Acceleration of Frequency-Domain Full Wave Inversion through Krylov Reuse

Auteur : Gabriel, Tim
Promoteur(s) : Geuzaine, Christophe
Faculté : Faculté des Sciences appliquées
Diplôme : Master en ingénieur civil physicien, à finalité approfondie
Année académique : 2022-2023
URI/URL : https://gitlab.onelab.info/timgabriel/ddm-master-thesis-tim-gabriel; http://hdl.handle.net/2268.2/18323

University of Liège
Faculty of applied sciences
Department of Electrical Engineering and Computer Science

Acceleration of Frequency-Domain Full Waveform Inversion through Krylov Reuse

Master's thesis conducted by Tim Gabriel
in the pursuit of obtaining the Master's Degree in Engineering Physics.

Academic advisor:
Prof. Christophe Geuzaine

Author:
Tim Gabriel

acedemic year 2022–2023

Acknowledgements

I want to sincerely thank Professor Christophe Geuzaine for proposing the engaging topic of my master's thesis. I'm truly grateful for the chance to explore such an intriguing subject under his mentorship. I also want to express my sincere thanks to both Professor Geuzaine and Boris Matin for their constant support and valuable advice. Our regular meetings were very enriching and provided me with all the necessary information to progress with my thesis. Their feedback greatly shaped my work, and collaborating with them has been a rewarding experience. Finally, I would like to express my appreciation to the jury members: Professor Maarten Arnst, Professor Frédéric Nguyen, and Boris Martin, for their agreement to evaluate my work.

Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11 and by the Walloon Region.

Contents

Introduction

The resolution of a sequence of linear problems with different right-hand sides arises in various physics applications. While the resolution of such a sequence is easily handled by direct solvers through the reuse of factorization, the same cannot be said for iterative methods. In domain decomposition methods, solving an interface problem is crucial to attain a global solution. The resolution of this problem can leverage matrix-free approaches, including certain iterative methods, to enhance the overall efficiency of the method. Therefore, when addressing challenges like full waveform inversion using domain decomposition, the necessity emerges for an efficient resolution through iterative methods of a sequence of problems with varying right-hand sides. This would then significantly reduce computational load of the inversion. In [1], potential is shown in using information from diverse problem resolutions with the Orthodir iterative method. Thus, this work focuses on reusing information during the resolution of linear problems with varying right-hand sides, using different Krylov iterative solvers.

From this perspective, Chapter 1 introduces the integration of the full waveform inversion framework into the domain of subsurface tomography. This methodology necessitates addressing wave propagation, and the mathematical foundation to tackle such a challenge is presented. The necessity to resolve a series of problems for conducting an inversion is unveiled, along with the associated formulation of these problems. Subsequently, details concerning the numerical resolution of such problems are provided, leading to the transformation of this sequence into linear problems with distinct right-hand sides. Following this, the domain decomposition method is introduced, accompanied by an exploration of the motivations underlying the adoption of iterative methods.

Then, in Chapter 2, a comprehensive presentation of Krylov subspace iteration methods is provided, with a specific focus on minimization solvers. This encompasses three types of solvers: the GCR, GMRES, and SGMRES methods. Additionally, the concept of residual based algorithms is introduced, leading to variations in certain algorithms. Furthermore, the key characteristics of these solvers are presented and compared. This will offer guidance on the preferable methods for addressing a specific linear problem, while also providing the essential knowledge for the upcoming chapter.

In Chapter 3, the extension of the Krylov iteration method is presented to enable the efficient resolution of sequences of linear problems with different right-hand sides. The presentation of augmentation methods will then be introduced to incorporate additional information for improving the convergence of iterative methods. Subsequently, the application of these augmentation methods to the reuse of information from preceding resolutions, referred to as subspace recycling techniques, will be discussed. The comparison of the different iterative methods under consideration will once more be carried out. Additionally, a method for selecting only a portion of the reused information will be introduced, with the goal of reducing the

storage requirements of the methods.

In Chapter 4, the practical application of subspace recycling iterative solvers is explored within the context of two-dimensional wave scattering problems using domain decomposition. The impact of the subspace recycling method will be presented, along with an examination of how different simulation parameters affect the results. Additionally, experiments involving partial subspace recycling will be conducted, employing various selection strategies. Lastly, the implications of choosing different iterative methods will be discussed in relation to the domain decomposition framework.

Chapter 1

Full waveform inversion context

Scattering of waves is an omnipresent phenomenon that occurs when waves encounter heterogeneities in physical systems. These heterogeneities can take many forms, such as obstacles, boundaries or fluctuations in the medium's properties. When waves encounter these variations, they interact with them, leading to possible changes in the wavefield direction, amplitude, phase and polarization. These changes can result in complex wavefields that contain valuable information about the properties of the medium and the heterogeneities within it. Scattering waves take place with mechanical waves, electromagnetic waves and wavelike particles. Therefore, their applications are diverse and span multiple fields of science and engineering, such as astronomy, material characterization, medical imaging, microscopy, etc.

In geophysics, scattering waves are of particular interest due to their essential role in the imaging of subsurface geological features. Using excitation sources and receivers at the surface, the information in the resulting waveform can be extracted and analyzed to obtain a high resolution representation of the subsurface structures. More generally, waveform tomography corresponds to the inversion of a wave scattering problem to image an object or a region of interest in a non-intrusive manner. The inversion is achieved by minimizing the mismatch between the measured data and the outcomes from an approximate modelization of the underground. This optimization requires a significant amount of wave propagation simulations referred as the forward problem. The development of full waveform inversion (FWI) and the increased computational capabilities have allowed waveform tomography with unprecedented resolution and accuracy, making FWI a popular tool in many areas including geophysics.

However, the high computational cost and lengthy processing times required for FWI remain a challenge and ongoing efforts are focused on developing more efficient algorithms and techniques to speed up the inversion process and improve the imaging quality. In order to achieve high-performance inversion, it is vital to effectively manage the forward problems that arise during the inversion process. These forward problems involve diverse source terms, and here the focus will be on finding the time harmonic solutions to these wave scattering. The resolution of these problems will typically be achieved using a combination of the finite element method and the domain decomposition method. In addition, some information could be reused from one resolution to another, which is the subject of the current work.

1.1 Full waveform inversion

In the doctoral dissertation by X. Adriaens [2], an extensive examination of full waveform inversion is presented across three distinct fields: acoustic, electromagnetic, and elastodynamic. This current study builds upon this work, with a specific focus on the acoustic aspect in the context of marine seismology surveys, i.e. the imaging of geological structures under the seabed (Figure 1.1). However, the extension to other fields and applications would be natural.

The initial and crucial phase of the full waveform inversion is to gather data from wave scattering at the surface of the targeted domain. Sound waves are produced through excitation sources and recorded by hydrophones. Both the emission and reception equipment are positioned in the water above the seafloor, necessitating precise placement and calibration to effectively capture extensive information about the medium's response to the excitations. The effectiveness and precision of the FWI procedure are significantly impacted by the quality of the acquired data. Therefore, this first stage requires special considerations.

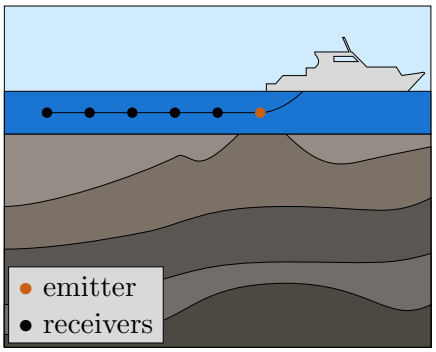


Figure 1.1: Representation of a marine seismology survey.

