F_0}\left(\mathbf{u}_0\right) = R_0 J_F\left(P_0\mathbf{u}_0\right)P_0$, $\forall \mathbf{u}_0 \in V_0$. To compute a coarse correction we rely on the FAS approach [1]. Given a current approximation $\mathbf{u}$, the coarse correction $C_0(\mathbf{u})$ is computed as the solution of

$$F_0(C_0(\mathbf{u}) + R_0\mathbf{u}) = F_0(R_0\mathbf{u}) - R_0F(\mathbf{u}). \tag{39}$$

Two-level nonlinear RAS is described by Algorithm 1 and it consists of a coarse correction followed by one iteration of nonlinear RAS (see [22] for different approaches).

We now focus on its substructured counterpart. We introduce a coarse substructured space $\overline{V}_0 \subset \overline{V}$, a restriction operator $\overline{R}_0 : \overline{V} \to \overline{V}_0$ and a prolongation operator $\overline{P}_0 : \overline{V}_0 \to \overline{V}$. We define the coarse substructured function as

$$\overline{\mathcal{F}}_0(\mathbf{v}_0) := \overline{R}_0\overline{\mathcal{F}}(\overline{P}_0(\mathbf{v}_0)), \quad \forall \mathbf{v}_0 \in \overline{V}_0. \tag{40}$$

From the definition it follows that $J_{\overline{\mathcal{F}}_0}(\mathbf{v}_0) = \overline{R}_0 J_{\overline{\mathcal{F}}}(\overline{P}_0\mathbf{v}_0)\overline{P}_0$, $\forall \mathbf{v}_0 \in \overline{V}_0$. There is a profound difference between two-level nonlinear RAS and two-level

Algorithm 1 Two-level nonlinear RAS

1: Solve the coarse problem $F_0(\mathbf{y}) = F_0\left(R_0\mathbf{u}^k\right) - R_0F\left(\mathbf{u}^k\right)$ and set $C_0\left(\mathbf{u}^k\right) = \mathbf{y} - R_0\mathbf{u}^k$.
2: Add the coarse correction to the current iterate, $\mathbf{u}^{k+\frac{1}{2}} = \mathbf{u}^k + P_0C_0\left(\mathbf{u}^k\right)$.
3: Compute one step of nonlinear RAS, $\mathbf{u}^{k+1} = \sum_{j\in\mathcal{J}} \widetilde{P}_jG_j\left(\mathbf{u}^{k+\frac{1}{2}}\right)$.
4: Repeat steps 1 to 3 until convergence.

Algorithm 2 Two-level iterative nonlinear SRAS

1: Solve the coarse problem $\overline{\mathcal{F}}_0(\mathbf{y}) = \overline{\mathcal{F}}_0\left(\overline{R}_0\mathbf{v}^k\right) - \overline{R}_0\overline{\mathcal{F}}\left(\mathbf{v}^k\right)$ and set $C_0^S\left(\mathbf{v}^k\right) = \mathbf{y} - \overline{R}_0\mathbf{v}^k$.
2: Add the coarse correction to the current iterate, $\mathbf{v}^{k+\frac{1}{2}} = \mathbf{v}^k + \overline{P}_0C_0^S\left(\mathbf{v}^k\right)$.
3: Compute one-step of nonlinear SRAS, $\mathbf{v}^{k+1} = \sum_{j\in\mathcal{J}} \overline{G}_j\left(\mathbf{v}^{k+\frac{1}{2}}\right)$.
4: Repeat steps 1 to 3 until convergence.

nonlinear SRAS: in the first one (Algorithm 1), the coarse function is obtained restricting the original nonlinear system $F(\mathbf{u}) = 0$ onto a coarse mesh. In the substructured version, the coarse substructured function is defined restricting the fixed point equation of nonlinear SRAS to $\overline{V}_0$. That is, the coarse substructured function corresponds to a coarse version of SRASPEN. Hence, we remark that this algorithm is the nonlinear counterpart of the linear two-level algorithm described in [10,11]. Two-level nonlinear SRAS is then defined in Algorithm 2. As in the linear case, numerical experiments will show that two-level iterative nonlinear SRAS exhibits faster convergence in terms of iteration counts compared to two-level nonlinear RAS. However, we remark that evaluating F_0 is rather cheap, while evaluating $\overline{\mathcal{F}}_0$ could be quite expensive as it requires to perform subdomain solves on the fine mesh. One possible improvement is to approximate $\overline{\mathcal{F}}_0$ replacing $\overline{\mathcal{F}}$ in its definition with another function which performs subdomain solves on a coarse mesh. Further, we emphasize that a prerequisite of any domain decomposition method is that the subdomain solves are cheap to compute in a high performance parallel implementation, so that in such a setting evaluating $\overline{\mathcal{F}}_0$ needs to be cheap as well.

