Asynchronous global-local non-invasive coupling for linear elliptic problems

Ahmed El Kerim, Pierre Gosselet, Frederic Magoules

Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 406 (2023) 115910
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Asynchronous global–local non-invasive coupling for linear elliptic problems

Ahmed El Kerim[a,c], Pierre Gosselet[b], Frédéric Magoulès[c,d,*]

[a] Université Paris-Saclay, CentraleSupélec, ENS Paris-Saclay, CNRS, LMPS, Gif-sur-Yvette, 91190, France
[b] Université de Lille, CNRS, Centrale Lille, UMR 9013, LaMcube, Lille, 59000, France
[c] Université Paris-Saclay, CentraleSupélec, MICS, Gif-sur-Yvette, 91190, France
[d] Faculty of Engineering and Information Technology, University of Pécs, Pécs, Hungary

Received 15 November 2022; received in revised form 17 January 2023; accepted 18 January 2023
Available online xxxx

Abstract

This paper presents the first asynchronous version of the Global/Local non-invasive coupling, capable of dealing efficiently with multiple, possibly adjacent, patches. We give a new interpretation of the coupling in terms of primal domain decomposition method, and we prove the convergence of the relaxed asynchronous iteration. The asynchronous paradigm lifts many bottlenecks of the Global/Local coupling performance. We illustrate the method on several linear elliptic problems as encountered in thermal and elasticity studies.

Keywords: Non-invasive coupling; Asynchronous domain decomposition; Linear domain decomposition method; Monotone operator; Paracontraction techniques; MPI-RDMA

1. Introduction

Engineering problems are often defined on very different scales, ranging from a coarse scale to model the whole structure to very fine scales that allow for the local details to be resolved. A method frequently used in the industry to link the scales is the submodeling [1–3]. This non-intrusive method is simple to implement but has shown limits regarding the accuracy of the results.

The non-invasive Global–Local coupling technique was first proposed and implemented in [4]. It aims at making submodeling accurate by means of iterations. It extends some previous reanalysis techniques [5–7], and it has strong connections with Schwarz domain decomposition methods [8,9] and multiscale methods [10] while preserving the non-intrusive character of submodeling. Thus, it was implemented to couple research codes and legacy commercial software like Abaqus [11], Code_Aster [12], or Z-set [13].

The philosophy is to start from a simplified global model and then allow local alterations (geometry, material, load, and mesh) to be inserted and their effect to be evaluated without heavy intervention on the initial model (see [14] for a pedagogic presentation). It was successfully applied in many contexts like the introduction of local plasticity and geometrical refinements [4], the computation of the propagation of cracks in a sound model [12],

* Corresponding author at: Université Paris-Saclay, CentraleSupélec, MICS, Gif-sur-Yvette, 91190, France.
E-mail address: frederic.magoules@hotmail.com (F. Magoulès).

https://doi.org/10.1016/j.cma.2023.115910

the evaluation of stochastic effects with deterministic computations [15,16], the taking into account of the exact geometry of connectors in an assembly of plates [17]. In [12] the method was used in order to implement a nonlinear domain decomposition method [18–21] in a non-invasive manner in Code_Aster. Extension of the approach to explicit dynamics was proposed in [22], improved in [23] and applied to the prediction of delamination under impact loading in [24].

All the above applications were developed in a synchronous framework that has been taken advantage of by accelerators (Aitken, quasi-Newton, Krylov), see [9] where the method is proved to be an implementation of an alternating Dirichlet–Robin approach where the Robin parameter corresponds to the condensation of the coarse domain covered by the patch. However, due to the alternating nature of the method, its computational performance is inherently limited, with some processors idling while others are computing. This paper aims at deriving an asynchronous version of the global–local coupling, which enables us to get rid of most waiting periods.

Asynchronous iteration was introduced in [25], under the name of chaotic relaxation, to solve large linear systems. It has subsequently been the subject of several studies, [26] generalized the method to nonlinear problems, the work in [27] allowed the first implementation of asynchronous methods on multiprocessor architectures, in [28,29] convergence results for the asynchronous iterations based on the notion of classical contraction was presented, recent work in [30] show interesting theoretical and practical results for the Richardson iterations from the asynchronous point of view.

Several works have shown that domain decomposition methods are well suited for asynchronous parallel computation, such as alternating Schwarz [31], optimized Schwarz [32–34], sub-structuring methods [35,36], primal Schur domain decomposition method [37] and also multigrid methods [38]. In [39,40], one can find a global review of asynchronous iterations from both theoretical and implementation points of view.

Our study is conducted on linear elliptic problems discretized by the finite element approach. We prove the convergence of relaxed iterations using the theory of paracontractions [41], and illustrate it on several examples of thermal and elasticity problems.

The paper is organized as follows: in Section 2 a new derivation of the method is proposed, in Section 3 the asynchronous framework is exposed and studied, illustrations are given in Section 4.

2. The non-invasive global/local coupling

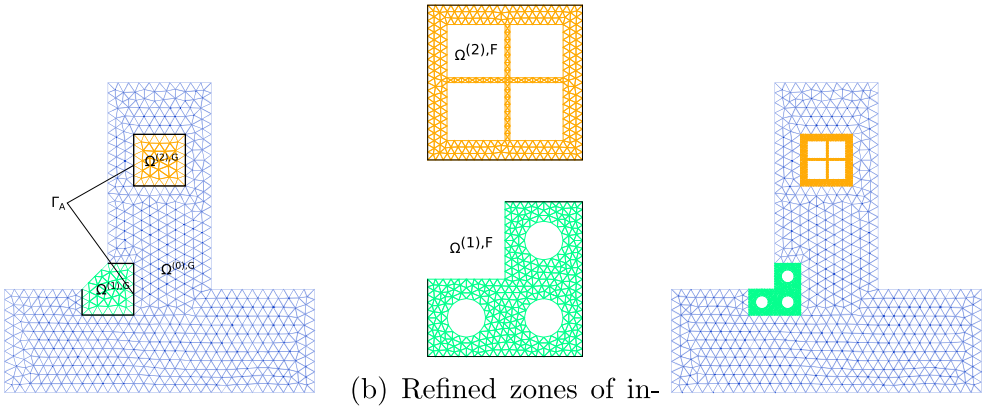
The framework chosen to develop the method is the one of linear elliptic problems. This corresponds to certain thermal or elasticity static problems. We propose to derive the method as an evolution of the submodeling technique, we also give another (original) interpretation in terms of domain decomposition method.

2.1. Principle of the method

The classical scenario is illustrated on Fig. 1. A linear Global coarse model is used to describe a large structure. After the initial computation (Fig. 2(a)), some zones of interest $\Omega^{(s),G}$ ($s > 0$) are selected because some criterion has been exceeded or because it was known from the beginning that some details were missing in the Global model. This is the case for our illustration where geometrical details and adapted meshes are introduced in the Local modeling of the zones of interest $\Omega^{(s),F}$. Material laws could also be modified by the introduction of some heterogeneity. Local computations are run in parallel on the patches using the Global solution as Dirichlet boundary condition (for $s > 0$, the interior of the Local and Global subdomains may differ, but their interface $\Gamma^{(s)}$ must be the same $\Gamma^{(s)} = \Omega \cap \partial\Omega^{(s),G} = \Omega \cap \partial\Omega^{(s),F}$).

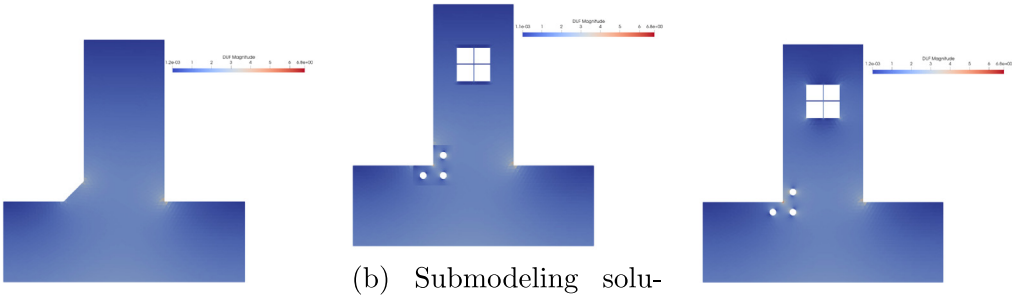
This sequence of computations corresponds to the (in)famous submodeling technique which is known to result in large errors because the effects of Local patches are not sent back to the Global model, and interactions between patches are thus impossible to be accounted for.

The error can be materialized by the lack of balance of the fluxes between the Global zone not covered by patches, denoted by $\Omega^{(0),G} = \Omega^G \setminus \left(\bigcup_{s>0} \Omega^{(s),F}\right)$ and the Local models, we note also that $\Omega^{(0),G} = \Omega^{(0),F}$, as the local representation of the complementary domain is the same as the global representation. As can be seen on Fig. 2(b), which shows the norm of the heat flux and where the Local models overwrite the Global ones. There is a discontinuity at the interface which does not exist in the Reference computation where all interactions are taken into account; see Fig. 2(c) which corresponds to a direct computation of the Reference model where the zones of interest are described with the Local models, see Fig. 1(c).



(a) Global problem (b) Refined zones of interest (c) Reference problem

Fig. 1. Models and subdomains for the Global/Local coupling.



(a) Global solution (b) Submodeling solution (c) Reference solution

Fig. 2. Comparison of the norm of the heat flux for the submodeling and reference approaches (thermal problem).

The Global/Local coupling is a simple iterative technique (a Richardson iteration for its simpler version) aiming at obtaining the Reference solution from computations carried on the Global and Local models (that is to say without the potentially cumbersome creation of the Reference model) with minimal intervention on the models and software.

2.2. Derivation of the Global/Local coupling

There exist many ways to derive the Global/Local coupling. This subsection just sets up the method, the convergence of the asynchronous iteration being the subject of the next section.

We use boldface for discrete (nodal) quantities, lower case for vectors and upper case for matrices.

2.2.1. Global problem

The Global problem is the classical finite element discretization of a coarse model of the structure, with one extra interface load. Let $\mathbf{p}_\Gamma$ denote the vector of nodal fluxes applied on the interface nodes $\Gamma = \bigcup_{s=0}^{N} \Gamma^{(s)}$. To position the interface in the Global domain we introduce the boolean trace operator $\mathbf{T}^G : \Omega^G \to \Gamma$, its transpose is the extension-by-0 operator, with : $\mathbf{u}_\Gamma^G = \mathbf{T}^G \mathbf{u}^G$.