The next stage is to find an initial approximate model of the subsurface structures by the means of their physical properties. This model corresponds to a discretization of the studied domain with the associated physical properties, such as the velocity of acoustic waves. It is typically achieved by using existing knowledge of the geology and geophysics of the area, as well as relying on previous exploration reports with other imaging methods, e.g. traveltime tomography [3]. This approximate model is then used to generate synthetic waveforms by solving the wave scattering problem. The synthetic and observed waveforms are then compared and their mismatch is quantified using an objective function, e.g. the squared differences of the wave fields at the receivers. This function is then minimized through an iterative optimization process over the model parameters. At each iteration, the model is refined and the fit between the synthetic and observed waveforms is improved. Hopefully, after enough iterations, the model gives an accurate representation of the physical properties.

The difficulties remains in the optimization process since the misfit function must be appropriate for wave scattering fields, the solution might not be unique and nothing can guarantee that it corresponds to the actual field. For more information about this optimization, see [2]. To facilitate the optimization process, local optimization techniques will be used. These methods require both the present value of the objective function and the related gradients with respect to the parameters of the model. This approach aims then to enhance the misfit function, denoted as $\mathcal{J}(m)$, in relation to these model parameters m. The optimization process can then be formulated as the search for optimal parameters m^* such that

$$m^* = \arg\min_m \mathcal{J}(m), \tag{1.1}$$

at least locally. Therefore, it is essential to have the capability to compute not only the misfit function but also the sensitivity of this function in response to variations in the model parameters. Fortunately using the adjoint state method (see Section 1.2.3), the sensitivity can be easily computed by solving adjoint problems which are similar to the direct wave scattering but with different source terms. Altogether, one forward problem involves computing the value

and the sensitivity of the misfit function by resolving one direct and one adjoint problem per emitter and per frequency. Therefore, the most computationally expensive part of the full waveform inversion process is solving wave scattering problems with different source terms, and this explains the need for efficient resolution techniques.

1.2 Forward problem

The aim of a forward problem is to compute the value and the sensitivity of the misfit function by solving wave propagation problems. These can be carried out in the time domain, however research in [4] suggests that the frequency domain is better suited for tomographic inversion. One of the key advantages of time-harmonic representation is that wave propagation can be modeled efficiently for multiple sources problems. Another advantage of inversion in the frequency domain, is the possibility to limit the number of forward models to a reduced number of frequency components without affecting significantly the accuracy of the image. On top of a time-harmonic representation of wave scattering, appropriate boundary conditions must be imposed. Indeed, numerical simulations are typically conducted in a finite space and therefore the studied domain is restricted. The boundary conditions are important because they ensure that the simulated system behaves in a physically realistic manner. Once the direct scattering problem is defined, the adjoint state method can be used to derive the adjoint problem and then compute efficiently the sensitivity with respect to the parameter variations.

1.2.1 Time-harmonic wave propagation

The wave equation provides a mathematical description of how waves propagate through different media. It captures the complexity of wave scattering, including the effects of reflection, refraction, diffraction and interference. Solving the wave equation enables the determination of how the wave evolves in both space and time. It is therefore a powerful tool for analyzing and understanding wave phenomena.

In the case of acoustic wave propagation, this equation takes the form of a partial differential equation that describes the behavior of a complex acoustic field $U(\boldsymbol{x}, t)$ representing the acoustic pressure (or shear) and phase of the wave. It can be derived for liquid and solids from the Newton's second law, providing elastic behavior, etc [5]. The wave equation relates then the second-order time and space derivatives of the acoustic field U with source terms represented by a function F and in the constant density approximation it writes as

$$\frac{\partial^2 U(\boldsymbol{x}, t)}{\partial t^2} - c^2(\boldsymbol{x})\Delta U(\boldsymbol{x}, t) = F(\boldsymbol{x}, t), \tag{1.2}$$

where Δ is the Laplace operator and $c(\boldsymbol{x})$ is the speed of the wave propagation which is a property of the medium. All sources can be decomposed as a sum of harmonic contributions and using the Fourier transform this wave equation can be expressed in the frequency domain as

$$\Delta u(\boldsymbol{x}, \omega) + \omega^2 s^2(\boldsymbol{x}) u(\boldsymbol{x}, \omega) = -s^2(\boldsymbol{x}) f(\boldsymbol{x}, \omega). \tag{1.3}$$

This last equation is the Helmholtz equation and it gives the spatial representation of the wave through the complex field $u(\boldsymbol{x}, \omega)$ with the angular frequency ω. The slowness of the medium is defined as $s(\boldsymbol{x}) = 1/c(\boldsymbol{x})$, while the associated wave number is given by $k(\boldsymbol{x}) = \omega s(\boldsymbol{x})$. The sources of excitation are represented as sources terms $f(\boldsymbol{x}, \omega)$ for each angular frequency ω. Once the Helmholtz equation (1.3) has been solved, the time-dependent solution of wave

propagation can be obtained by combining the spatial and temporal components of the wave for all angular frequencies:

$$U(\boldsymbol{x}, t) = \int_{\omega} u(\boldsymbol{x}, \omega) e^{-i\omega t} d\omega. \tag{1.4}$$

Here, the symbol i denotes the imaginary unit ($i = \sqrt{-1}$). Therefore, in the context of acoustic waves, the pressure field will be given by the real part of this field $U(\boldsymbol{x}, t)$.

1.2.2 Boundary conditions

The choice of boundary conditions (BC) is a critical aspect when solving wave propagation problems. In many cases, the boundaries of the computational domain do not correspond to physical boundaries of the system, but rather delimit the region of interest for the simulation. Therefore, the choice of BC must be appropriate to accurately represent the physical behavior of the waves at the domain boundaries. In wave propagation, the presence of a boundary can introduce reflections of incident waves. The resulting solution may no longer correspond to the physical problem being studied. Therefore, it is important to prevent too much reflections of waves from the fictitious boundaries. To address this issue, various types of absorbing boundary conditions (ABC) have been developed to minimize or eliminate reflections from the boundaries.

One commonly used ABC for wave propagation problems is the Sommerfeld radiation condition [6], which models the behavior of waves in unbounded space by requiring that the wave field satisfy a certain decay rate at infinity. This condition effectively "absorbs" the outgoing waves at the boundaries and prevents them from reflecting back into the computational domain. For two dimensions, it is expressed as

$$\lim_{r\to\infty} \sqrt{r}\big(\nabla_r u - i\omega s u\big) = 0. \tag{1.5}$$

where r represent the distance from the origin and ∇_r is the directional derivative with respect to r. To use this boundary condition on a finite size domain, one can approximate it by applying the condition on the boundary instead, thus defining a Sommerfelt-like absorbing boundary condition:

$$\partial_n u - i\omega s u = 0 \quad \text{on the boundary}, \tag{1.6}$$

where $\partial_n = \boldsymbol{n} \cdot \nabla$, $\boldsymbol{n}$ is the outward normal vector to the domain and ∇ is the gradient operator. This Sommerfeld-like ABC is an efficient and easy to implement absorbing boundary condition. Higher order approximation of (1.5) on the boundary exist [7, 8] and other solutions such as perfectly match layers [9, 10] have been developed to replicate this absorbing boundary condition. These can be used to reduce even more the unwanted reflections if they lead to nonphysical results. The simple Sommerfeld-like absorbing boundary condition will be considered here.

1.2.3 Adjoint state method

The adjoint state method is commonly used in optimization and inversion problems to efficiently compute the sensitivities of the objective functions with respect to the parameter variations. For each objective function, adjoint variables are found by solving an adjoint problem. Then, using the state and the adjoint variables together, it is possible to easily compute the sensitivity of the corresponding objective function with respect to any possible parameters variations. A general derivation of the adjoint state methods has been conducted for different time-harmonic scattering problems in [11] and similar notations will be used.