5.2 Two-level preconditioners for Newton's method

Once we have defined the two-level iterative methods, we are ready to introduce the two-level versions of RASPEN and SRASPEN. The fixed point equation of two-level nonlinear RAS is

$$\begin{aligned}\mathcal{F}_{2L}(\mathbf{u}) &:= \mathbf{u} - \sum_{j\in\mathcal{J}} \widetilde{P}_jG_j(\mathbf{u} + P_0C_0(\mathbf{u})) \\ &= -P_0C_0(\mathbf{u}) - \sum_{j\in\mathcal{J}} \widetilde{P}_jC_j(\mathbf{u} + P_0C_0(\mathbf{u})) = 0,\end{aligned} \tag{41}$$

where we have introduced the correction operators $C_j(\mathbf{u}) := G_j(\mathbf{u}) - R_j\mathbf{u}$. Thus two-level RASPEN defined in [13] consists in applying Newton's method to the fixed point equation (41).

Similarly, the fixed point equation of two-level nonlinear SRAS is

$$\overline{\mathcal{F}}_{2L}(\mathbf{v}) := \mathbf{v} - \sum_{j\in\mathcal{J}} \overline{G}_j\left(\mathbf{v} + \overline{P}_0\overline{C}_0(\mathbf{v})\right) = -\overline{P}_0\overline{C}_0(\mathbf{v}) - \sum_{j\in\mathcal{J}} \overline{C}_j\left(\mathbf{v} + \overline{P}_0\overline{C}_0(\mathbf{v})\right) = 0, \tag{42}$$

where the correction operators $\overline{C}_j$ are defined as $\overline{C}_j(\mathbf{v}) := \overline{G}_j(\mathbf{v}) - \overline{R}\widetilde{P}_j R_j \overline{P}\mathbf{v}$. Two-level SRASPEN consists in applying Newton's method to the fixed point equation (42).

6 Numerical results

We discuss three different examples in this section to illustrate our theoretical results. In the first example, we consider a linear problem where we study the performance of GMRES when it is applied to the preconditioned volume system and the preconditioned substructured system. In the next two examples, we present numerical results in order to compare Newton's method, NKRAS [5], nonlinear RAS, nonlinear SRAS, RASPEN, and SRASPEN for the solution of a one-dimensional Forchheimer equation and for a two-dimensional nonlinear diffusion equation.

6.1 Linear example

We consider the diffusion equation $-\Delta u = f$, with source term $f \equiv 1$ and homogeneous boundary conditions inside the unit cube $\Omega := (0,1)^3$ decomposed into N equally-sized bricks with overlap, each discretized with 27000 degrees of freedom. The size of the overlap is $\delta := 4 \times h$. In Table 1, we study the computational effort and memory required by GMRES when applied to the preconditioned volume system (13) (GMRES-RAS) and to the preconditioned substructured system (15) (GMRES-SRAS). We let the number of subdomains grow, while keeping their sizes constant, that is the global problem becomes larger as N increases. We report the computational times to reach a relative residual smaller than 10^{-8}, and the number of gigabytes required to store the orthogonal matrices of the Arnoldi iteration, both for the volume and substructured implementations. The subdomain solves are performed in a serial fashion, we precompute the Cholesky factorizations for the subdomain matrices A_j, and for SRAS we use the fixed point equation related to formulation (12).

Table 1 shows that GMRES applied to the preconditioned substructured system is faster in terms of computational time compared to the volume implementation. This advantage becomes more evident as the global problem becomes larger. We emphasize that GMRES required the same number of iterations to reach the tolerance for both methods in all cases considered. Thus

$N_v(N) - \overline{N}$	729000(27)-92944	1728000(64)-246456	3375000(125)-511712
GMRES-RAS	11.14	47.55	136.54
GMRES-SRAS	9.86	40.37	112.06

$N_v(N) - \overline{N}$	729000(27)-92944	1728000(64)-246456	3375000(125)-511712
GMRES-RAS	0.09	0.34	0.81
GMRES-SRAS	0.01	0.05	0.12

Table 1: On the top, time in seconds required by GMRES-RAS and GMRES-SRAS to reach a relative error smaller than 10^{-8} for increasingly larger problems. At the bottom, memory use expressed in gigabytes to store the Arnoldi orthogonal matrices in both GMRES implementations.

the faster time to solution of GMRES-SRAS is due to the smaller number of floating point operations that GMRES-SRAS has to perform, since the orthogonalization steps are performed in a much smaller space, and SRAS avoids unnecessary volume computations. Furthermore, GMRES-SRAS significantly outperforms GMRES-RAS in terms of memory requirements; in this particular case, GMRES-SRAS computes and stores orthogonal matrices which are about seven times smaller than the ones used by GMRES-RAS.