The discrete Global problem can be written as:

$$\left|\begin{array}{l}\text{For given } \mathbf{p}_\Gamma \text{ on } \Gamma, \text{ find } \mathbf{u}^G \text{ in } \Omega^G, \text{ such that}\\ \mathbf{K}^G\mathbf{u}^G = \mathbf{f}^G + \mathbf{T}^{G^T}\mathbf{p}_\Gamma\end{array}\right. \tag{1}$$

where one can recognize the symmetric definite positive stiffness matrix $\mathbf{K}^G$, the vector of generalized loads $\mathbf{f}^G$, the vector of unknowns $\mathbf{u}^G$.

The interface load is non-standard since it is a Neumann condition applied on an immersed surface. This corresponds to imposing a flux discontinuity in the Global model. It appears that such a load can easily be applied in industrial software, and the Global solution is obtained with a classical solver.

We need to clarify the role played by Subdomain 0, which might be non-existent. It is a subdomain, sometimes called Complement domain in the Global/Local literature, where the Local and Global model coincide (same geometry $\Omega^{(0),G}$, same properties, same load, same approximation).

In order to single out the contribution of subdomains, we introduce the boolean assembly operators $\mathbf{A}^{(s)} : \Gamma^{(s),G} \rightarrow \Gamma^G$ as classically encountered in the primal domain decomposition methods, see [42] for instance. Their transpose enables us to restrict some Global interface data to the boundary of a subdomain.

2.2.2. Local problems

The Local problems are set on the discretized subdomains $\Omega^{(s),F}$. Boolean matrix $\mathbf{T}^{(s),F}$ is the trace operator on the Local mesh $\Omega^{(s),F} \rightarrow \Gamma^{(s),F}$. For a good matching of the models, the interface is assumed to suit edges of the Local elements. Anyhow, we do not require matching Global and Local discretization, and we introduce Global-to-Local transfer matrix $\mathbf{J}^{(s)}$ which enables us to define Local Dirichlet problems with boundary conditions coming from the Global model.

The fine problems can be written as:

$$\left|\begin{array}{l}\text{Given } \mathbf{u}^G_\Gamma \text{ on } \Gamma; \forall s > 0, \text{ find } \mathbf{u}^{(s),F} \text{ in } \Omega^{(s),F} \text{ and } \boldsymbol{\lambda}^{(s),F} \text{ on } \Gamma^{(s)} \text{ such that}\\ \mathbf{K}^{(s),F}\mathbf{u}^{(s),F} = \mathbf{f}^{(s),F} + \mathbf{T}^{(s),F^T}\boldsymbol{\lambda}^{(s),F}\\ \mathbf{T}^{(s),F}\mathbf{u}^{(s),F} = \mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{u}^G_\Gamma\end{array}\right. \tag{2}$$

with $\boldsymbol{\lambda}^{(s),F}$ the local nodal reactions of the subdomain s.

2.2.3. Reference problem

The Reference problem is the collection of Local problems connected to the same interface displacement $\mathbf{u}^G_\Gamma$ and such that the nodal reactions are in balance once projected back on the Global interface.

Presented in the previous section, the main role of the complement subdomain 0 is to help process the nodal reaction $\boldsymbol{\lambda}^{(0),G}$ using $\mathbf{u}^{(0),G}$, the restriction of the global solution on subdomain:

$$\boldsymbol{\lambda}^{(0),G} = \mathbf{T}^{(0),G}(\mathbf{K}^{(0),G}\mathbf{u}^{(0),G} - \mathbf{f}^{(0),G}) \tag{3}$$

We are now in position to formulate the Reference problem:

$$\left|\begin{array}{l}\text{Find } \mathbf{u}^G_\Gamma \text{ on } \Gamma \text{ s.t}\\ \mathbf{r}_\Gamma := -\left(\mathbf{A}^{(0),G}\mathbf{J}^{(0),G}\boldsymbol{\lambda}^{(0),G} + \sum_{s=1}^{N}\mathbf{A}^{(s)}\mathbf{J}^{(s)^T}\boldsymbol{\lambda}^{(s),F}\right) = 0\\ \text{where the reactions are obtained from (2) and (3).}\end{array}\right. \tag{4}$$

with $\mathbf{r}_\Gamma$ the residual corresponding to the lack of balance on the interface Γ. Note that $\mathbf{J}^{(0),G}$ is in fact the identity matrix.

2.2.4. Condensed problems

As usual with domain decomposition methods, the process is fully driven by the convergence of interface quantities. For the analysis of the method, it is thus convenient to condense these previous problems at the interface.

We then deduce from the system (2) the Dirichlet-to-Neumann operator for the Local problems which can be written as:

$$\boldsymbol{\lambda}^{(s),F} = \mathbf{S}^{(s),F}\mathbf{u}_{\Gamma}^{(s),F} - \mathbf{b}^{(s),F} \tag{5}$$

With:

$$\begin{cases} \mathbf{S}^{(s),F} & = \mathbf{K}_{\Gamma\Gamma}^{(s),F} - \mathbf{K}_{\Gamma i}^{(s),F}\mathbf{K}_{ii}^{(s),F^{-1}}\mathbf{K}_{i\Gamma}^{(s),F} \\ \mathbf{b}^{(s),F} & = \mathbf{f}_{\Gamma}^{(s),F} - \mathbf{K}_{\Gamma i}^{(s),F}\mathbf{K}_{ii}^{(s),F^{-1}}\mathbf{f}_{i}^{(s),F} \end{cases}$$

where $\mathbf{S}^{(s),F}$ is the well-known Schur complement and $\mathbf{b}^{(s),F}$ is the condensed right-hand side.

We use then the same notation for the condensation of Global subdomains, we can rewrite the Global problem (1) as:

$$\underbrace{\left(\sum_{s=0}^{N}\mathbf{A}^{(s)}\mathbf{S}^{(s),G}\mathbf{A}^{(s)^T}\right)}_{\mathbf{S}^G}\mathbf{u}_{\Gamma}^{G} = \underbrace{\left(\sum_{s=0}^{N}\mathbf{A}^{(s)}\mathbf{b}^{(s),G}\right)}_{\mathbf{b}^G} + \mathbf{p}_{\Gamma} \tag{6}$$

The reference then ends up to being:

$$\left|\begin{array}{l} \text{Find } \hat{\mathbf{p}}_{\Gamma} \text{ such that} \\ \displaystyle\sum_{s=0}^{N}\mathbf{A}^{(s)}\mathbf{J}^{(s)^T}\left(\mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\left(\mathbf{S}^{G^{-1}}\left(\hat{\mathbf{p}}_{\Gamma}+\mathbf{b}^{G}\right)\right)-\mathbf{b}^{(s),F}\right)=0 \end{array}\right.$$

In order to ease the reading, we introduce the notations:

$$\begin{cases} \hat{\mathbf{S}}^{(s),F} & = \mathbf{A}^{(s)}\mathbf{J}^{(s)^T}\mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T} \\ \hat{\mathbf{b}} & = \sum_{s=0}^{N}\mathbf{A}^{(s)}\mathbf{J}^{(s)^T}(\mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{S}^{G^{-1}}\mathbf{b}^{G}-\mathbf{b}^{(s),F}) \end{cases} \tag{7}$$

so that the system to be solved can be written as:

$$\left(\sum_{s=0}^{N}\hat{\mathbf{S}}^{(s),F}\right)\mathbf{S}^{G^{-1}}\hat{\mathbf{p}}_{\Gamma}+\hat{\mathbf{b}}=0 \tag{8}$$

This system can be viewed as the primal domain decomposition formulation [43] of the Reference problem $\left(\sum_{s=0}^{N}\hat{\mathbf{S}}^{(s),F}\right)\mathbf{u}_{\Gamma}^{G} = \left(\sum_{s=0}^{N}\mathbf{A}^{(s)}\mathbf{J}^{(s)^T}\mathbf{b}^{(s),F}\right)$ right-preconditioned by the Global problem $\mathbf{u}_{\Gamma}^{G} = \mathbf{S}^{G^{-1}}(\mathbf{b}^{G}+\hat{\mathbf{p}}_{\Gamma})$. This preconditioner is of course much less scalable than the classical BDD strategy [44] where local inverses of the Local representation are used in conjunction with a much smaller coarse (global) problem. But this preconditioner provides a pertinent initialization $\mathbf{u}_{\Gamma,0}^{G} = \mathbf{S}^{G^{-1}}\mathbf{b}^{G}$ and it can be expected to introduce less irregularity at the interface, making it useless to add an enriched (spectral) coarse problem [45]. Contrarily to the BDD approach where Krylov solver is mandatory (because the spectrum of the preconditioned operator is bounded from below by 1 [46]), the Global/Local coupling supports stationary iteration. More, the right-preconditioning does not modify the nature of the residual of the system to be solved, allowing flexibility, and in our context, asynchronism.