In the context of waveform tomography, the model parameters m correspond to the slowness squared $s^2(\boldsymbol{x})$ of the wave, but the general notation m is kept. Also, the misfit function for a given emitter e is defined by the difference between the simulated field u_e and the data collected by the receivers $d_{e,r}$ at the positions of the receivers $\boldsymbol{x}_r$. For an angular frequency ω, a typical objective function $\mathcal{J}$ for the inversion is given as

$$\mathcal{J}(m) = \frac{1}{2}\sum_e \sum_r |u_e(\boldsymbol{x}_r) - d_{e,r}|^2 = \sum_e \mathcal{H}_e(m). \tag{1.7}$$

where the dependance on the model parameters m is due to the state fields u_e. The optimization procedure remains the same for all emitters since each one corresponds to a state field u_e and an objective function $\mathcal{H}_e$. Therefore, the optimization for only one emitter will be considered in the following.

For one emitter, the state field u is found by solving the Helmholtz equation (1.3), on the considered domain Ω with appropriate boundary conditions (1.6) on the domain boundary $\partial\Omega$. If the excitation source is situated at the position of the emitter $\boldsymbol{x}_e$ and is of amplitude $c^2(\boldsymbol{x}_e)$, the direct problem writes as

$$\begin{cases} \mathcal{F}_e(u_e, m) = \Delta u_e + \omega^2 s^2 u_e = -\delta(\boldsymbol{x} - \boldsymbol{x}_e) & \text{in } \Omega, \\ \mathcal{B}_e(u_e, m) = \partial_n u_e - i\omega s u_e = 0 & \text{on } \partial\Omega, \end{cases} \tag{1.8}$$

where $\mathcal{F}_e$ and $\mathcal{B}_e$ are compact notations for bulk and boundary operators and δ is the delta Dirac function defined on the domain. Then, the sensitivity of the objective function $\mathcal{H}_e$ can be expressed as the directional derivative or Gâteaux derivative:

$$\{D_m \mathcal{H}_e(m)\}(\delta m) = \lim_{\epsilon \to 0} \frac{\mathcal{H}_e(m + \epsilon\delta m) - \mathcal{H}_e(m)}{\epsilon} \quad \forall \delta m, \tag{1.9}$$

where D_m is the derivation operator w.r.t. the parameters and δm is a given perturbation of the parameters. A more compact representation is possible using a Fréchet-Wirtinger's gradient kernel, or simply gradient kernel, denoted j' and defined as

$$\{D_m \mathcal{H}_e(m)\}(\delta m) = \mathrm{Re}\langle j'_e, \delta m\rangle \quad \forall \delta m, \tag{1.10}$$

where $\mathrm{Re}\langle ., .\rangle$ is the real part of the inner product in the parameter space. Indeed, a gradient kernel is an element of the parameter space where each value is the variation of $\mathcal{H}_e$ for an arbitrary small variation of the corresponding parameter. The aim will therefore be to find such gradient kernels.

Direct approach

The direct approach consist in using the definition of the objective function (1.7) to find its directional derivative with respect to m. Since the misfit function is a function of the parameters only through the state field u_e, the sensitivity is

$$\begin{aligned} \big\{D_m \mathcal{H}_e\left[u_e(m)\right]\big\}(\delta m) = \{D_u \mathcal{H}_e\}(\delta u_e) &= \frac{1}{2}\sum_r \Big[\overline{\big(u_e(\boldsymbol{x}_r) - d_r\big)}\delta u_e(\boldsymbol{x}_r) + \big(u_e(\boldsymbol{x}_r) - d_r\big)\overline{\delta u_e(\boldsymbol{x}_r)}\Big] \\ &= \mathrm{Re}\sum_r \overline{\big(u_e(\boldsymbol{x}_r) - d_r\big)}\delta u_e(\boldsymbol{x}_r), \end{aligned} \tag{1.11}$$

where $\overline{(.)}$ represent the complex conjugate. The variation $\delta u_e(\boldsymbol{x}_r)$ can be decomposed using the identity $\delta u_e(\boldsymbol{x}_r) = \int_\Omega \delta u_e \delta(\boldsymbol{x} - \boldsymbol{x}_r) d\boldsymbol{x}$, where the second δ is a delta Dirac function. The

sensitivity becomes

$$\{D_m\mathcal{H}_e\}(\delta m) = \mathrm{Re}\int_\Omega \sum_r \overline{\big(u_e(\boldsymbol{x}_r) - d_r\big)}\delta(\boldsymbol{x}-\boldsymbol{x}_r)\delta u_e d\boldsymbol{x} \tag{1.12}$$

Similarly to the definition of a gradient kernel in (1.10), the gradient kernel h'_e can be defined but relative to the variation of the state field δu_e:

$$\{D_m\mathcal{H}_e\}(\delta m) = \mathrm{Re}\langle h'_e, \delta u_e\rangle \qquad \text{with} \quad h'_e = \sum_r \big(u_e(\boldsymbol{x}_r) - d_r\big)\delta(\boldsymbol{x}-\boldsymbol{x}_r). \tag{1.13}$$

Unfortunately, in order to recover the sensitivity relative to the parameter variation δm, the corresponding variation of the state field δu_e must be computed. This is done by solving, for δu_e, the system resulting from the total variation of the direct problem (1.8) in the direction δm:

$$\begin{cases} \{\partial_{u_e}\mathcal{F}_e(u_e,m)\}(\delta u_e) = -\{\partial_m\mathcal{F}_e(u_e,m)\}(\delta m), \\ \{\partial_{u_e}\mathcal{B}_e(u_e,m)\}(\delta u_e) = -\{\partial_m\mathcal{B}_e(u_e,m)\}(\delta m). \end{cases} \tag{1.14}$$

Then, for each emitter, there are as much systems to solve as there are parameters. This is a huge amount of computation since the parameters are here the slowness squared of the wave $s^2(\boldsymbol{x})$ defined on each element of the discretization of the domain.

Adjoint approach

On the other hand, the adjoint method is based on the definition of an adjoint field $u^\dagger$ and the associated adjoint operator $\partial_u^\dagger\mathcal{F}(u,m)$ such that the following relationship is respected:

$$\mathrm{Re}\left\langle u^\dagger, \{\partial_u\mathcal{F}(u,m)\}(\delta u)\right\rangle = \mathrm{Re}\left\langle \{\partial_u^\dagger\mathcal{F}(u,m)\}(u^\dagger), \delta u\right\rangle + \left[u^\dagger, \delta u\right]_{\partial_u\mathcal{F}}, \tag{1.15}$$

where $\left[u^\dagger, \delta u\right]_{\partial_u\mathcal{F}}$ is a boundary term which vanishes since the objective function does not depend on the boundary, see [11]. The adjoint operator is then $\partial_u^\dagger\mathcal{F} = \overline{\partial_u\mathcal{F}}$ and similarly on the boundary operator $\partial_u^\dagger\mathcal{B} = \overline{\partial_u\mathcal{B}}$. Also, the adjoint field $u^\dagger$ is defined as the solution to

$$\begin{cases} \{\partial_u^\dagger\mathcal{F}(u,m)\}(u^\dagger) = h' & \text{in } \Omega, \\ \{\partial_u^\dagger\mathcal{B}(u,m)\}(u^\dagger) = 0 & \text{on } \partial\Omega. \end{cases} \tag{1.16}$$

Using the definition of the adjoint state field in combination to the adjoint operator, the sensitivity given in (1.12) becomes

$$\begin{aligned} \{D_m\mathcal{H}\}(\delta m) &= \mathrm{Re}\langle h', \delta u\rangle = \mathrm{Re}\langle\{\partial_u^\dagger\mathcal{F}\}(u^\dagger), \delta u\rangle \\ &= \mathrm{Re}\left\langle u^\dagger, \{\partial_u\mathcal{F}\}(\delta u)\right\rangle = -\mathrm{Re}\left\langle u^\dagger, \{\partial_m\mathcal{F}\}(\delta m)\right\rangle. \end{aligned} \tag{1.17}$$