6.2 Forchheimer equation in 1D

Forchheimer equation is an extension of the Darcy equation for high flow rates, where the linear relation between the flow velocity and the gradient flow does not hold anymore. In a one dimensional domain $\Omega := (0, 1)$, the Forchheimer model is

$$\begin{aligned} &q(-\lambda(x)u(x)'))' = f(x) \quad \text{in } \Omega, \\ &u(0) = u_L \quad \text{and} \quad u(1) = u_R, \end{aligned} \tag{43}$$

where $u_L, u_R \in \mathbb{R}$, $\lambda(x)$ is a positive and bounded permeability field and $q(y) := \text{sign}(y)\frac{-1+\sqrt{1+4\gamma|y|}}{2\gamma}$, with $\gamma > 0$. To discretize (43), we use the finite volume scheme described in detail in [13]. In our numerical experiments, we set $\lambda(x) = 2 + \cos(5\pi x)$, $f(x) = 50\sin(5\pi x)e^x$, $\gamma = 1$, $u(0) = 1$ and $u(1) = e^1$. The solution field $u(x)$ and the force field $f(x)$ are shown in Fig. 2. We then study the convergence behavior of our different methods. Fig. 3 shows how the relative error decays for the different methods and for a decomposition into 20 subdomains (left panel) and 50 subdomains (right panel). The initial guess is equal to zero for all these methods. Both plots in Fig. 3 show that the convergence rate of iterative nonlinear RAS and nonlinear SRAS is the same and very slow. As expected, NKRAS with line search converges better than Newton's method and further RASPEN and SRASPEN converge in the same number of outer Newton iterations as they produce the same iterates. Moreover, it seems that the convergence of RASPEN and SRASPEN is not affected by the number of subdomains. However, these plots do not tell the whole story, as one should focus not only on the number of iterations but also

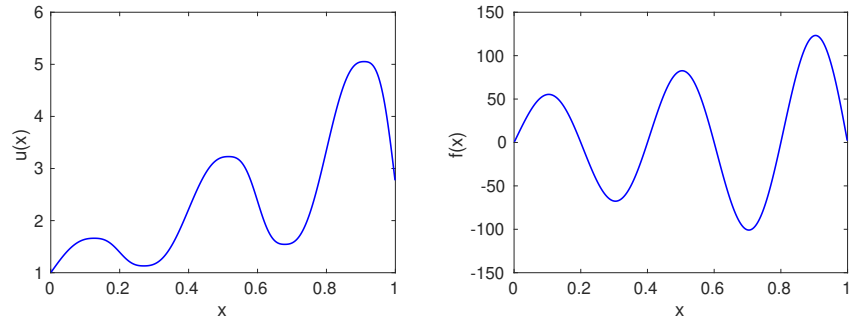

Fig. 2: Solution field $u(x)$ of Forchheimer equation (left panel) and force term $f(x)$ (right panel).

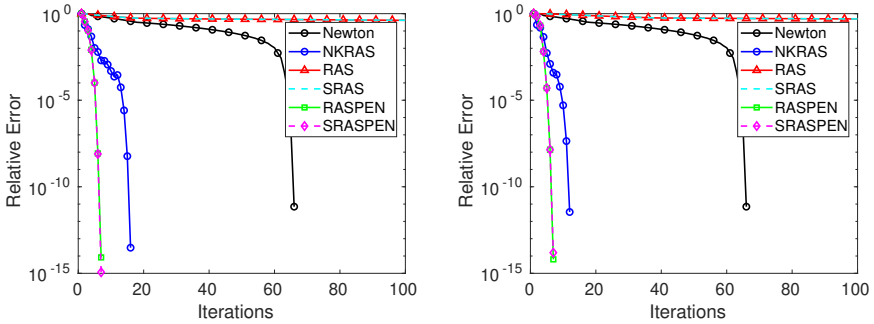

Fig. 3: Convergence behavior for Newton's method, NKRAS, nonlinear RAS, nonlinear SRAS, RASPEN and SRASPEN applied to Forchheimer equation. On the left, the simulation refers to a decomposition into 20 subdomains while on the right we consider 50 subdomains. The mesh size is $h = 10^{-3}$ and the overlap is $8h$.