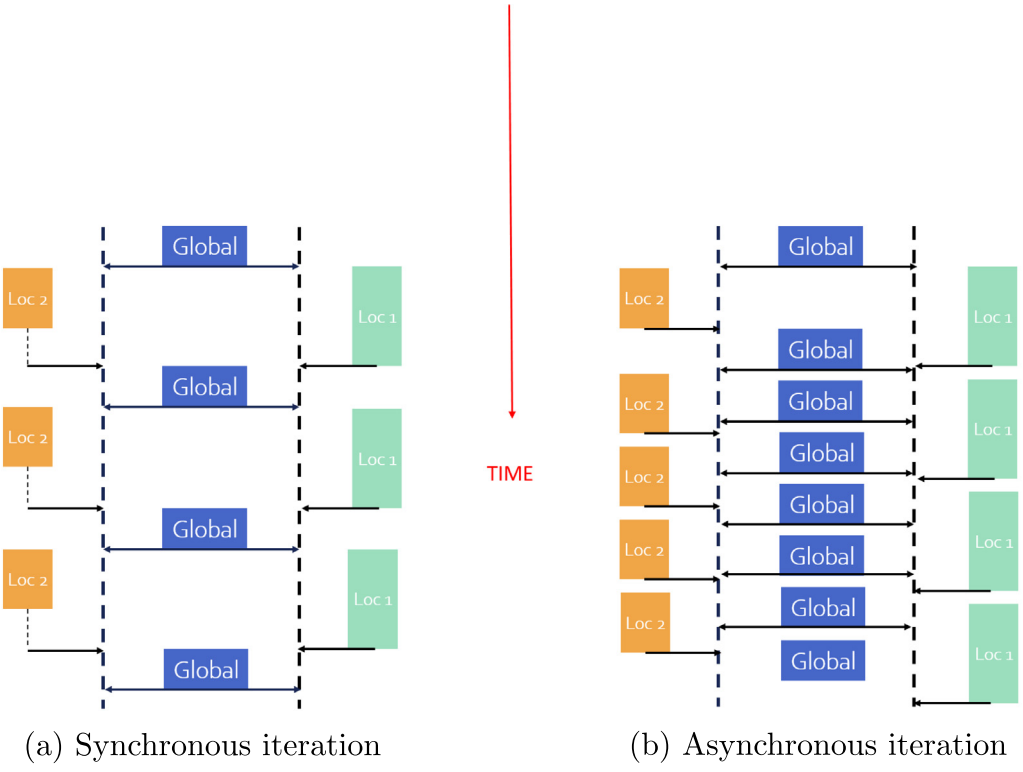
2.2.5. Global/Local coupling

The aim of the coupling is to achieve (4) using ((1), (2), (3)). To do so, a simple modified Richardson iteration is used. Starting from $\mathbf{p}_{\Gamma} = 0$, we compute $\mathbf{u}^{G}$ as in (1), then we use $\mathbf{u}_{\Gamma}^{G}$ as a Dirichlet condition to compute the Local reactions $\boldsymbol{\lambda}^{(s),F}$ using (2) and (3), finally the residual $\mathbf{r}_{\Gamma}$ is the lack of balance between the nodal reactions as in (4). If the residual is not small enough, the interface load is updated as $\mathbf{p}_{\Gamma} = \mathbf{p}_{\Gamma} + \omega\mathbf{r}_{\Gamma}$. It can be proved that under the chosen hypothesis, there exist $\omega_{\max} > 0$ such that the iteration converges for all $0 < \omega < \omega_{\max}$. In practice, dynamic relaxation through Aitken's δ^2 gives excellent performance.

Algorithm 1 corresponds to applying a modified Richardson iteration to (8). The relaxation parameter is discussed in the next section as a particular case of the asynchronous iteration. In practice, it is recommended to use dynamic relaxation with Aitken's formula.

Algorithm 1: Synchronous stationary iterations

Initialization $\mathbf{p}_\Gamma = 0$, ω sufficiently small
while $\|\mathbf{r}\|$ is too large **do**
  Resolution of the Global system (1) or (6), $\mathbf{u}_\Gamma^G = \mathbf{S}^{G^{-1}}(\mathbf{p}_\Gamma + \mathbf{b}^G)$
  if $\Omega^{(0),G}$ exists **then**
    Post-processing (3), $\mathbf{q}^{(0),G} := \boldsymbol{\lambda}^{(0),G} = \mathbf{S}^{(0),G}\mathbf{u}_\Gamma^{(0),G} - \mathbf{b}^{(0),G}$
  end
  Global *scatters* $\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G$ to subdmains $s > 0$
  for $s > 0$ **do**
    Patch *receives* $\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G$
    Local solution (2), $\boldsymbol{\lambda}^{(s),F} = \mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G - \mathbf{b}^{(s),F}$
    Patch *sends* of $\mathbf{q}^{(s)} := \mathbf{J}^{(s)^T}\boldsymbol{\lambda}^{(s),F}$ to the Global
  end
  Global *gathers* all $\mathbf{q}^{(s)}$
  Global computes residual $\mathbf{r} = -\sum_s \mathbf{A}^{(s)}\mathbf{q}^{(s)}$
  Global updates $\mathbf{p}_\Gamma = \mathbf{p}_\Gamma + \omega\mathbf{r}$
end



(a) Synchronous iteration (b) Asynchronous iteration

Fig. 3. Time course of the Global/Local coupling in the case of two patches.

3. Asynchronous version

3.1. Introduction

In previous section, the Global/Local coupling has been presented as a robust and non-invasive method. However, from a performance point of view, it remains limited and less adapted to high performance computing, due to its alternating nature, see [9]. As an illustration, we consider the case of two zones of interest and a global problem as presented in Fig. 1.

Fig. 3(a) presents the time sequence of the classical synchronous approach, which alternates between global and parallel local calculations. Such an organization generates waiting and inactivity times on both sides, which seriously affects the performance. This phenomenon would be even amplified by bad load balancing, communication delays, or machine failures.

We establish an asynchronous parallel version of the Global/Local coupling to address these problems. The idea is to allow each processor to work at its own pace without waiting for the other processors, considering only the latest version of the available data. This technique leads to the time sequence of Fig. 3(b) where processors only wait when they have no new data to process

Based on Fig. 3(b), the algorithm Section 3.1, presents an asynchronous version of the algorithm 1.

Algorithm 2: Asynchronous iterations

Initialization $\mathbf{p}_\Gamma = 0$, ω sufficiently small
while $\|\mathbf{r}\|$ is too large **do**
  if Rank 0 is available and detects at least one new $\mathbf{q}^{(s)}$ **then**
    Resolution of the Global system (1) or (6),
    $\mathbf{u}_\Gamma^G = \mathbf{S}^{G^{-1}}(\mathbf{p}_\Gamma + \mathbf{b}^G)$
    if $\Omega^{(0),G}$ exists **then**
      Post-processing (3), $\mathbf{q}^{(0),G} := \boldsymbol{\lambda}^{(0),G} = \mathbf{S}^{(0),G}\mathbf{u}_\Gamma^{(0),G} - \mathbf{b}^{(0),G}$
    end
    Global *scatters* $\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G$ to subdmains $s > 0$
  end
  for $s > 0$ **do**
    if Subdomain $s > 0$ is available and detects new $(\mathbf{A}^{(s)^T}\mathbf{u}^G)$ **then**
      Patch *receives* $\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G$
      Local solution (2), $\boldsymbol{\lambda}^{(s),F} = \mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G - \mathbf{b}^{(s),F}$
      Patch *sends* of $\mathbf{q}^{(s)} := \mathbf{J}^{(s)^T}\boldsymbol{\lambda}^{(s),F}$ to the Global
    end
  end
  Global *gathers* all $\mathbf{q}^{(s)}$
  Global computes residual $\mathbf{r} = -\sum_s \mathbf{A}^{(s)}\mathbf{q}^{(s)}$
  Global updates $\mathbf{p}_\Gamma = \mathbf{p}_\Gamma + \omega\mathbf{r}$
end

Note that the detection of the convergence of asynchronous iteration may require a specific, sometime complex, protocol [47,48]. Since the Global/Local coupling always assembles the residual on the Global model, our stopping criterion can be the same as in the synchronous case, simply based on the norm of the residual.

3.2. Convergence proof of the asynchronous iteration

Proving the convergence of asynchronous iteration can be tedious. In our case, we have the advantage of the Global domain playing a special role such that it can be used to cadence the solver. Referring to Algorithm Section 3.1, we can consider that during the step from iteration j to $j+1$, some patches provide new pieces of information in order to evaluate the residual, anyhow these pieces of information may be related to old configurations

$p_{j-\sigma(s,j)}$ where $\sigma(s,\, j) \geqslant 0$ is a delay function. So that we can model the asynchronous iteration as:

$$\begin{cases} \mathbf{u}^G_{\Gamma,j} = \mathbf{S}^{G^{-1}}(\mathbf{b}^G + \mathbf{p}_{\Gamma j}) \\ \text{If } s=0 : \mathbf{q}^{(0),G}_j = \mathbf{S}^{(0),G}(\mathbf{A}^{(0)^T}\mathbf{u}^G_{\Gamma,j} - \mathbf{b}^{(0),G}) \\ \text{If } s>0 : \mathbf{q}^{(s)}_j = \begin{cases} \mathbf{J}^{(s)^T}(\mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{u}^G_{\Gamma,j-\sigma(s,j)} - \mathbf{b}^{(s),F}) \text{ if updated} \\ \mathbf{q}^{(s)}_{j-1}, \text{ if not updated} \end{cases} \\ \mathbf{r}_j = -\left(\mathbf{A}^{(0),G}\mathbf{q}^{(0),G}_j + \sum_{s>0}\mathbf{A}^{(s)}\mathbf{q}^{(s)}_j\right) \\ \mathbf{p}_{\Gamma j+1} = \mathbf{p}_{\Gamma j} + \omega\mathbf{r}_j \end{cases} \tag{9}$$

For subdomains not updated, we set: $\sigma(s,\, j) = \sigma(s,\, j-1) + 1$.

It is crucial to note that if it exists, subdomain 0 always contributes to the evaluation of the residual because computing $\mathbf{q}^{(0),G}_{j+1}$ is only a cheap postprocessing of the Global solution. In order to unify notations, we introduce $\sigma(0,\, j) = 0$, $\forall j$, and then:

$$\begin{aligned} \mathbf{p}_{\Gamma j+1} &= \mathbf{p}_{\Gamma j} - \omega\sum_{s=0}^{N}\mathbf{A}^{(s)}\mathbf{J}^{(s)^T}(\mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{S}^{G^{-1}}(\mathbf{p}_{\Gamma j-\sigma(s,j)} + \mathbf{b}^G) - \mathbf{b}^{(s),F}) \\ &= \mathbf{p}_{\Gamma j} - \omega\left(\sum_{s=0}^{N}\hat{\mathbf{S}}^{(s),F}\mathbf{S}^{G^{-1}}\mathbf{p}_{\Gamma j-\sigma(s,j)} + \hat{\mathbf{b}}\right) \end{aligned} \tag{10}$$

Note that this expression is valid only after all local patches have at least contributed once to the estimation of the residual.