The partial derivative of the bulk operator $\partial_m\mathcal{F}$ with respect to the parameter m is straightforward by making use of the operator definition in (1.8). Eventually, the sensitivity can be expressed by a gradient kernel j' as

$$\{D_m\mathcal{H}\}(\delta m) = -\mathrm{Re}\left\langle u^\dagger, \omega^2 u\delta m\right\rangle = \mathrm{Re}\left\langle j', \delta m\right\rangle \qquad \text{with} \quad j' = -\omega^2\overline{u}u^\dagger. \tag{1.18}$$

If all emitters are considered again, it is then possible to efficiently compute the sensitivity of the objective function $\mathcal{J}$ for any parameter variation $\delta m = \delta s^2(\boldsymbol{x})$. Firstly, the state field u_e for each emitter must be found by solving the direct problems (1.8) represented as

$$\begin{cases} \Delta u_e + \omega^2 s^2 u_e = -\delta(\boldsymbol{x}-\boldsymbol{x}_e) & \text{in } \Omega, \\ \partial_n u_e - i\omega s u_e = 0 & \text{on } \partial\Omega. \end{cases} \tag{1.19}$$

Secondly, the complex conjugate of the adjoint state field $\overline{u}_e^\dagger$ must be found by solving the adjoint problems (1.16) represented as

$$\begin{cases} \Delta\overline{u}_e^\dagger + \omega^2 s^2 \overline{u}_e^\dagger = \sum_r \left(\overline{u}_e(\boldsymbol{x}_r) - \overline{d}_r\right)\delta(\boldsymbol{x} - \boldsymbol{x}_r) & \text{in } \Omega, \\ \partial_n \overline{u}_e^\dagger - i\omega s \overline{u}_e^\dagger = 0 & \text{on } \partial\Omega. \end{cases} \tag{1.20}$$

Thirdly, the complex conjugate of the gradient kernel $\overline{j}'_{tot}$ is computed and finally the sensitivity to any parameter variation δs^2 can be found using

$$\overline{j}'_{tot} = -\omega^2 \sum_e u_e \overline{u}_e^\dagger \qquad \text{and} \qquad \{D_{s^2}\mathcal{J}\}(\delta s^2) = \operatorname{Re}\left\langle j'_{tot}, \delta s^2 \right\rangle. \tag{1.21}$$

In the end, by applying the adjoint state method, the waveform inversion only requires the resolution of one direct problem and one adjoint problem per emitter, for each frequency considered. This makes the adjoint state method a powerful tool for inversion. Also, both the direct and adjoint problems are wave propagation problems and the only difference between them lies in the sources terms. Therefore, efficient methods can be used to take advantage of solving multiple scattering wave problems with different sources.

1.3 Numerical simulation

The essence of full waveform inversion is centered around effectively resolution of wave scattering problems. For these resolutions, numerical simulations will be employed given their capability to handle nonlinear behavior and accommodate complex geometries. The finite element method is commonly employed approach to solve boundary value problems. By applying an appropriate meshing technique to the computational domain, the finite element method transforms the original problem into a square system of linear equations. This system is represented by a linear sparse matrix problem and must be solved by a direct or iterative method.

The exploration of variances between direct and iterative solvers is undertaken in Chapter 2. However, the primary characteristics can be outlined as follows. Direct methods encounters the difficulty of fill-in when dealing with sparse matrices, which leads to significant memory usage to handle large systems. However, they offer the benefit of quickly solving small systems and are also well-suited for efficiently handling problems with multiple right-hand sides. Conversely, iterative methods might require less memory storage at the cost of a potentially slower resolution due to the possible large number of iterations required to converge. Additionally, iterative methods are generally not effective when dealing with systems involving multiple right-hand sides, particularly when these right-hand sides are not all available simultaneously. Indeed, the entire iteration process must be carried out for each individual right-hand side problem if no information is shared between the resolutions. In Chapter **??**, methods for enhancing the convergence of the iterative approach applied to a sequence of right-hand sides will be explored. This approach enables local resolutions of the problem through a rapid direct method while addressing a larger global problem using an iterative solver. Another important advantage of the domain decomposition method is its ability to capitalize on distributed computing.

1.3.1 Finite element method

The finite element method (FEM) is a powerful numerical technique to solve boundary value problems, i.e. partial differential equations associated with boundary conditions. A detailed description of this method is provided in [12]. In accordance with Section 1.2, for

a given angular frequency ω, the task at hand is to solve the Helmholtz equation (1.3) on a domain Ω. The different source terms are represented by $f(\boldsymbol{x}, \omega)$ and the absorbing boundary condition (1.6) is applied on the domain boundary $\partial\Omega$. Using test functions $u'(\boldsymbol{x})$ defined in Sobolev space $H^1(\Omega) = \{u' \in L^2(\Omega) \text{ such that } \nabla u' \in (L^2(\Omega))^3\}$, the problem can be rewritten in its weak formulation:

$$\begin{cases} \text{Find } u \in H^1(\Omega) \text{ such that for all } u' \in H^1(\Omega): \\ -\displaystyle\int_\Omega \nabla\overline{u}' \cdot \nabla u \, d\Omega + \int_\Omega \overline{u}' \omega^2 s^2 u \, d\Omega + \int_{\partial\Omega} \overline{u}' i\omega s u \, d\partial\Omega = -\int_\Omega \overline{u}' s^2 f \, d\Omega. \end{cases} \tag{1.22}$$

Next, the domain is discretized into finite elements and shape functions are chosen with compact support. In addressing high-frequency problems characterized by oscillating solutions, the utilization of high-order basis functions becomes imperative. Among the potential approaches, hierarchical basis functions stand out as a viable choice [13]. This discretization process enables the approximation of the solution $u(\boldsymbol{x})$ through a linear combination of shape functions $N_i(\boldsymbol{x})$ as $u(\boldsymbol{x}) = \sum_i q_i N_i(\boldsymbol{x})$ where q_i are the corresponding coefficients. Also, the shape functions can be used as test functions $u'(\boldsymbol{x})$, thus leading to the Galerkin method. This allows the weak form (1.22) to be expressed as a square system of linear equations where the unknowns are the coefficients q_i. Providing an appropriate definition of the system matrix $\mathbf{K}$ and the right hand side vector $\mathbf{g}$, see [12], the problem reduces to finding the vector $\mathbf{q} = [q_1 \ q_2 \ \cdots]^T$ satisfying the square linear system $\mathbf{Kq} = \mathbf{g}$. Since the shape functions have compact support, the resulting matrix $\mathbf{K}$ is sparse. This linear system is then solved using direct or iterative methods, depending on the size of the matrix.

However, applying the finite element method directly on large problems, such as high frequency wave scattering, can result in solving huge sparse systems. Therefore, direct solvers can not be used as they don't scale well. On the other hand, iterative solvers are not a satisfying solution neither since they may converge too slowly or even diverge [14]. In this context, domain decomposition methods offer an alternative approach to handle large boundary value problems.

1.3.2 Domain decomposition method

The domain decomposition method (DDM) is a numerical technique to solve large-scale boundary value problems. The method is based on the idea of dividing the computational domain into a set of subdomains, each of which is solved independently. The solutions obtained from the subdomains are then coordinated in order to have matching solutions between adjacent subdomains. Hereafter, the solutions are combined to obtain the solution for the entire domain. This method enables to leverage the efficiency of direct solvers on small subproblems. Moreover, the framework of the domain decomposition method allows to easily exploit the power of distributed computing by solving the subproblems on multiple processors simultaneously. The problem of matching the subdomains solutions can also be cast into a square linear problem, however only the matrix-vector product is easily available and therefore iterative methods handling matrix-free resolutions are required. In this context, the substructured optimized Schwarz method will be employed, as it has demonstrated its efficacy in addressing time-harmonic high-frequency wave scattering problems [15]. A comprehensive review of optimized Schwarz method for time-harmonic wave problems with different transmission conditions is done in [16] and similar notations will used.