on the cost of each iteration. To compare the cost of an iteration of RASPEN and SRASPEN, we have to distinguish two cases, that is, if one solves the Jacobian system directly or with some Krylov methods, e.g., GMRES. First, suppose that we want to solve the Jacobian system with a direct method and thus we need to assemble and store the Jacobians. From the expressions in equation (34) we remark that the assembly of the Jacobian of RASPEN requires $N \times N_v$ subdomain solves, where N is the number of subdomains and N_v is the number of unknowns in volume. On the other hand, the assembly of the Jacobian of SRASPEN requires $N \times \overline{N}$ solves, where $\overline{N}$ is the number of unknowns on the substructures and $\overline{N} \ll N_v$. Thus, while the assembly of $J_{\mathcal{F}}$ is prohibitive, it can still be affordable to assemble $J_{\overline{\mathcal{F}}}$. Further, the direct solution of the Jacobian system is feasible as $J_{\overline{\mathcal{F}}}$ has size $\overline{N} \times \overline{N}$. Suppose now that we solve the Jacobian systems with GMRES. Let us indicate with $I(k)$

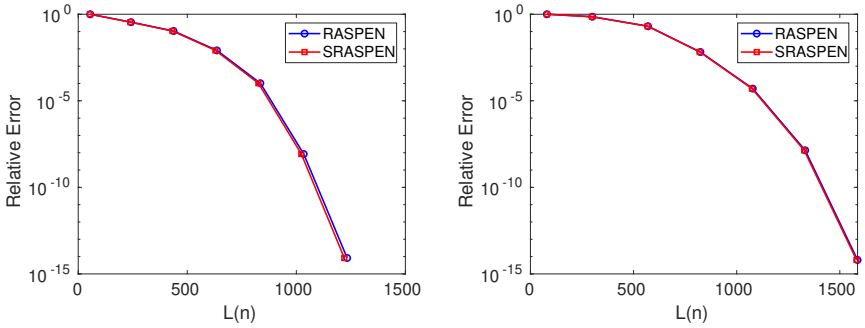

Fig. 4: Relative error decay for RASPEN and SRASPEN applied to Forchheimer equation with respect to the number of linear solves. On the left, the simulation refers to a decomposition into 20 subdomains while on the right we consider 50 subdomains. The mesh size is $h = 10^{-3}$.

and $I^S(k)$ the number of GMRES iterations to solve the volume and substructured Jacobian systems at the k-th outer Newton iteration. Each GMRES iteration requires N subdomain solves which can be performed in parallel. In our numerical experiment, we have observed that generally $I^S(k) \leq I(k)$, with $I(k) - I^S(k) \approx 0, 1, 2$, that is GMRES requires the same number of iterations or slightly less to solve the substructured Jacobian system compared to the volume one.

To better compare these two methods, we follow [13] and introduce the quantity $L(n)$ which counts the number of subdomain solves performed by these two methods till iteration n, taking into account the advantages of a parallel implementation. We set $L(n) = \sum_{k=1}^{n} L_{in}^k + I(k)$, where L_{in}^k is the maximum over the subdomains of the number of Newton iterations required to solve the local subdomain problems at iteration k. The number of linear solves performed by GMRES should be $I(k) \times N$, but as the N linear solves can be performed in parallel, the total cost of GMRES corresponds approximately to $I(k)$ linear solves. Fig. 4 shows the error decay as a function of $L(n)$. We note that the two methods require approximately the same computational cost and SRASPEN is slightly faster. For the decomposition into 50 subdomains, RASPEN requires on average 91.5 GMRES iterations per Newton iteration, while SRASPEN requires an average of 90.87 iterations. The size of the substructured space $\overline{V}$ is $\overline{N} = 98$. For the decomposition into 20 subdomains, RASPEN requires an average of 40 GMRES iterations per Newton iteration, while SRASPEN needs 38 iterations. The size of $\overline{V}$ is $\overline{N} = 38$, which means that GMRES reaches the given tolerance of 10^{-12} after exactly $\overline{N}$ steps, which is the size of the substructured Jacobian. Under these circumstances, it can be convenient to actually assemble $J_{\overline{\mathcal{F}}}$, as it requires $\overline{N} \times N$ subdomain solves which is the total cost of GMRES. Furthermore, the $\overline{N} \times N$ subdomain solves are embarrassingly parallel, while the $\overline{N} \times N$ solves of GMRES can be parallelized in the spatial direction, but not in the iterative one.