In order to ensure that at some point all patches provide new information, we assume that:

$$\exists D \geqslant 0 \text{ such that } \forall(s,\, j),\ \ \sigma(s,\, j) \leqslant D \tag{11}$$

For a given delay $0 \leqslant k \leqslant D$, we write $\varpi(k,\, j)$ the set of subdomains (s) such that $\sigma(s,\, j) = k$ so that the iteration can be rewritten as:

$$\mathbf{p}_{\Gamma j+1} = \mathbf{p}_{\Gamma j} - \omega\left(\sum_{k=0}^{D}\left(\sum_{s\in\varpi(k,j)}\hat{\mathbf{S}}^{(s),F}\right)\mathbf{S}^{G^{-1}}\mathbf{p}_{\Gamma j-k} + \hat{\mathbf{b}}\right) \tag{12}$$

3.2.1. Tools for convergence study

The asynchronous Richardson iteration was the object of [49] in the case of a maximal delay of 2. In order to extend the method, we rely on the theory of paracontractions [41].

Let (T_m) be a finite family of paracontractions with a common fixed point $\hat{x}$ in some Hilbert space E. In other words:

- $\forall x \in E,\ \ \|T_m(x) - \hat{x}\| < \|x - \hat{x}\|$ or $T_m(x) = x$,
- $\forall m,\ \ T_m(\hat{x}) = \hat{x}$.

Then a sequence of the form:

$$x_{j+1} = T_{m(j)}(x_j) \tag{13}$$

converges to $\hat{x}$, assuming that all the paracontractions (T_m) are sufficiently frequently activated.

3.2.2. Analysis of Global/Local coupling

In order to make appear paracontraction, we assume a non-zero delay $D > 0$, and we work in the "history space" obtained by concatenating the last $(D+1)$ values of $\mathbf{p}_{\Gamma j}$.

We can rewrite the history at iteration $j+1$ as:

$$\begin{pmatrix} \mathbf{p}_{\Gamma\, j+1} \\ \mathbf{p}_{\Gamma\, j} \\ \vdots \\ \mathbf{p}_{\Gamma\, j-D+1} \end{pmatrix} = \underbrace{\begin{pmatrix} \mathbf{I}-\omega\mathbf{X}_{j,0} & -\omega\mathbf{X}_{j,1} & \dots & -\omega\mathbf{X}_{j,D} \\ \mathbf{I} & 0 & \dots & 0 \\ 0 & \mathbf{I} & 0 & \dots \\ \dots & 0 & \mathbf{I} & 0 \end{pmatrix}}_{\mathbf{B}_j} \begin{pmatrix} \mathbf{p}_{\Gamma\, j} \\ \mathbf{p}_{\Gamma\, j-1} \\ \vdots \\ \mathbf{p}_{\Gamma\, j-D} \end{pmatrix} - \begin{pmatrix} \omega\tilde{\mathbf{b}} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{14}$$

$$\text{with } \mathbf{X}_{j,k} = \left(\sum_{s\in\varpi(k,j)} \hat{\mathbf{S}}^{(s),F} \right) \mathbf{S}^{G^{-1}}$$

Since $\forall j,\ \sum_k \mathbf{X}_{j,k}\hat{\mathbf{p}}_\Gamma + \tilde{\mathbf{b}} = 0$, the vector obtained by repeating the solution $\hat{\mathbf{p}}_\Gamma$ of (8) is a fixed point for the above iteration.

In order to prove the paracontracting nature of the iteration, it suffices to prove that any matrix $\mathbf{B}_j$ of (14) can be turned into contraction by correctly selecting the relaxation $\omega > 0$. Since $\mathbf{B}_j$ is a block companion matrix, it seems natural to study its spectrum and prove that it can be bounded by 1.

The eigenvalues (λ) of $\mathbf{B}_j$ are the roots of the polynomial:

$$\det\left((1-\lambda)\lambda^D\mathbf{I} - \omega\sum_{k=0}^{D} \lambda^{D-k}\mathbf{X}_{j,k} \right) = 0 \tag{15}$$

This is the determinant of a real momic matrix polynomial [50] (a polynomial of the complex variable λ with matrix coefficients where the highest power has the identity matrix as coefficient). In order to benefit from the underlying symmetry, we can introduce the Cholesky factorization of $\mathbf{S}^G = \mathbf{L}\mathbf{L}^T$, left-multiply the polynomial by $\mathbf{L}^{-1}$ and right-multiply it by $\mathbf{L}$, the roots of (15) are also the root of the polynomial $P_{j,\omega}(\lambda)$:

$$P_{j,\omega}(\lambda) = \det\left((1-\lambda)\lambda^D\mathbf{I} - \omega\sum_{k=0}^{D} \lambda^{D-k}\hat{\mathbf{X}}_{j,k} \right) = 0 \tag{16}$$

where $\hat{\mathbf{X}}_{j,k} = \mathbf{L}^{-1}\mathbf{X}_{j,k}\mathbf{L} = \mathbf{L}^{-1}\left(\sum_{s\in\varpi(k,j)} \hat{\mathbf{S}}^{(s),F}\right)\mathbf{L}^{-T}$.

Using the absolute continuity of the roots of a polynomial with respect to its coefficients (see [51,52] for instance), we see that for a small enough ω, the eigenvalues tend to concentrate around the roots of $P_{j,0}(\lambda) = \det((1-\lambda)\lambda^D\mathbf{I})$, that is to say around 0 and 1.

Let $\tilde{\lambda}_{j,\omega}$ be one of the roots of $P_{j,\omega}$, and $\varepsilon = \min\left(\sin\left(\frac{\pi}{3D}\right), \frac{1}{2}\right)$, we can find ω_0 such that $\omega < \omega_0 \Rightarrow |\tilde{\lambda}_{j,\omega} - \tilde{\lambda}_{j,0}| < \varepsilon$. At that point, the roots that tend to zero have all modulus less than $\varepsilon < 1$, only the roots that tend to 1 could pose a problem. In what follows, $\tilde{\lambda}_{j,\omega}$ is such a root that tends to 1, we can bound its modulus and argument, see Fig. 4.

$|\tilde{\lambda}_{j,\omega} - 1| < \varepsilon$ implies that:

$$\begin{aligned} &1-\varepsilon < |\tilde{\lambda}_{j,\omega}| < 1+\varepsilon \\ &|\sin(\arg(\tilde{\lambda}_{j,\omega}))| < \varepsilon \end{aligned} \tag{17}$$

For $\varepsilon = \sin\frac{\pi}{3D}$ and $0 \leqslant k \leqslant D$, we have bounds on the modulus and on the reel part (symbol $\Re$):

$$\begin{aligned} &(1-\varepsilon)^D < |\tilde{\lambda}_{j,\omega}|^k < (1+\varepsilon)^D \\ &\Re(\tilde{\lambda}^k_{j,\omega}) = |\tilde{\lambda}_{j,\omega}|^k \cos(k\arg(\tilde{\lambda}_{j,\omega})) > \frac{(1-\varepsilon)^D}{2} \end{aligned} \tag{18}$$

Let $\tilde{\mathbf{v}}_{j,\omega}$ be an eigenvector of the matrix polynomial associated with $\tilde{\lambda}_{j,\omega}$:

$$(1-\tilde{\lambda}_{j,\omega})\tilde{\lambda}^D_{j,\omega}\tilde{\mathbf{v}}_{j,\omega} - \omega\sum_{k=0}^{D} \tilde{\lambda}^{D-k}_{j,\omega}\hat{\mathbf{X}}_{j,k}\tilde{\mathbf{v}}_{j,\omega} = 0 \tag{19}$$

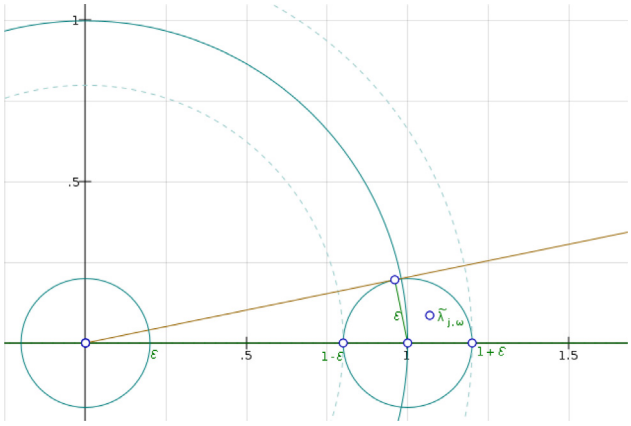


Fig. 4. Constraining roots near 1.

We can left-multiply the expression by the Hermitian transpose $\tilde{\mathbf{v}}_{j,\omega}^H$:

$$(1-\tilde{\lambda}_{j,\omega})\tilde{\lambda}_{j,\omega}^D \tilde{\mathbf{v}}_{j,\omega}^H \tilde{\mathbf{v}}_{j,\omega} - \omega \sum_{k=0}^{D} \tilde{\lambda}_{j,\omega}^{D-k} \tilde{\mathbf{v}}_{j,\omega}^H \hat{\mathbf{X}}_{j,k} \tilde{\mathbf{v}}_{j,\omega} = 0 \tag{20}$$

To simplify, $\tilde{\mathbf{v}}_{j,\omega}$ can be chosen of unit Euclidean norm. For $\omega < \omega_0$ we have $\tilde{\lambda}_{j,\omega} \neq 0$, and then:

$$\begin{aligned} \tilde{\lambda}_{j,\omega} &= 1 - \omega \sum_{k=0}^{D} \frac{\|\tilde{\mathbf{v}}_{j,\omega}\|^2_{\hat{\mathbf{X}}_{j,k}}}{\tilde{\lambda}_{j,\omega}^k} \\ |\tilde{\lambda}_{j,\omega}|^2 &= 1 - 2\omega \sum_{k=0}^{D} \frac{\Re(\tilde{\lambda}_{j,\omega}^k)\|\tilde{\mathbf{v}}\|^2_{\hat{\mathbf{X}}_{j,k}}}{|\tilde{\lambda}_{j,\omega}|^{2k}} + \omega^2 \left| \sum_{k=0}^{D} \frac{\|\tilde{\mathbf{v}}\|^2_{\hat{\mathbf{X}}_{j,k}}}{\tilde{\lambda}_{j,\omega}^k} \right|^2 \end{aligned} \tag{21}$$

Using (18), we have:

$$|\tilde{\lambda}_{j,\omega}|^2 < 1 - \omega \frac{\left(\sum_{k=0}^{D} \|\tilde{\mathbf{v}}\|^2_{\hat{\mathbf{X}}_{j,k}}\right)}{(1+\varepsilon)^D} + \omega^2 \frac{\left(\sum_{k=0}^{D} \|\tilde{\mathbf{v}}\|^2_{\hat{\mathbf{X}}_{j,k}}\right)^2}{(1-\varepsilon)^{2D}} \tag{22}$$