One possibility for the choice of subdomains is to have non-overlaping subdomains such that adjacent domains only share their interface. The domain Ω is then decomposed in n_{dom} subdomains Ω_i and the boundaries of the subdomains are noted $\partial\Omega_i$ with $i \in \{0, \ldots, n_{dom}-1\}$.

The domain boundaries are given by $\Sigma_i^\infty = \partial\Omega_i \cap \partial\Omega$ and the transmission boundaries between subdomains are represented as $\Sigma_{ij} = \partial\Omega_i \cap \partial\Omega_j$ for all $j \in \{0, \dots, n_{dom} - 1\}$. On these transmission boundaries, the continuity between subdomains is ensured by

$$\partial_{n_i} u_i + \mathcal{S} u_i = \partial_{n_i} u_j + \mathcal{S} u_j \qquad \text{on } \Sigma_{ij}, \tag{1.23}$$

where $\mathcal{S}$ is a general notation for all possible transmission conditions. This continuity results in fields g_{ij} on the interfaces Σ_{ij}. If g correspond to all interface unknowns $(g_{ij})_{\forall i,\forall j}$, the volume problem corresponds to a boundary value problem in the subdomain Ω_i and is referred as $u_i = \mathcal{V}_i(f, g)$:

$$\begin{cases} \Delta u_i + \omega^2 s^2 u_i = -s^2 f & \text{in } \Omega_i, \\ \partial_{n_i} u_i - i\omega s u_i = 0 & \text{on } \Sigma_i^\infty, \\ \partial_{n_i} u_i + \mathcal{S} u_i = g_{ij} & \text{on } \Sigma_{ij}, \quad \forall j. \end{cases} \tag{1.24}$$

The interface problem match the interface fields such as in (1.23) and it is referred as $g_{ji} = \mathcal{J}_{ji}(g_{ij}, u_i)$:

$$g_{ji} = -\partial_{n_i} u_i + \mathcal{S} u_i = -g_{ij} + 2\mathcal{S} u_i \qquad \text{on } \Sigma_{ij}. \tag{1.25}$$

In the volume problems (1.24), the sources f are referred as physical sources since they directly come from the definition of the problem. By opposition, the interface fields g_{ij} are artificial sources introduced by the DDM construction.

In order to find a solution to (1.24), both the physical and artificial sources are needed, however by linearity of the volume problem the solution can be decomposed in $u_i = v_i + \tilde{u}_i$ with $v_i = \mathcal{V}_i(f, 0)$ and $\tilde{u}_i = \mathcal{V}_i(0, g)$ in each Ω_i. The quantity v_i can be computed without the knowledge of the interface fields. This is not the case of $\tilde{u}_i$ which must be evaluated together with the interface fields g. It can result in an iterative process which updates these fields at the n^{th} iteration by

1. Solving the volume problems for $\tilde{u}_i^n$ with previous interface fields: $u_i^n = \mathcal{V}_i(0, g^n)\ \forall i$.
2. Finding the interface fields g_{ji}^{n+1} from the volume solutions and previous interface fields: $g_{ji}^{n+1} = \mathcal{J}_{ji}(g_{ij}^n, \tilde{u}_i^n) + 2\mathcal{S} v_i\ \forall i$ and $\forall j$.

If the vector $b = (b_{ji})_{\forall i,\forall j}$ is defined as the vector of all terms $b_{ji} = 2\mathcal{S} v_i$ on Σ_{ij}, and $\mathcal{A}$ is the transformation $g^n \to \mathcal{A} g^n$ such that $(\mathcal{A} g^n)_{ij} = \mathcal{J}_{ji}(g_{ij}^n, \tilde{u}_i^n)$ where $\tilde{u}_i^n = \mathcal{V}_i(0, g^n)$, then the iterative process reduces to

$$g^{n+1} = \mathcal{A} g^n + b. \tag{1.26}$$

This correspond to one iteration of the Jacobi method applied to the system

$$\big(\mathcal{I} - \mathcal{A}\big) g = b, \tag{1.27}$$

where $\mathcal{I}$ is the identity operator and the dependance on $\tilde{u}_i$ is hidden in operator $\mathcal{A}$. As suggested in [16] and [15], (1.27) can be solved by any iterative solver. More precisely, the iterative solver should only use matrix-vector multiplication (matrix-free resolution) because the explicit construction of the operator $\mathcal{I} - \mathcal{A}$ would require as much resolutions of $\tilde{u}_i = \mathcal{V}_i(0, g)$ as there are degrees of freedom in g while one matrix-vector product $(\mathcal{I} - \mathcal{A})g$ only require one resolution for $\tilde{u}_i$. Once the solution for the interface quantities g are found, it is then possible to obtain the solution for the whole domain u by solving one last time the volume problems: $u_i = \mathcal{V}_i(f, g)$.

In [16], a wide variety of transmission conditions is given, but here the simple zero-order impedance boundary condition will be used: $\mathcal{S} = -i\omega s$. This condition has the same structure

as the absorbing boundary condition applied on $\partial\Omega$, hence the notations will be simplified. To solve the volume and interface problems, the finite element method as described in Section 1.3.1 will be use and thus weak formulations of the problems must be developed. Volume test functions $u'_i \in H^1(\Omega_i)$ and surface test functions $g'_{ji} \in H^1(\Sigma_{ij})$ are accordingly defined for all i and j. The domain decomposition method can be summarized as

1. Construct the right-hand side b for all $b_{ij} = 2i\omega s v_i$ on Σ_{ij} by solving for the physical contributions $v_i = \mathcal{V}_i(f, 0)$ with the volume weak formulation

$$\begin{cases} \text{Find } v_i \in H^1(\Omega_i) \text{ such that, for every } u' \in H^1(\Omega_i): \\ -\displaystyle\int_{\Omega_i} \nabla\overline{u}' \cdot \nabla v_i \, d\Omega_i + \int_{\Omega_i} \overline{u}'\omega^2 s^2 v_i \, d\Omega_i \\ \qquad\qquad + \displaystyle\int_{\partial\Omega_i} \overline{u}' i\omega s v_i \, d\partial\Omega_i = -\int_{\Omega_i} \overline{u}' s^2 f \, d\Omega_i. \end{cases} \tag{1.28}$$

2. Solve iteratively $(\mathcal{I} - \mathcal{A})g = b$ by using the matrix-vector product $(\mathcal{I} - \mathcal{A})g^n$ at the iteration n. This product is constructed by first solving for $\tilde{u}_i^{n+1} = \mathcal{V}_i(0, g^n)$

$$\begin{cases} \text{Find } \tilde{u}_i^n \in H^1(\Omega_i) \text{ such that, for every } u' \in H^1(\Omega_i): \\ -\displaystyle\int_{\Omega_i} \nabla\overline{u}' \cdot \nabla \tilde{u}_i^n \, d\Omega_i + \int_{\Omega_i} \overline{u}'\omega^2 s^2 \tilde{u}_i^n \, d\Omega_i \\ \qquad\qquad + \displaystyle\int_{\partial\Omega_i} \overline{u}' i\omega s \tilde{u}_i^n \, d\partial\Omega_i = \sum_j \int_{\Sigma_{ij}} \overline{u}' g_{ij}^n \, d\Sigma_{ij}. \end{cases} \tag{1.29}$$

   and then $\mathcal{A}g^n$ can be computed using the surface weak formulation

$$\begin{cases} \text{Find } (\mathcal{A}g^n)_{ji} \in H^1(\Sigma_{ij}) \text{ such that, for every } g'_{ji} \in H^1(\Sigma_{ij}) \text{ and for all } i \text{ and } j: \\ \displaystyle\int_{\Sigma_{ij}} \overline{g}'_{ji} (\mathcal{A}g^n)_{ji} \, d\Sigma_{ij} = -\int_{\Sigma_{ij}} \overline{g}'_{ji} g_{ij}^n \, d\Sigma_{ij} + 2\int_{\Sigma_{ij}} \overline{g}'_{ji} i\omega s \tilde{u}_i^{n+1} \, d\Sigma_{ij}. \end{cases} \tag{1.30}$$