As future work, we believe it will be interesting to study the convergence of a Quasi-Newton method based on SRASPEN, where one assembles the Jacobian substructured matrix after every few outer Newton iterations, reducing the overall computational cost.

As a final remark, we specify that Fig. 4 has been obtained setting a zero initial guess for the nonlinear subdomain problems. However, at the iteration k of RASPEN one can use the subdomain restriction of the updated volume solution, that is $R_j\mathbf{u}^{k-1}$, which has been obtained by solving the volume Jacobian system at iteration $k-1$, and is thus generally a better initial guess for the next iteration. On the other hand in SRASPEN, one could use the subdomain solutions computed at iteration $k-1$, i.e. $\mathbf{u}_i^{k-1}$, as initial guess for the nonlinear subdomain problems, as the substructured Jacobian system corrects only the substructured values. Numerical experiments showed that with this particular choice of initial guess for the nonlinear subdomain problems, SRASPEN requires generally more Newton iterations to solve the local problems. In this setting, there is not a method that is constantly faster than the other as it depends on a delicate trade-off between the better GMRES performance and the need to perform more Newton iterations for the nonlinear local problems in SRASPEN.

6.3 Nonlinear Diffusion

In this subsection we consider the nonlinear diffusion problem on a square domain $\Omega := (0,1)^2$,

$$\begin{aligned} -\nabla \cdot \left(1 + u(x)^2\right) \nabla u(x) = f, \quad & \text{in } \Omega, \\ u(x) = g(x) \quad & \text{on } \partial\Omega, \end{aligned} \tag{44}$$

where the right hand side f is chosen such that $u(x) = \sin(\pi x)\sin(\pi y)$ is the exact solution. We start all these methods with an initial guess $u^0(x) = 10^5$, so that we start far away from the exact solution, and hence Newton's method exhibits a long plateau before quadratic convergence begins. Fig. 5 shows the convergence behavior for the different methods as function of the number of iterations and the number of linear solves. The average number of GMRES iterations is 8.1667 for both RASPEN and SRASPEN for the four subdomain decomposition. For a decomposition into 25 subdomains, the average number of GMRES iterations is 19.14 for RASPEN and 19.57 for SRASPEN. We remark that as the number of subdomains increases, GMRES needs more iterations to solve the Jacobian system. This is consistent with the interpretation of (34) as a Jacobian matrix $J\left(\mathbf{u}^{(j)}\right)$ preconditioned by the additive operator $\sum_{j\in\mathcal{J}} \left(R_j J\left(\mathbf{u}^{(j)}\right) P_j\right)^{-1}$; We expect this preconditioner not to be scalable since it does not involve a coarse correction. In Table 2 we compare the computational time in seconds to reach a tolerance of 10^{-8} by RASPEN and SRASPEN. SRASPEN is faster due to the less expensive GMRES iteration which is inherited by the linear analysis, see Table 1.