The sum of norms is simplified because each subdomain appears only once:

$$\begin{aligned} \sum_{k=0}^{D} \|\tilde{\mathbf{v}}\|^2_{\hat{\mathbf{X}}_{j,k}} &= \sum_{k=0}^{D} \tilde{\mathbf{v}}_{j,\omega}^H \hat{\mathbf{X}}_{j,k} \tilde{\mathbf{v}}_{j,\omega} = \tilde{\mathbf{v}}_{j,\omega}^H \mathbf{L}^{-1} \left(\sum_{k=0}^{D} \sum_{s \in \varpi(k,j)} \hat{\mathbf{S}}^{(s),F} \right) \mathbf{L}^{-T} \tilde{\mathbf{v}}_{j,\omega} \\ &= \tilde{\mathbf{v}}_{j,\omega}^H \mathbf{L}^{-1} \left(\sum_{s=0}^{N} \hat{\mathbf{S}}^{(s),F} \right) \mathbf{L}^{-T} \tilde{\mathbf{v}}_{j,\omega} \end{aligned} \tag{23}$$

Since $\tilde{\mathbf{v}}_{j,\omega}$ is of unit Euclidean norm, the term above can directly be bounded by the extremal eigenvalues of $\mathbf{L}^{-1}\left(\sum_{s=0}^{N} \hat{\mathbf{S}}^{(s),F}\right)\mathbf{L}^{-T}$ which coincide to the generalized eigenvalues of the pair of matrices $(\sum_{s=0}^{N} \hat{\mathbf{S}}^{(s),F}, \mathbf{S}^G)$:

$$\begin{aligned} &\alpha_{\min} \leqslant \sum_{k=0}^{D} \|\tilde{\mathbf{v}}\|^2_{\hat{\mathbf{X}}_{j,k}} \leqslant \alpha_{\max} \\ &\text{where the } (\alpha) \text{ solve } \det\left(\left(\sum_{s=0}^{N} \hat{\mathbf{S}}^{(s),F}\right) + \alpha \mathbf{S}^G\right) = 0 \end{aligned} \tag{24}$$

We thus obtain the upper bound:

$$|\tilde{\lambda}_{j,\omega}|^2 \leqslant 1 - \omega\frac{\alpha_{\min}}{(1+\varepsilon)^D} + \omega^2\frac{\alpha_{\max}^2}{(1-\varepsilon)^{2D}}, \qquad \forall 0 < \omega < \omega_0 \tag{25}$$

This is a bound of the form $|\tilde{\lambda}_{j,\omega}|^2 \leqslant 1 - A\omega + B\omega^2$ (with $0 < A < B$) which is a second degree polynomial in ω, and which is less than 1 for $0 < \omega < A/B$. As a consequence:

$$|\tilde{\lambda}_{j,\omega}| < 1 \text{ for } 0 < \omega < \omega_{\text{async}} = \min\left(\omega_0, \frac{(1-\varepsilon)^D\alpha_{\min}}{(1+\varepsilon)^{2D}\,\alpha_{\max}^2}\right) \tag{26}$$

This is probably an extremely crude bound, but it has the advantage to only depend on D and not on the configuration of the iteration (index j). Thus, such a relaxation makes any $\mathbf{B}_j$ a paracontraction, and it makes the asynchronous iteration converge.

Remark 1. For the synchronous iteration, the bound can be derived from (15) with $D = 0$, it is $0 < \omega < \omega_{\text{sync}} = \frac{2}{\alpha_{\max}}$. Note that $\omega_{\text{sync}} > \omega_{\text{async}}$.

3.3. Implementation details

Several approaches are available in the literature to implement asynchronous model. In [53,54], an efficient library is proposed for asynchronous domain decomposition solvers, based on classical non-blocking two-sided communications. In [33,55] the use of one-sided communications, also known as MPI-RDMA (Remote Direct Memory Access), is considered. The one-sided communication is meant to reduce management overhead. Note that the performance of the RDMA strongly depends on the MPI implementation and the network hardware.

The basic idea is that each rank exposes a so-called *window* of its local memory and grants other ranks write or read access. Ranks, in this case, are no longer identified as sender or receiver but as *origin* rank who initiates the operation and *target* rank. The latter does not participate in the data exchange.

The RMA-MPI workflow is based on the following five steps:

1. **Allocation of the window** (local memory buffer accessible from other ranks).
2. **Epoch opening**: beginning of the period when the window is open the other ranks.
3. **Data accessing**: Each origin rank can access the target ranks' window to *Put* (write data) or to *Get* (read data). See Fig. 5(a) where Processor 0 puts a data Y in Processor 1 window and Fig. 5(b) where Processor 1 gets a data Y from Processor 0 window.
4. **Epoch closing**: the target rank which closes the windows ensures that all accesses are completed (local synchronization). At this point, the target rank can read and process the data put by other ranks.
5. **Window freeing**: liberation of the memory buffer.

To secure the data access in a window, one may consider two ways:

Active synchronization consists in performing a collective blocking call on both the *target* and *origin* using the *MPI.Fence()* command at the beginning and at the end of the epoch to synchronize the data.

Passive synchronization emulates shared memory. The target processor is not involved in the management of the data, full asynchronous communication is possible. *MPI.Lock(Target rank)* opens an epoch and allows the origin processor to access securely the target's window. The epoch is then closed by *MPI.Unlock(Target rank)*. To ensure the completion of an operation within an epoch, one can use *MPI.Flush(Target rank)*

Algorithm 3 proposes an RDMA implementation of the asynchronous version of the Global/Local coupling algorithm Section 3.1 with passive synchronization. The principle is to have the subdomains compute whenever they

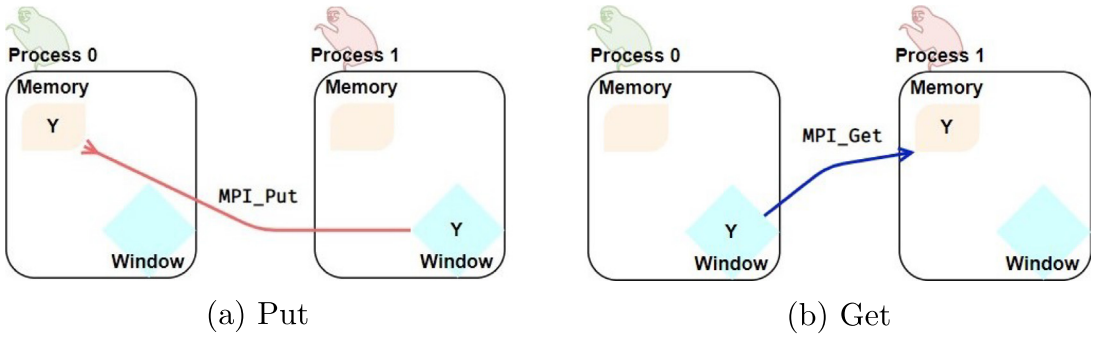


Fig. 5. One-sided communication concepts [56].

idle and a new piece of information becomes available: a new interface Dirichlet condition for the *local* patches, any new interface nodal reaction for the *global* model.

Algorithm 3: Asynchronous iterations using RDMA

Window creation + Initialization $\mathbf{p}_\Gamma = 0$, ω sufficiently small
MPI.Lock(target) (For all the window by specifying the specific target of each one) **while** $\|\mathbf{r}\|$ is too large **do**
  if Rank 0 is available and detects at least one new $\mathbf{q}^{(s)}$ **then**
    Resolution of the Global system (1) or (6), $\mathbf{u}_\Gamma^G = \mathbf{S}^{G^{-1}}(\mathbf{p}_\Gamma + \mathbf{b}^G)$
    if $\Omega^{(0),G}$ exists **then**
      Post-processing (3), $\mathbf{q}^{(0),G} := \boldsymbol{\lambda}^{(0),G} = \mathbf{S}^{(0),G}\mathbf{u}_\Gamma^{(0),G} - \mathbf{b}^{(0),G}$
    end
    Put $\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G$ in subdmains $s > 0$ windows + *Flush(subdomains s window)*
  end
  for $s > 0$ **do**
    if Subdomain $s > 0$ is available and detects new $(\mathbf{A}^{(s)^T}\mathbf{u}^G)$ **then**
      Local solution (2), $\boldsymbol{\lambda}^{(s),F} = \mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G - \mathbf{b}^{(s),F}$
      Put of $\mathbf{q}^{(s)} := \mathbf{J}^{(s)^T}\boldsymbol{\lambda}^{(s),F}$ to the rank 0 window + *Flush(0)*
    end
  end
  Global computes residual $\mathbf{r} = -\sum_s \mathbf{A}^{(s)}\mathbf{q}^{(s)}$
  Global updates $\mathbf{p}_\Gamma = \mathbf{p}_\Gamma + \omega\mathbf{r}$
end
MPI.UnLock(target) (For all the window by specifying the specific target of each one)

Remark 2. An RDMA implementation of the synchronous coupling in algorithm 1 is proposed in Algorithm 4, it makes use of active synchronization with **MPI.Fence()**.