3. The interface fields g are given by the vector $\mathbf{x}$ and the final solution, $u_i = \mathcal{V}_i(f, g)$ for all i, corresponds to

$$\begin{cases} \text{Find } u_i \in H^1(\Omega_i) \text{ such that, for every } u' \in H^1(\Omega_i): \\ -\displaystyle\int_{\Omega_i} \nabla\overline{u}' \cdot \nabla u_i \, d\Omega_i + \int_{\Omega_i} \overline{u}'\omega^2 s^2 u_i \, d\Omega_i + \int_{\partial\Omega_i} \overline{u}' i\omega s u_i \, d\partial\Omega_i \\ \qquad\qquad = -\displaystyle\int_{\Omega_i} \overline{u}' s^2 f \, d\Omega_i - \sum_j \int_{\partial\Omega_i} \overline{u}' g_{ij} \, d\partial\Omega_i. \end{cases} \tag{1.31}$$

The volume (1.28) (1.29) (1.31) and surface (1.30) weak formulations are solved with direct solvers and their factorization can be reused.

Using the shape functions as in Section 1.3.1, the interface problem (1.27) can be represented as a square linear system $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{x}$ correspond to the interface unknowns g, $\mathbf{b}$ to the vector b and $\mathbf{A}$ to the operator $(\mathcal{I} - \mathcal{A})$. Furthermore, the matrix $\mathbf{A}$ will be sparse since the computation of $(g_{ji})_{\forall j}$ depends on g_{ji}, g_{ij} and $\tilde{u}_i$ on Σ_{ij} for all i and j.

Chapter 2

Krylov iterative solvers

The resolution of square linear system, presented in the matrix formulation $\mathbf{Ax} = \mathbf{b}$, has led to the derivation of a wide variety of algorithms. A comprehensive review of various numerical solvers and preconditioning techniques to address the Helmholtz equation can be found in reference [17]. This work identifies two distinct categories of solvers: direct and iterative. Each of these solver categories exhibits distinct advantages and drawbacks. In order to enhance the capabilities of iterative solvers, this review present multiple preconditioning techniques adequate for the Helmholtz equation. This enable to cast the considered problem into a form that is more suitable for numerical resolution. In general, the term "preconditioner" refers to the application of a transformation to the linear problem. This transformation is typically represented as the multiplication by a specific matrix, aptly referred to as the preconditioning matrix. The aim of this multiplication is then to formulate a problem with better convergence properties when solved with iterative methods. The preconditioning techniques can also refer to any resolution strategy improving the convergence such as domain decomposition methods, multigrid methods, etc. The choice of an appropriate preconditioner is itself a challenge and many solutions have been developed for the resolution of the Helmholtz equation [17], [14], [18], [19], [20]. As described in Section 1.3.2, the domain decomposition method is used in the current work. It allows to resolve many smaller subproblems and one interface problem instead of one large scale wave scattering problem. Thus, the interface problem can be viewed as a preconditioned form of the whole problem where a direct solver is used in the preconditioning and an iterative method is used for the resolution.

Direct solvers, or factorization solvers, are based on the factorization of the problem matrix into a simpler structure in order to easily solve the system. Typically, this structure is a LU factorization and efficient software packages implementing such methods have already being developed, for instance the library MUMPS [21] will be used in the experimentations. Direct solvers have the advantage of being fast and efficient on small problems. Also they provide the most accurate solution within the limitation of finite precision arithmetic. However, direct solvers suffer from several disadvantages as they scale badly both in time and storage requirements, they can introduce fill-in when applied to sparse matrices, and most importantly here they need the explicit construction of the problem matrix to proceed. Therefore, direct solvers will not be used to solve the interface problem but rather the subproblems related to each subdomain. Another advantage of direct solver is that once the factorization of the matrix is done, it can be used to solve any problem with a different right hand side. The most expensive part of the resolution being the factorization then the successive resolutions of problems with

different right hand sides can be done at a limited cost.

Conversely, indirect or iterative solvers approach linear problems by repeatedly improving an initial estimation of the solution. As a result, the obtained solution will only be accurate within a specified tolerance, and it may require a significant number of iterations to reach it. However, some iterative solvers are able to solve systems in a matrix-free manner which is crucial for the domain decomposition method as stated in Section 1.3.2. These methods also have the advantage of requiring only a limited amount of storage. For the resolution of multiple right hand sides, some information could be reused from one resolution to another, and this will be the objective of the current work. Specified libraries for iterative resolution already exist and here the PETSc library [22] will be used since it provides not only the implementation but also the tools to implement these methods. An in-depth exploration of iterative methods is conducted in [23] and it will be the basis of all the development in the following. Stationary methods such as Jacobi, Gauss-Seidel or successive over-relaxation, constitute the simplest kind of iterative solvers. At each iteration, these involve applying a fixed update rule on the curent approximate solution. Among these solvers, only the Jacobi method can solve systems in a matrix-free manner. However, a more general framework for iterative solvers is the Krylov subspace method which enables the development of more advanced iterative methods.

The conjugate gradient (CG) is a well known Krylov method, first described in [24], that enables the resolution of linear problems with an hermitian positive definite matrix. It consists in a short recurrence update algorithm, i.e. it only requires the information of the last iteration to proceed. Indeed, in each iteration, this algorithm involves computing a new search direction using the current residual vector and the previous search direction, except for the first one. This search direction is then used to improve the approximate solution. In order to compute the right correction to the approximate solution, one matrix vector product must be computed at each iteration. From CG, many generalizations were proposed to find similar algorithms for general square matrices. Some were proposed with a long recurrence update leading for instance to the generalized conjugate residual method (GCR) [25], the generalized minimal residual method (GMRES) [26], etc. These methods improve the solution by first computing new search directions, but these depend on all preceding directions instead of only the last one. Then, the approximate solution is improved by using all these directions to minimize the residual norm. Here again, only one matrix vector product is needed at each iteration. The memory requirement however is larger since the search directions and the associated information corresponding to all the iterations must be saved. Also, to compute a new search direction, all precedent directions must be used and therefore additional operations will come with each additional direction. This can lead to a bad scaling of the methods as the number of search direction increases. Hopefully, the memory and computation requirement can be limited by simply restarting the algorithms. The restarting strategy can in some cases lead to a decrease in convergence or even to the stagnation of the method, and therefore more advanced restarting strategies where developed such as adaptive restarting [27], [28] or other improved restarting methods as discussed in Section 2.6. Some of these methods illustrate the possibility to reuse some information about the search directions between restarts to improve convergence. This information recycling is also possible between the resolutions of systems with multiple right hand sides. Therefore this will then greatly accelerate the inversion process and it will be discussed in Chapter 3.

Another kind of generalization of CG which conserves the short recurrence property was also proposed, such as the biconjugate gradient method [29]. This one replace the orthogonal sequence of residuals by two mutually orthogonal sequences, but it no longer provides

a minimization of the residual. Therefore, the biconjugate gradient is not a monotonically decreasing method. The biconjugate gradient stabilized method (BICGSTAB) was develop to improve the convergence of the method [30]. These methods enable the resolution of the system at low memory and computation requirement since only the last iterate information is needed. The drawback however is that at each iterations two matrix vector multiplications are computed, one for each sequence. Therefore, even if the BiCGSTAB might converge in fewer iterations than GMRES for example, it can take more time since more matrix vector product are needed overall. Due to the larger number of matrix vector products during the resolution, the biconjugate gradient stabilized method will not be considered here since the matrix vector products will be expensive in comparison to the other operations. Recycling strategy for the biconjugate gradient method and its variations was proposed in [31], but the memory requirement for each iteration will correspond to store the information about the two sequences of directions. Therefore, the recycling biconjugate gradient stabilized will not be more interesting than GCR, GMRES, etc.