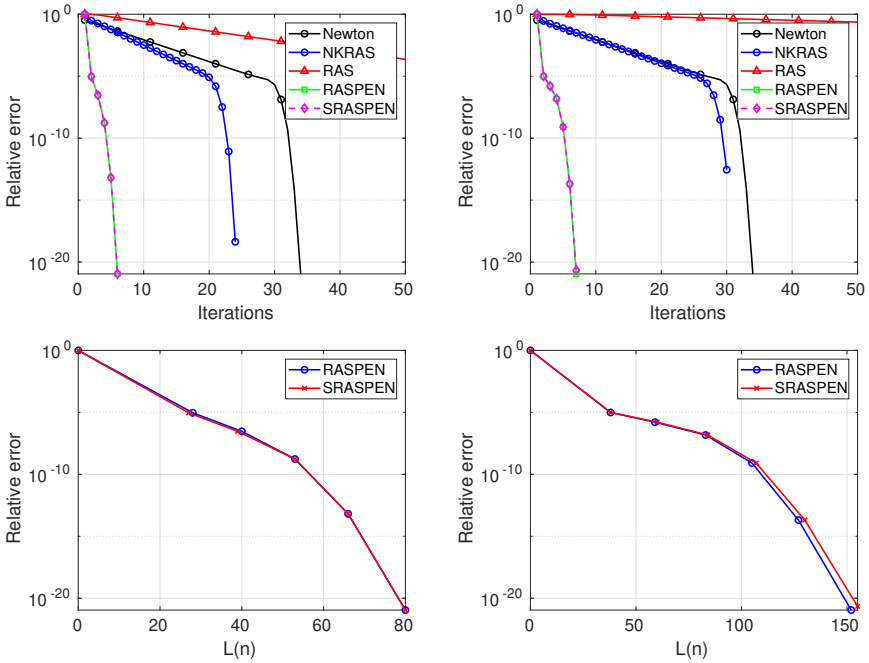

Fig. 5: Relative error decay versus the number of iterations (top row) and error decay versus the number of linear solves (bottom row). The left figures refer to a decomposition into four subdomains, the right figures to a decomposition into 25 subdomains. The mesh size is $h = 0.012$ and the overlap is $8h$.

$N_v(N) - \overline{N}$	961(4)-120	6241(25)-1200	12321(49)-2520
RASPEN	0.40	17.30	107.57
SRASPEN	0.34	15.54	98.18

Table 2: Time in seconds required by RASPEN and SRASPEN to reach a relative error smaller than 10^{-8} for the nonlinear diffusion equation with 4, 25 and 49 subdomains.

We conclude this section by showing the convergence behavior for the two-level variants of nonlinear RAS, nonlinear SRAS, RASPEN, and SRASPEN. We use a coarse grid in volume taking half of the points in x and y, and a coarse substructured grid taking half of the unknowns as depicted in Fig. 1. The interpolation and restriction operators $P_0, R_0, \overline{P}_0$ and $\overline{R}_0$ are the classical linear interpolation and fully weighting restriction operators defined in Section 5. From Fig. 6, we note that two-level nonlinear SRAS is much faster than two-level nonlinear RAS, and this observation is in agreement with the linear case treated in [10,11]. Since the two-level iterative methods are not equivalent, we also remark that two-level SRASPEN shows a better performance than two-level RASPEN in terms of iteration count. As the one-level smoother is

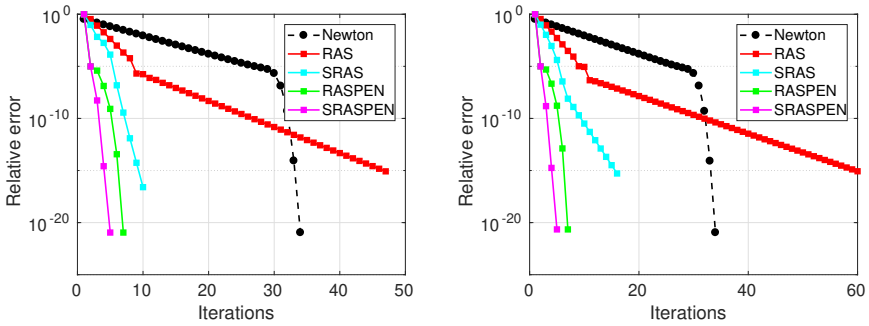

Fig. 6: Relative error decay versus the number of iterations for Newton's method, iterative two-level nonlinear RAS and SRAS, and the two-level variants of RASPEN and SRASPEN. The left figure refers to a decomposition into 4 subdomains, while the right figure refers to a decomposition into 16 subdomains. The mesh size is $h = 0.012$ and the overlap is $4h$.

the same in all methods, the better convergence of the substructured methods implies that the coarse equation involving $\overline{\mathcal{F}}_0$ provides a much better coarse correction than the classical volume one involving F_0.

Even though the two-level substructured methods are faster in terms of iteration count, the solution of the FAS problem involving $\overline{\mathcal{F}}_0 = \overline{R}_0\overline{\mathcal{F}}(\overline{P}_0(\mathbf{v}_0))$ is rather expensive as it requires to evaluate twice the substructured function $\overline{\mathcal{F}}$ (each evaluation requires subdomain solves) to compute the right hand side, to solve a Jacobian system involving $J_{\overline{\mathcal{F}}_0}$, and to evaluate $\overline{\mathcal{F}}$ on the iterates, which again require the solution of subdomain problems. Unless one has a fully parallel implementation available, the coarse correction involving $\overline{\mathcal{F}}_0$ is doomed to represent a bottleneck.