Algorithm 4: Synchronous stationary iterations using RDMA

Window creation + Initialization $\mathbf{p}_\Gamma = 0$, ω sufficiently small
while $\|\mathbf{r}\|$ is too large **do**
  MPI.Fence()(For the global displacement window)
  if rank == 0 **then**
    Resolution of the Global system (1) or (6),
    $\mathbf{u}_\Gamma^G = \mathbf{S}^{G^{-1}}(\mathbf{p}_\Gamma + \mathbf{b}^G)$
    if $\Omega^{(0),G}$ exists **then**
      Post-processing (3), $\mathbf{q}^{(0),G} := \boldsymbol{\lambda}^{(0),G} = \mathbf{S}^{(0),G}\mathbf{u}_\Gamma^{(0),G} - \mathbf{b}^{(0),G}$
    end
    Put $\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G$ in subdmains $s > 0$ windows ;
  end
  MPI.Fence()(For the global displacement window)
  MPI.Fence()(For the local nodal reaction)
  if rank != 0 **then**
    Local solution (2), $\boldsymbol{\lambda}^{(s),F} = \mathbf{S}^{(s),F}\mathbf{J}^{(s)}\mathbf{A}^{(s)^T}\mathbf{u}_\Gamma^G - \mathbf{b}^{(s),F}$
    Patch *Put* $\mathbf{q}^{(s)} := \mathbf{J}^{(s)^T}\boldsymbol{\lambda}^{(s),F}$ in the rank 0 window
  end
  MPI.Fence()(For the local nodal reaction)
  MPI.Fence()(For the convergence detection window)
  if rank == 0 **then**
    Global computes residual $\mathbf{r} = -\sum_s \mathbf{A}^{(s)}\mathbf{q}^{(s)}$
    Global updates $\mathbf{p}_\Gamma = \mathbf{p}_\Gamma + \omega\mathbf{r}$
  end
  MPI.Fence()(For the convergence detection window)
end
Window free

4. Applications

To illustrate the theory presented above, we consider two kind of equations. First the Poisson equation, which models thermal problems:

$$\begin{aligned} \text{Find } u : \Omega \subset \mathbb{R}^d &\to \mathbb{R} \\ \operatorname{div}(a \operatorname{grad}(u)) &= 1 \text{ in } \Omega \\ u &= 0 \text{ on } \partial_d\Omega \\ \frac{\partial u}{\partial n} &= 0 \text{ on } \partial\Omega \setminus \partial_d\Omega \end{aligned} \tag{27}$$

for simplicity, we used unit source term and homogeneous boundary conditions. In some cases a contrast of conductivity coefficient a is used.

Second, the linear elasticity equation:

$$\begin{aligned} \text{Find } u : \Omega \subset \mathbb{R}^d &\to \mathbb{R}^d \\ \operatorname{div}(\sigma) + f &= 0 \text{ in } \Omega \\ u &= 0 \text{ on } \partial_d\Omega \\ \sigma \cdot n &= 0 \text{ on } \partial\Omega \setminus \partial_d\Omega \\ \sigma &= \frac{E}{1+\nu}\left(\varepsilon(u) + \frac{\nu}{1-2\nu}\operatorname{tr}(\varepsilon(u))I\right) \\ \varepsilon(u) &= \frac{1}{2}(\nabla u + (\nabla u)^T) \end{aligned} \tag{28}$$

Table 1
Size of the domains or the 2D test case.

Problem	Global	1st zone of interest	2nd zone of interest
#nodes	701	381	379

E is Young's modulus, and $\nu = 0.3$ is Poisson's coefficient. In some cases, a contrast of Young's modulus is used. The value of the source term f varies with the study cases.

We propose to assess the asynchronous Global/Local coupling on two academic examples: the simple 2D case of Figs. 1(a) and 1(b), and a more challenging 3D case involving many patches. In order to evaluate the performance we compare the following approaches:

- non-relaxed synchronous iteration ($\omega = 1$),
- Aitken-accelerated (synchronous) iteration,
- non-relaxed asynchronous iteration ($\omega = 1$),
- asynchronous iteration with optimized relaxation.

Aitken's acceleration can be viewed as an efficient way to find a good dynamic relaxation. The optimized relaxation coefficient for the asynchronous iteration is obtained by trial-and-error.

Our Ethernet network does not support RDMA communication by default. It generates implicit synchronizations when we use *MPI.Lock()* and *MPI.Unlock()* commands to check if new data is available in the target processor. In order to achieve the best possible time, we used a computational sequence slightly different than Fig. 3(b): processors always compute with the available data without checking for their novelty (thus possibly redoing the same calculus several times but never triggering unwanted sync).

Our code is realized in Python with *mpi4py* module [57]. It uses several other tools and software like *GMSH* [58] to generate the geometries and meshes of the studied cases. For the finite element approximation, we use the *GetFEM* library [59].

The study was carried out with the cluster of the LMPS simulation center using several workstations with an Ethernet network. These machines are quite heterogeneous with 4 different generation of CPUs :(Intel(R) Xeon(R) CPU E5-1660 v3 (Haswell) @ 3.00 GHz, Intel(R) Xeon(R) CPU E5-2630 v4 (Broadwell) @ 2.20 GHz, Intel(R) Xeon(R) Silver 4116 CPU (Skylake) @ 2.10 GHz, Intel(R) Xeon(R) W-2255 CPU (Cascade Lake) @ 3.70 GHz. These machines have between 8 and 12 cores.

To carry out the following studies, we use several machines that can be considered computational nodes of the cluster presented above with heterogeneous architectures. The increase in the number of computational cores implies the addition of new nodes. We do not oversubscribe but exploit all the cores of the used nodes. It can therefore be assumed that the global problem, which is computed by processor 0, will communicate faster with the subdomains that are computed on cores of the same processor. It means that even if the size of the subdomains is the same and the resolution time is the same, the switch to asynchronous will allow the subdomains closest to the processor 0 to advance faster than the others. Note that Node 0 is a 12-core machine so that all computations involving no more than 11 patches are run on a single node.

4.1. Simple 2D test-case

To begin with the illustrations, we use the test-case of Figs. 1(a) and 1(b) where the patches only introduce geometric alterations. The patches and the global model are treated on three different CPUs.

As shown in Table 1, the problem is of very small dimension, and the patches are well-balanced, which is in favor of synchronous algorithms.

Tables 2 and 3 present the performance in terms of time and number of iterations. In the asynchronous cases, the number of local solves may differ from the number of iterations, so the range of the number of local solves is also indicated. For these small cases, Aitken remains unbeatable. We observe the interest of finding a good relaxation for the asynchronous iteration to perform better than the non-relaxed synchronous iteration.

To explain the choice of optimal relaxation coefficient, we present in Fig. 6 the computation times obtained for different relaxation coefficients. The plot tends to show a fairly well-marked minimum. Note that the same

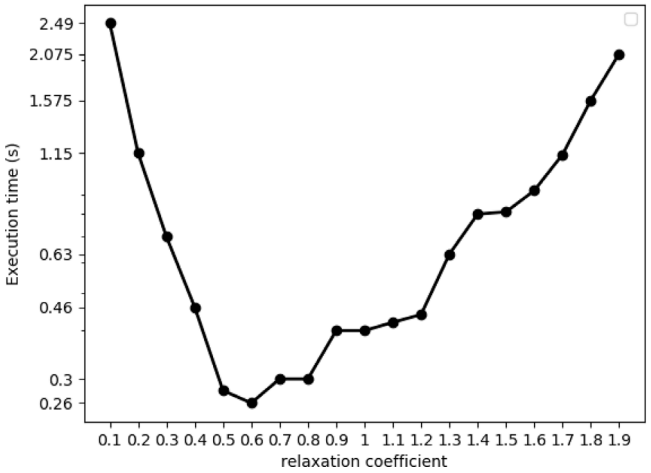


Fig. 6. Relaxation coefficient study for thermal problem.

Table 2
2D test-case: performance for thermal problem.

Variant	Sync.$\omega = 1$	Sync.Aitken	Async.$\omega = 1$	Async.ω_{opt}
Time (s)	0.31	0.17	0.4	0.26
#iter. glob.	23	12	43	35
#loc. sol. [min, max]	·	·	[95, 106]	[82, 85]

Table 3
2D test-case: performance for the elasticity problem.

Variant	Sync.$\omega = 1$	Sync.Aitken	Async.$\omega = 1$	Async.ω_{opt}
Time (s)	0.67	0.3	0.6	0.52
#iter. glob.	43	16	53	48
#loc. sol. [min, max]	·	·	[112,119]	[100,107]

approach to choose the relaxation coefficient is applied to all the examples that will follow. However, the value of the coefficient is insignificant as it depends not only on the mechanical problem but also on the hardware configuration, so we did not find it useful to systematically write it.

What is more interesting to observe is the large amount of computation that can be done by the asynchronous solver thanks to the removal of waiting time.

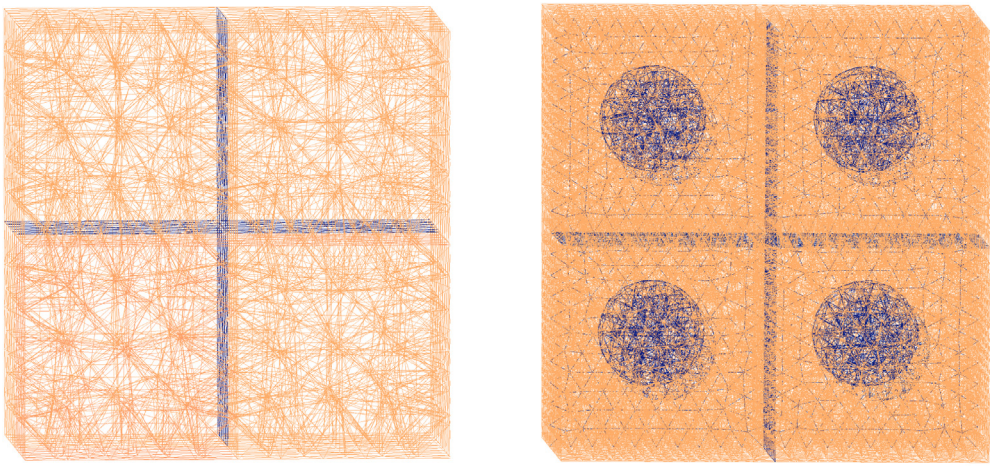
4.2. Weak scalability 3D test-case

Weak scalability tests aim at proving the ability of the method to solve large problems in reasonable time.

In order to be able to generate test-cases with many patches, we created a cuboid geometry made out of n^3 ($n = 2..7$) cube patches. As classically done for weak scalability assessment of domain decomposition methods, the size of the domain increases with the number of subdomains. Note that the whole domain is covered with patches ($\Omega^{(0),G} = \emptyset$). The Global model is homogeneous, whereas the Local models contain one softer spherical inclusion, see Figs. 7(a) and 7(b). One side of the Global model is submitted to Dirichlet conditions.

In the case of thermal problems, the inclusions have a diffusion coefficient 10 times lower than the rest of the domain, whereas in the elasticity case the Young's modulus in the inclusions is 100 times lower than in the rest of the domain.

Even if their meshes are not identical, the patches are well-balanced in terms of degrees of freedom and numerical complexity (since the problem is linear). Of course, the Global model grows along the study, from 8 times smaller than one patch to 3.7 times larger. This is a strong limitation of the method in comparison with classical domain



(a) Global model (b) Reference model

Fig. 7. Weak scalability test-case: 2 × 2 × 2 subdomains.