2.1 General procedure

The linear problem considered here is assumed to be of size n and therefore is noted as $\mathbf{A}\mathbf{x} = \mathbf{b}$ where $\mathbf{A}$ is a n by n matrix and $\mathbf{b}$ is a vector of size n. Iterative solvers will then use an initial vector $\mathbf{x}_0$ of size n as a first approximation to generate a sequence of improving solutions $\{\mathbf{x}_0, \mathbf{x}_1, \dots\}$ for this linear problem. The approximate solution at the m^{th} iteration is given by $\mathbf{x}_m = \mathbf{x}_0 + \boldsymbol{\delta}_m$ where $\boldsymbol{\delta}_m$ is the correction to the initial approximation. The corresponding error vector is defined as $\mathbf{e}_m = \mathbf{x} - \mathbf{x}_m$, however it is not available since the exact solution $\mathbf{x}$ is the unknown. On the other hand, the residual vector, represented as $\mathbf{r}_m = \mathbf{b} - \mathbf{A}\mathbf{x}_m$, serves as an alternative metric for assessing solution precision, and its calculation is feasible. For Krylov subspace solvers, the correction $\boldsymbol{\delta}_m$ is chosen in the m^{th} Krylov subspace generated by the problem matrix $\mathbf{A}$ and the initial residual $\mathbf{r}_0$:

$$\boldsymbol{\delta}_m \in \mathcal{K}_m = \mathcal{K}_m(\mathbf{A}, \mathbf{r}_0) = \text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{m-1}\mathbf{r}_0\}.$$

The correction vector $\boldsymbol{\delta}_m$ must be found in a m dimensional subspace and to do so m constraints must be imposed. The typical way of describing these constraints is to impose the orthogonality of the residual $\mathbf{r}_m$ to another subspace $\mathcal{L}_m$ of m dimensions. By choosing $\mathcal{L}_m$ equal to $\mathcal{K}_m$, it is possible to derive methods such as the full orthogonalization method, the conjugate gradient method (if $\mathbf{A}$ is symmetric), etc. However, other methods with slightly better convergence properties for indefinite matrices $\mathbf{A}$ can be derived if the subspace $\mathcal{L}_m$ is chosen as $\mathbf{A}\mathcal{K}_m$. This choice of orthogonal subspace lead to the minimization the norm of the residual, and therefore these methods are called minimization methods. Indeed, the correction $\boldsymbol{\delta}_m$ must ensure the orthogonality of the residual to every vector in the subspace $\mathbf{A}\mathcal{K}_m$ which is equivalent to minimizing the 2-norm of the residual for $\boldsymbol{\delta}_m \in \mathcal{K}_m$:

$$(\mathbf{b} - \mathbf{A}\mathbf{x}_m, \mathbf{A}\mathbf{w}) = 0 \quad \forall\, \mathbf{w} \in \mathcal{K}_m \qquad \Leftrightarrow \qquad \mathbf{x}_m = \underset{\tilde{\mathbf{x}}_m \in \mathbf{x}_0 + \mathcal{K}_m}{\arg\min} \|\mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}_m\|, \tag{2.1}$$

where the operator $(.,.)$ represents the inner product between two vectors. In general, the subspace $\mathcal{L}_m$ must not be necessarily defined from $\mathcal{K}_m$ as it is the case for the biconjugate gradient methods, for more details see [23]. However, only the minimization methods will be considered in the present work.

In the case of minimization methods, the approximate solution at the m^{th} iteration must be found such that the correction vector $\boldsymbol{\delta}_m$ is defined in the Krylov subspace $\mathcal{K}_m$ and the residual

vector $\mathbf{r}_m$ is orthogonal to $\mathcal{L}_m = \mathbf{A}\mathcal{K}_m$. Therefore, in order to represent these constraints, one basis is formed for the subspace $\mathcal{K}_m$ and it is typically noted as $\{\mathbf{v}_i \ \forall i = 1, \dots, m\}$. In matrix notation, this basis can be represented as a matrix where each column is a basis vector: $\mathbf{V}_m = [\mathbf{v}_1 \cdots \mathbf{v}_m]$. Since these vectors will be used to improve the approximate solution, they can then be described as search directions. This allows to write the approximate solution and the orthogonalization property as

$$\mathbf{x}_m = \mathbf{x}_0 + \mathbf{V}_m \mathbf{y}_m \qquad \text{and} \qquad (\mathbf{A}\mathbf{V}_m)^* \mathbf{r}_m = \mathbf{0}, \tag{2.2}$$

where $\mathbf{y}_m$ is an unknown vector of size m representing the correction $\boldsymbol{\delta}_m$ in the basis $\mathbf{V}_m$. Also the notation $(.)^*$ corresponds to the conjugate transpose operation and $\mathbf{0}$ is the null vector. Another interpretation of the orthogonality property is that the dot product between the residual vector $\mathbf{r}_m$ and every search direction multiplied by the matrix $\{\mathbf{A}\mathbf{v}_i \ \forall \ i = 1, \dots, m\}$ must be equal to zero, e.g. $(\mathbf{r}_m, \mathbf{A}\mathbf{v}_i) = \mathbf{v}_i^* \mathbf{A}^* \mathbf{r}_m = 0$ for all i from 1 to m. The first relation of (2.2) implies that the residual can be expressed with respect to the vector $\mathbf{y}_m$ as

$$\mathbf{r}_m = \mathbf{r}_0 - \mathbf{A}\mathbf{V}_m \mathbf{y}_m. \tag{2.3}$$

With this representation of the residual, the coefficient vector $\mathbf{y}_m$ can be computed by exploiting the orthogonalization constraint in (2.2). As it is described in (2.1), the orthogonalization property can equivalently expressed as a minimization problem. Therefore, the vector coefficients y_m can be found by either solving a square linear problem or by minimizing the norm of the residual vector:

$$(\mathbf{A}\mathbf{V}_m)^* \mathbf{A}\mathbf{V}_m \mathbf{y}_m = (\mathbf{A}\mathbf{V}_m)^* \mathbf{r}_0. \qquad \Leftrightarrow \qquad \mathbf{y}_m = \underset{\tilde{\mathbf{y}}_m}{\arg\min} \|\mathbf{r}_0 - \mathbf{A}\mathbf{V}_m \tilde{\mathbf{y}}_m\|. \tag{2.4}$$

The definitions and properties of the basis $\mathbf{V}_m$ will be important as they will define the subsequent methods. In the following, three minimization solvers will be describe using this methodology: the generalized conjugate residual method (GCR), the generalized minimal residual method (GMRES) and the simpler generalized minimal residual method (SGMRES). These algorithms differ in their implementation, but they exploit the same Krylov subspaces and all minimize the residual. Therefore, these methods are mathematically equivalent and would provide the exact same sequence of approximate solutions $\mathbf{x}_n$ if they were implemented in infinite precision arithmetic. The main differences between the considered methods are then linked to their implementation.