7 Conclusions

We presented an analysis of the effects of substructuring on RAS when it is applied as an iterative solver and as a preconditioner. We proved that iterative RAS and iterative SRAS converge at the same rate, both in the linear and nonlinear case. For the nonlinear case, we showed that the preconditioned methods, namely RASPEN and SRASPEN, also have the same rate of convergence as they produce the same iterates once these are restricted to the interfaces. Surprisingly, the equivalence between volume and substructured RAS breaks down when they are considered as preconditioners for Krylov methods. We showed that the Krylov spaces are equivalent, once the volume one is restricted to the substructure, however we obtained that the iterates are different by carefully deriving the least squares problems solved by GMRES. Our analysis shows that GMRES should be applied to the substructured system as it converges similarly when applied to the volume formulation, but needs

much less memory. This allows us to state that, while nonlinear RASPEN and SRASPEN produce the same iterates, SRASPEN has advantages when solving the Jacobian system, either because the use of a direct solve is feasible, or because the Krylov method can work at the substructured level. Finally, we introduced substructured two-level nonlinear SRAS and SRASPEN, and showed numerically that these methods have better convergence properties than their volume counterparts in terms of iteration count, although they are quite expensive in the present form per iteration. Future efforts will be in the direction of approximating $\overline{\mathcal{F}}_0$, by replacing the function $\overline{\mathcal{F}}$, which is defined on a fine mesh, with an approximation on a very coarse mesh, thus reducing the overall cost of the substructured coarse correction, or by using spectral coarse spaces.

8 Acknowledgements

The third author acknowledges financial support from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project-ID 258734477 - SFB 1173. The last author acknowledges Gabriele Ciaramella for several insightful discussions on domain decomposition methods.

References

1. Brandt, A., Livne, O.E.: Multigrid Techniques. Society for Industrial and Applied Mathematics (2011)
2. Cai, X.C., Dryja, M.: Domain decomposition methods for monotone nonlinear elliptic problems. Contemporary Mathematics **180** (1994)
3. Cai, X.C., Keyes, D.E.: Nonlinearly preconditioned inexact Newton algorithms. SIAM Journal on Scientific Computing **24**(1), 183–200 (2002)
4. Cai, X.C., Keyes, D.E., Young, D.P.: A nonlinear additive Schwarz preconditioned inexact Newton method for shocked duct flow. In: Proceedings of the 13th International Conference on Domain Decomposition Methods (2001)
5. Cai, X.C., Li, X.: Inexact Newton methods with restricted additive Schwarz based nonlinear elimination for problems with high local nonlinearity. SIAM Journal on Scientific Computing **33**(2), 746–762 (2011)
6. Cai, X.C., Sarkis, M.: A restricted additive Schwarz preconditioner for general sparse linear systems. SIAM Journal on Scientific Computing **21**(2), 792–797 (1999)
7. Chaouqui, F., Gander, M.J., Kumbhar, P.M., Vanzan, T.: On the nonlinear Dirichlet-Neumann method and preconditioner for Newton's method. accepted in Domain Decomposition Methods in Science and Engineering XXVI (2021)
8. Ciaramella, G., Gander, M.J.: Iterative Methods and Preconditioners for Systems of Linear Equations. SIAM (2022)
9. Ciaramella, G., Hassan, M., Stamm, B.: On the Scalability of the Schwarz Method. The SMAI journal of computational mathematics **6**, 33–68 (2020)
10. Ciaramella, G., Vanzan, T.: Substructured two-grid and multi-grid domain decomposition methods. preprint available at https://infoscience.epfl.ch/record/288182?ln=en, submitted (2021)
11. Ciaramella, G., Vanzan, T.: Spectral substructured two-level domain decomposition methods. arXiv:1908.05537v3 (submitted, 2021)
12. Deuflhard, P.: Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms. Springer Series in Computational Mathematics. Springer Berlin Heidelberg (2010)