Table 4
Number of nodes in the meshes for the weak scalability study.

#subdomains	8	27	64	125	216	343
#nodes of glob. problem	233	667	1449	2681	4465	6903
#nodes of per loc. subdomain	1858	1858	1858	1858	1858	1858

Table 5
Analysis of the time spent in communication (64-subdomain thermal case).

#ranks	Aitken #iter #time (s) [% communication time]	Asynchronous #iter. glob.[#loc. sol. [min, max]] #time (s) [% communication time]
9	25 & 11.72 s [30%]	334[49 – 54] & 22.4 s [10%]
17	25 & 8.08 s [80%]	182[56 – 77] & 13.25 s [10%]
33	25 & 4.53 s [71%]	104[65 – 124] & 8.13 s [16%]
65	25 & **8.57 s [97%]**	105[81 – 160] & **8.40 s [46%]**

decomposition methods were the coarse problem's growth is much more moderate. Table 4 sums up the number of nodes for each case.

In order to better grasp the impact of synchronization on communication and waiting time, we propose a preliminary study based on the thermal problem set on the well-balanced 64-subdomain case (with a heterogeneity ratio of 100, different from the rest of the thermal experiments). We use different numbers of MPI processes. This means that for less than 65 processes, one MPI rank has to handle several subdomains. We study the percentage of communication time in the total simulation time.

We consider the Aitken accelerator for synchronous relaxation and optimal relaxation for asynchronous. In Table 5, we summarize the number of iterations, the calculation time, and the percentage that the communication time represents of this calculation time.

We observe that, in this case, the asynchronous approach is globally slower than the accelerated synchronous one. However, that the proportion of time spent in communication increases strongly in the synchronous case (up to 97%) and much more moderately in the asynchronous case (never more than 50%), which leads to the asynchronous approach being faster in the 65-process case. In particular the transition between one node (9 subdomains) computation and two nodes (17 subdomains) leads to a strong increase of the time spent in communication in the synchronous case whereas it is unmoved in the asynchronous case.

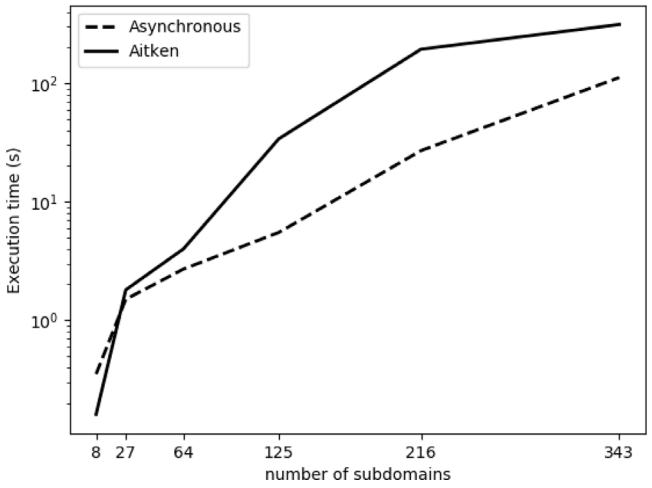


Fig. 8. Time performance in the weak scalability study for linear thermal problem.

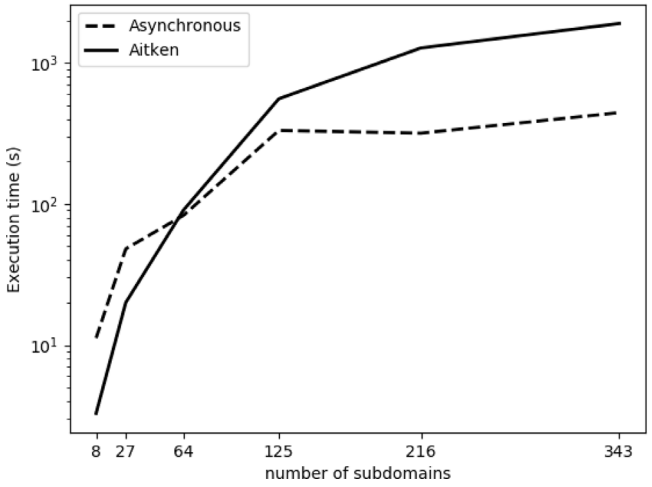


Fig. 9. Time performance in the weak scalability study for linear elasticity problem.

Figs. 9 and 8 compare the performance in wall-clock time of the relaxed asynchronous iteration (with hand-tuned relaxation) and the synchronous iteration with Aitken's dynamic relaxation. We observe the good performance of the asynchronous version despite the good load-balancing.

For the small test-cases (8 and 27 subdomains), the size of the global problem is negligible compared to the size of the local problems. This means that the sequential phase of the synchronous coupling is realized very quickly and this leads to the Aitken accelerator being faster than the asynchronous solver. However, for 64 subdomains and more, this step becomes heavier and takes more synchronous time. For the asynchronous method, the Global solve is realized simultaneously with the local solves. Thus, the execution time increases very slightly from one case to another and remains 2 to 3 times less than for Aitken.

Tables 6 and 7 gather the number of iterations for each case. In the asynchronous case, the number of iterations (or Global solves) is given as well as the minimum and maximum numbers of patches' solve. We see that the number of iterations barely varies in the synchronous experiments (in particular for the thermal problem) for all studied cases.

For the asynchronous solver, it can be seen that in the 8 and 27 patches cases where the global problem is very small, many more solves are performed by the global domain than by the local patches. Because of the non-waiting asynchronous model the global problem repeats several times the same calculation without having new information

Table 6
Weak scalability: Number of iterations in the thermal case.

#patches	8	27	64	125	216	343
Aitken #iter.	11	13	12	11	11	11
Async. #iter. glob.	255	256	87	65	69	71
Async. #loc. sol. [min, max]	[32,39]	[43,74]	[49,153]	[84,207]	[276,694]	[407,2902]

Table 7
Weak scalability: Number of iterations in the elasticity case.

#patches	8	27	64	125	216	343
Aitken #iter.	22	21	25	25	26	29
Async. #iter. glob.	2065	1349	372	296	295	209
Async. #loc. sol. [min, max]	[78,240]	[102,237]	[128,475]	[157,517]	[147,514]	[175,407]

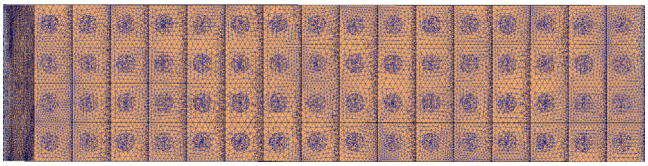


Fig. 10. Local representation with unbalanced subdomains.

Table 8
Mesh.

	Global	Smallest local	Biggest local
#nodes	5490	534	4698

from the locals, however when the size of this problem increases (more than 64 subdomains), we begin to see that the patches make more repeated iterations while waiting for the update of the global problem which performs only a few iterations.

Note the performance achieved in the elasticity case (2 times faster) despite the tremendous number of iterations (7 times more).

4.3. Poor load balancing

We wish to evaluate the influence of a significant disequilibrium in the number of nodes to be handled by processors. We start from a geometry formed with a $16 \times 4 \times 4$ repetition of cubes with spherical inclusion (this time 1000 times stiffer than the rest of the domain), see Fig. 10.

Each Local subdomain has a randomly chosen number of nodes compared to the other subdomains, allowing to have very refined subdomains and others slightly refined. Table 8 summarizes the number of nodes for the global problem and the smallest and largest number of nodes among the 256 Local subdomains. We can see that the most refined subdomain is ten times larger than the least refined.

This case study has been performed using 257 processors, one for the global problem and one processor for each one of the 256 local problems.

Tables 9 and 10 show the computation time and the number of iterations.

We see that even if the number of iterations can be very large in the asynchronous case, the CPU time is much reduced: 10 times in the thermal case and 2 times for the elasticity case. Again, this highlights the prohibitive cost of synchronization.

Table 9
Poor load balancing case: Iterations & Time (thermal problem).

Variant	Sync. Aitken	Async. ω_{opt}
Time (s)	881.55	79.44
#iter. glob.	36	506
#loc. sol. [min, max]	·	[348, 6788]

Table 10
Poor load balancing case: Iterations & Time (linear elasticity problem).

Variant	Sync. Aitken	Async. ω_{opt}
Time (s)	3509.6	1904.34
#iter. glob.	113	2354
#loc. sol. [min, max]	·	[818, 2951]

5. Conclusion

An asynchronous version of the non-intrusive global/local computation method has been presented for linear elliptic problems, starting from the new interpretation of the method as a right-preconditioned primal domain decomposition method. A proof of convergence has been established for the discretized system using paracontractions techniques. An implementation with MPI RDMA parallelization has been set. The coupling has been tested on linear thermal and elasticity problems involving up to hundreds of patches. The performance in terms of computation time is convincing: the asynchronous method (with hand tuned relaxation) is faster than the synchronous solver with Aitken's dynamic relaxation on a cluster of heterogeneous machines.

Future work should focus on finding an efficient estimation of the optimal relaxation for the asynchronous iteration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

This work was partly funded by the French National Research Agency as part of project ADOM, under grant number ANR-18-CE46-0008.

References

[1] F. Kelley, Mesh requirements for the analysis of a stress concentration by the specified boundary displacement method, in: Proceedings of the Second International Computers in Engineering Conference, ASME, 1982, pp. 39–42.
[2] J.B. Ransom, S.L. McCleary, M.A. Aminpour, N.F. Knight Jr., Computational methods for global/local analysis, NASA STI/Recon Technical Report N, vol. 92, 1992, p. 33104.
[3] N.G. Cormier, B.S. Smallwood, G.B. Sinclair, G. Meda, Aggressive submodelling of stress concentrations, Internat. J. Numer. Methods Engrg. 46 (6) (1999) 889–909.
[4] L. Gendre, O. Allix, P. Gosselet, F. Comte, Non-intrusive and exact global/local techniques for structural problems with local plasticity, Comput. Mech. 44 (2) (2009) 233–245.
[5] C.C. Jara-Almonte, C.E. Knight, The specified boundary stiffness/force SBSF method for finite element subregion analysis, Internat. J. Numer. Methods Engrg. 26 (7) (1988) 1567–1578.