13. Dolean, V., Gander, M.J., Kheriji, W., Kwok, F., Masson, R.: Nonlinear preconditioning: How to use a nonlinear Schwarz method to precondition Newton's method. SIAM Journal on Scientific Computing **38**(6), A3357–A3380 (2016)
14. Farhat, C., Roux, F.X.: A method of finite element tearing and interconnecting and its parallel solution algorithm. International journal for numerical methods in engineering **32**(6), 1205–1227 (1991)
15. Gander, M.J.: Optimized Schwarz methods. SIAM Journal on Numerical Analysis **44**(2), 699–731 (2006)
16. Gander, M.J.: Schwarz methods over the course of time. Electronic transactions on numerical analysis **31**, 228–255 (2008)
17. Gander, M.J.: On the origins of linear and non-linear preconditioning. In: C.O. Lee, X.C. Cai, D.E. Keyes, H.H. Kim, A. Klawonn, E.J. Park, O.B. Widlund (eds.) Domain Decomposition Methods in Science and Engineering XXIII, pp. 153–161. Springer International Publishing, Cham (2017)
18. Gander, M.J., Halpern, L.: Méthodes de décomposition de domaines – notions de base. Editions TI, Techniques de l'Ingénieur, France (2012)
19. Gong, S., Cai, X.C.: A nonlinear elimination preconditioned Newton method with applications in arterial wall simulation. In: International Conference on Domain Decomposition Methods, pp. 353–361. Springer (2017)
20. Gong, S., Cai, X.C.: A nonlinear elimination preconditioned inexact Newton method for heterogeneous hyperelasticity. SIAM Journal on Scientific Computing **41**(5), S390–S408 (2019)
21. Hackbusch, W.: Multi-Grid Methods and Applications. Springer Series in Computational Mathematics. Springer Berlin Heidelberg (2013)
22. Heinlein, A., Lanser, M.: Additive and hybrid nonlinear two-level Schwarz methods and energy minimizing coarse spaces for unstructured grids. SIAM Journal on Scientific Computing **42**(4), A2461–A2488 (2020)
23. Klawonn, A., Lanser, M., Niehoff, B., Radtke, P., Rheinbach, O.: Newton-Krylov-FETI-DP with adaptive coarse spaces. In: C.O. Lee, X.C. Cai, D.E. Keyes, H.H. Kim, A. Klawonn, E.J. Park, O.B. Widlund (eds.) Domain Decomposition Methods in Science and Engineering XXIII, pp. 197–205. Springer International Publishing, Cham (2017)
24. Klawonn, A., Lanser, M., Rheinbach, O.: Nonlinear FETI-DP and BDDC methods. SIAM Journal on Scientific Computing **36**(2), A737–A765 (2014)
25. Klawonn, A., Lanser, M., Rheinbach, O., Uran, M.: Nonlinear FETI-DP and BDDC methods: a unified framework and parallel results. SIAM Journal on Scientific Computing **39**(6), C417–C451 (2017)
26. Klawonn, A., Widlund, O.: FETI and Neumann-Neumann iterative substructuring methods: connections and new results. Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences **54**(1), 57–90 (2001)
27. Lanzkron, P.J., Rose, D.J., Wilkes, J.T.: An analysis of approximate nonlinear elimination. SIAM Journal on Scientific Computing **17**(2), 538–559 (1996)
28. Lions, P.L.: On the Schwarz alternating method. I. In: First international symposium on domain decomposition methods for partial differential equations, vol. 1, p. 42 (1988)
29. Lui, S.H.: On Schwarz alternating methods for nonlinear elliptic PDEs. SIAM Journal on Scientific Computing **21**(4), 1506–1523 (1999)
30. Lui, S.H.: On monotone iteration and Schwarz methods for nonlinear parabolic PDEs. Journal of computational and applied mathematics **161**(2), 449–468 (2003)
31. Luo, L., Cai, X.C., Yan, Z., Xu, L., Keyes, D.E.: A multilayer nonlinear elimination preconditioned inexact Newton method for steady-state incompressible flow problems in three dimensions. SIAM Journal on Scientific Computing **42**(6), B1404–B1428 (2020)
32. Mandel, J., Brezina, M.: Balancing domain decomposition for problems with large jumps in coefficients. Mathematics of Computation **65**(216), 1387–1401 (1996)
33. Przemieniecki, J.S.: Matrix structural analysis of substructures. AIAA Journal **1**(1), 138–147 (1963)
34. Quarteroni, A., Valli, A.: Domain Decomposition Methods for Partial Differential Equations. Numerical Mathematics and Scientific Computation. Oxford Science Publications (1999)

35. Saad, Y.: Iterative methods for sparse linear systems. SIAM (2003)
36. Toselli, A., Widlund, O.: Domain Decomposition Methods: Algorithms and Theory, *Series in Computational Mathematics*, vol. 34. Springer, New York (2005)
37. Trottenberg, U., Oosterlee, C., Schuller, A.: Multigrid. Elsevier Science (2000)