[6] J.D. Whitcomb, Iterative global/local finite element analysis, Comput. Struct. 40 (4) (1991) 1027–1031.
[7] J.D. Whitcomb, K. Woo, Application of iterative global/local finite-element analysis. Part 1: linear analysis, Commun. Numer. Methods. Eng. 9 (1993) 745.
[8] F. Hecht, A. Lozinski, O. Pironneau, Numerical zoom and the Schwarz algorithm, in: Proceedings of the 18th Conference on Domain Decomposition Methods, 2009, pp. 63–73.
[9] P. Gosselet, M. Blanchard, O. Allix, G. Guguin, Non-invasive global-local coupling as a Schwarz domain decomposition method: acceleration and generalization, Adv. Model. Simul. Eng. Sci. 5 (4) (2018).
[10] P. Ladevèze, O. Loiseau, D. Dureisseix, A micro-macro and parallel computational strategy for highly heterogeneous structures, Internat. J. Numer. Methods Engrg. 52 (1–2) (2001) 121–138.
[11] M. Blanchard, O. Allix, P. Gosselet, G. Desmeure, Space/time global/local noninvasive coupling strategy: Application to viscoplastic structures, Finite Elem. Anal. Des. 156 (2019) 1–12.
[12] M. Duval, J.-C. Passieux, M. Salaün, S. Guinard, Non-intrusive coupling: recent advances and scalable nonlinear domain decomposition, Arch. Comput. Methods Eng. (2014) 1–22.
[13] M. Wangermez, O. Allix, P.-A. Guidault, O. Ciobanu, C. Rey, Non-intrusive global-local analysis of heterogeneous structures based on a second-order interface coupling, Comput. Mech. 69 (2022) 1241–1257.
[14] O. Allix, P. Gosselet, Non intrusive global/local coupling techniques in solid mechanics: An introduction to different coupling strategies and acceleration techniques, in: L.D. Lorenzis, A. Düster (Eds.), Modeling in Engineering using Innovative Numerical Methods for Solids and Fluids, in: CISM International Centre for Mechanical Sciences – Courses and Lectures, vol. 599, Springer Nature Switzerland AG, 2020, pp. 203–220.
[15] M. Chevreuil, A. Nouy, E. Safatly, A multiscale method with patch for the solution of stochastic partial differential equations with localized uncertainties, Comput. Methods Appl. Mech. Engrg. 255 (2013) 255–274.
[16] A. Nouy, F. Pled, A multiscale method for semi-linear elliptic equations with localized uncertainties and non-linearities, ESAIM Math. Model. Numer. Anal. (2018) 39.
[17] G. Guguin, O. Allix, P. Gosselet, S. Guinard, On the computation of plate assemblies using realistic 3D joint model: a non-intrusive approach, Adv. Model. Simul. Eng. Sci. 3 (16) (2016).
[18] D.E. Keyes, Aerodynamic applications of Newton-Krylov-Schwarz solvers, in: Fourteenth International Conference on Numerical Methods in Fluid Dynamics, Springer, 1995, pp. 1–20.
[19] P. Cresta, O. Allix, C. Rey, S. Guinard, Nonlinear localization strategies for domain decomposition methods: application to post-buckling analyses, Comput. Methods Appl. Mech. Engrg. 196 (8) (2007) 1436–1446.
[20] J. Hinojosa, O. Allix, P. Guidault, P. Cresta, Domain decomposition methods with nonlinear localization for the buckling and post-buckling analyses of large structures, Adv. Eng. Softw. 70 (2014) 13–24.
[21] C. Negrello, P. Gosselet, C. Rey, J. Pebrel, Substructured formulations of nonlinear structure problems — Influence of the interface condition, Internat. J. Numer. Methods Engrg. 107 (13) (2016) 1083–1105.
[22] O. Bettinotti, O. Allix, B. Malherbe, A coupling strategy for adaptive local refinement in space and time with a fixed global model in explicit dynamics, Comput. Mech. (2013) 1–14.
[23] O. Bettinotti, O. Allix, U. Perego, V. Oncea, B.t. Malherbe, A fast weakly intrusive multiscale method in explicit dynamics, Internat. J. Numer. Methods Engrg. 100 (8) (2014) 577–595.
[24] O. Bettinotti, O. Allix, U. Perego, V. Oncea, B.t. Malherbe, Simulation of delamination under impact using a global local method in explicit dynamics, Finite Elem. Anal. Des. 125 (2017) 1–13.
[25] D. Chazan, W. Miranker, Chaotic relaxation, Linear Algebr. Appl. 2 (1969) 199–222.
[26] J.-C. Miellou, Algorithmes de relaxation chaotiques à retard, ESAIM Math. Model. Numer. Anal. 9 (1975) 55–82.
[27] G.M. Baudet, Asynchronous iterative methods for multiprocessors, J. Assoc. Comput. Mach. 25 (2) (1978).
[28] M.N. El Tarazi, Some convergence results for asynchronous algorithms, Numer. Math. 39 (1982) 325–340.
[29] D. P.Bertsekas, Distributed asynchronous computation of fixed points, Math. Program. 27 (599) (1983) 107–120.
[30] E. Chow, A. Frommer, D. B.Szyld, Asynchronous richardson iterations: theory and practice, Numer. Algorithms 87 (2021) 1635–1651.
[31] P. Spiteri, J.-C. Miellou, D. El Baz, Asynchronous Schwarz alternating methods with flexible communication for the obstacle problem, Calc. Parallèles, Rés. et Syst. Répar. 13 (2001) 47–66.
[32] F. Magoulès, D. B. Szyld, C. Venet, Asynchronous optimized Schwarz methods with and without overlap, Numer. Math. 137 (2017) 199–227.
[33] I. Yamazaki, E. Chow, A. Bouteiller, J. Dongarra, Performance of asynchronous optimized Schwarz with one-sided communication, Parallel Comput. 86 (2019) 66–81.
[34] J. C.Garay, F. Magoulès, D. B.Szyld, Synchronous and asynchronous optimized Schwarz methods for Poisson's equation in rectangular domains, Electron. Trans. Numer. Anal. 55 (2022) 744–791.
[35] F. Magoulès, C. Venet, Asynchronous iterative sub-structuring methods, Math. Comput. Simulation 145 (2018) 34–49.
[36] G. Gbikpi-Benissan, F. Magoulès, Asynchronous substructuring method with alternating local and global iterations, J. Comput. Appl. Math. 393 (2021) 116–133.
[37] G. Gbikpi-Benissan, F. Magoulès, Resilient asynchronous primal Schur method, Appl. Math. 67 (2022) 679–704.
[38] J. Wolfson-Pou, E. Chow, Asynchronous multigrid methods, IEEE Int. Parallel Distributed Process. Symp. 149 (2020).
[39] P. Spiteri, Parallel asynchronous algorithms: A survey, Adv. Eng. Softw. 149 (2020) 102896.
[40] A. Frommer, D. B.Szyld, On asynchronous iterations, J. Comput. Appl. Math. 123 (2000) 201–216.
[41] L. Eisner, I. Koltracht, M. Neumann, Convergence of sequential and asynchronous nonlinear paracontractions, Numer. Math. 62 (1992) 305–319.

[42] P. Gosselet, C. Rey, Non-overlapping domain decomposition methods in structural mechanics, Arch. Comput. Methods Eng. 13 (4) (2007) 515–572.
[43] P. Le Tallec, Y.H. De Roeck, M. Vidrascu, Domain decomposition methods for large linearly elliptic three-dimensional problems, J. Comput. Appl. Math. 34 (1) (1991) 93.
[44] J. Mandel, Balancing domain decomposition, Commun. Numer. Methods. Eng. 9 (3) (1993) 233.
[45] N. Spillane, D.J. Rixen, Automatic spectral coarse spaces for robust FETI and BDD algorithms, Int. J. Numer. Methods Eng. 95 (11) (2013) 953–990.
[46] A. Klawonn, O. Widlund, FETI and Neumann-Neumann iterative substructuring methods: Connections and new results, Comm. Pure Appl. Math. 54 (1) (2001) 57–90.
[47] J.-C. Miellou, P. Spiteri, D. El Baz, A new stopping criterion for linear perturbed asynchronous iterations, J. Comput. Appl. Math. 219 (2008) 471–483.
[48] F. Magoulès, G. Gbikpi-Benissan, Distributed convergence detection based on global residual error under asynchronous iterations, IEEE Trans. Parallel Distrib. Syst. 29 (2018).
[49] E. Chow, A. Frommer, D. B.Szyld, AsynchronousRichardson iterations: theory and practice, Numer. Algorithms 87 (4) (2021) 1635–1651.
[50] I. Gohberg, P. Lancaster, L. Rodman, Matrix Polynomials, Society for Industrial and Applied Mathematics, 2009.
[51] K. Hirose, Continuity of the roots of a polynomial, Amer. Math. Monthly 127 (4) (2020) 359–363.
[52] A. Parusinski, A. Rainer, Optimal regularity of roots of polynomials, 2016, working paper or preprint.
[53] F. Magoulès, G. Gbikpi-Benissan, JACK2: An MPI-based communication library with non-blocking synchronization for asynchronous iterations, Adv. Eng. Softw. 119 (2018) 116–133.
[54] F. Magoulès, G. Gbikpi-Benissan, JACK: An asynchronous communication kernel library for iterative algorithms, J. Supercomput. 73 (8) (2017) 3468–3487.
[55] C. Glusa, E. Boman, E. Chow, S. Rajamanickam, D. B. Szyld, Sacalable asynchronous domain decomposition solvers, SIAM J. Sci. Comput. 42 (6) (2020) 384–409.
[56] Sweden ENCC, Intermediate MPI : One-sided communication concepts, 2020, https://enccs.github.io/intermediate-mpi/one-sided-concepts.
[57] L. Dalcin, Y.-L. L.Fang, Mpi4py: Status update after 12 years of development, Comput. Sci. Eng. 23 (4) (2021) 47–54.
[58] C. Geuzaine, J.-F. Remacle, Gmsh : a three-dimensional nite element mesh generator with built-in pre- and post-processing facilities, Internat. J. Numer. Methods Engrg. 79 (11) (2009) 1309–1331.
[59] Y. Renard, K. Poulios, GetFEM: Automated FE modeling of multiphysics problems based on a generic weak form language, Adv. Eng. Softw. 47 (2021) 1–31.