2.2 The generalized conjugate residual method

2.2.1 Derivation of GCR and Orthodir

The generalized conjugate residual is a minimization Krylov iteration solver described in [25] and it is also called Orthomin in some circumstances. A closely related method called Orthodir, that is sometimes used in the literature as in [1], can be derived with relative ease from the GCR algorithm and will also be described in the following. In these methods, the search directions $\mathbf{v}_m$ are defined $\mathbf{A}^*\mathbf{A}$-orthogonal, i.e. such that all the vectors $\mathbf{A}\mathbf{v}_m$ are orthogonal to each other. Therefore, when considering the square linear problem in (2.4), the matrix in the left hand side $\mathbf{V}_m^* \mathbf{A}^* \mathbf{A}\mathbf{V}_m$ will be diagonal. This induces that all the components of $\mathbf{y}_m$ are independent from each other and can be computed at each iteration. The approximate solution (2.2) and the residual vector (2.3) can be simply computed at each

step using update equations:

$$\mathbf{x}_m = \underbrace{\mathbf{x}_0 + \sum_{i=1}^{m-1} y_i \mathbf{v}_i}_{\mathbf{x}_{m-1}} + y_n \mathbf{v}_m \qquad \text{and} \qquad \mathbf{r}_m = \underbrace{\mathbf{r}_0 - \sum_{i=1}^{m-1} y_i \mathbf{A}\mathbf{v}_i}_{\mathbf{r}_{m-1}} - y_m \mathbf{A}\mathbf{v}_m. \tag{2.5}$$

To express the value of the coefficient y_m, the orthogonality property in (2.2) is used together with the update rule for the residual vector (2.5). Since the vectors $\mathbf{A}\mathbf{v}_m$ are orthogonal, the coefficient y_m can directly be computed as the ratio of two vector dot products. Also, this coefficient can be determined either from the initial residual $\mathbf{r}_0$ or the last residual $\mathbf{r}_{m-1}$:

$$y_m = \frac{(\mathbf{r}_0, \mathbf{A}\mathbf{v}_m)}{(\mathbf{A}\mathbf{v}_m, \mathbf{A}\mathbf{v}_m)} = \frac{(\mathbf{r}_{m-1}, \mathbf{A}\mathbf{v}_m)}{(\mathbf{A}\mathbf{v}_m, \mathbf{A}\mathbf{v}_m)}. \tag{2.6}$$

The second possibility is often preferred in practice because it simplifies the implementation. Indeed, the initial residual vector $\mathbf{r}_0$ will not be need to be stored during the whole resolution. On top of that, this will lead to a more stable scheme since the coefficient will be computed with respect to the current residual.

Since the first search vector is defined from the Krylov subspace, it must necessarily correspond to the initial residual up to a scaling factor, for instance $\mathbf{v}_1 = \mathbf{r}_0$. In order to construct the subsequent dimensions of the Krylov subspace, the next search direction $\mathbf{v}_{m+1}$ can be defined from the last one multiplied by the problem matrix, $\mathbf{A}\mathbf{v}_m$. This will lead to the derivation of the Orthodir method. Another possibility is however available. Indeed, the residual vector $\mathbf{r}_m$ is a linear combination of vectors from the last Krylov subspace $\mathcal{K}_m$ and the last search direction multiplied by the matrix, $\mathbf{A}\mathbf{v}_m$, as it is represented in (2.5). Therefore, the last residual vector $\mathbf{r}_m$ can equivalently be used to define the next search direction $\mathbf{v}_{m+1}$. This second method corresponds to the generalized conjugate residual method. Once the next search direction is defined, it must still be $\mathbf{A}^*\mathbf{A}$-orthogonalized with respect to all the search vectors $\{\mathbf{v}_i \; \forall \; i = 1, \dots, m\}$. The GCR and Orthodir method are then tightly linked since the resulting search direction $\mathbf{v}_{m+1}$ will be the same in exact arithmetic. However, the GCR method proved to be more stable than Orthodir in finite precision arithmetic as it was highlighted in [32]. The GCR and Orthodir methods can be described by almost the same algorithm as it is done in Algorithm 1.

Algorithm 1 The GCR (or Orthodir) method.

1: Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$ then set $\mathbf{v}_1 = \mathbf{r}_0$.
2: **for** $m = 1, \dots$ until convergence **do**
3: $\quad y_m = (\mathbf{r}_{m-1}, \mathbf{A}\mathbf{v}_m) \,/\, (\mathbf{A}\mathbf{v}_m, \mathbf{A}\mathbf{v}_m)$ ▷ Update the solution and the residual.
4: $\quad \mathbf{x}_m = \mathbf{x}_{m-1} + y_m \mathbf{v}_m$
5: $\quad \mathbf{r}_m = \mathbf{r}_{m-1} - y_m \mathbf{A}\mathbf{v}_m$
6: $\quad \mathbf{v}_{m+1} = \mathbf{r}_m$ (or $\mathbf{v}_{m+1} = \mathbf{A}\mathbf{v}_m$ for Orthodir) ▷ Next search direction.
7: $\quad$ Compute $\beta_{i,m} = (\mathbf{A}\mathbf{v}_{m+1}, \mathbf{A}\mathbf{v}_i) \,/\, (\mathbf{A}\mathbf{v}_i, \mathbf{A}\mathbf{v}_i)$ for $i = 1, \dots, m$. ▷ Orthogonalization.
8: $\quad \mathbf{v}_{m+1} = \mathbf{v}_{m+1} - \sum_{i=1}^{m} \beta_{i,m} \mathbf{v}_i$

2.2.2 Practical implementation of GCR and Orthodir

The GCR and Orthodir algorithms are often described in a similar manner as in the Algorithm 1. Nevertheless, the practical implementations of these methods may exhibit sub-

stantial differences. Indeed, it is possible to adapt these methods to better accommodate the specifics of the considered system. For instance, the precision of the orthogonalization can be improved, the number of matrix vector products and vector operations can be reduced as much as possible, and also the overall storage and computational requirements can be limited.

The main source of numeric errors in Algorithm 1 relies in the orthogonalization of the search directions on lines 7-8. In fact, it is done using the classical Gram–Schmidt orthogonalization. This process computes the orthogonalization coefficients $\beta_{i,m}$ from the non-orthogonal new search direction directly. This allows to compute all these coefficients in parallel, however it can lead to the accumulation errors and eventually to a loss of orthogonality. A slight variation of this orthogonalization can be derived to improve the stability of the process. The idea is to compute the orthogonalization coefficients $\beta_{i,m}$ from the new search direction already orthogonalized by the precedent vectors $\{\mathbf{v}_1, \dots, \mathbf{v}_{i-1}\}$. This is known as the modified Gram–Schmidt orthogonalization. The coefficients $\beta_{i,m}$ can no longer be computed in parallel, but the resulting process achieve a greater stability. Some alternatives were also proposed in order to reduce even more the accumulation of errors, such as reorthogonalization of the basis or the use of Householder reflections as the orthogonalization process. These approach ensures the orthogonalization of the vector basis, however they require more operations as it is described in [23]. Therefore, in practice the modified Gram–Schmidt process is very popular for its trade-off between computational requirement and numerical stability. The modified Gram–Schmidt orthogonalization in then used in Algorithm 2.

In most of applications, the most expensive operation of these methods will be the matrix vector products. This is especially the case in the context of the resolution of the Helmholtz equation using the domain decomposition method as explained in Section 1.3.2. Indeed, one matrix vector product requires the direct resolution of the subdomain problems. Even if the factorization of these problems must be computed only once, the resolution still correspond to an expensive operation due to the large number of degrees of freedom. For that reason, the matrix vector products should be limited as much as possible. In order to limit the number of matrix vector product, the vector $\mathbf{A}\mathbf{v}_m$ will be computed only once and stored in a separated vector noted $(\mathbf{A}\mathbf{v})_m$. Then, this vector must also be affected by the orthogonalization on lines 7-8. It is achieved by using the following relations respectively for GCR and Orthodir:

$$\mathbf{A}\mathbf{v}_{m+1} = \mathbf{A}\mathbf{r}_m - \sum_{i=1}^{m} \beta_{i,m}\mathbf{A}\mathbf{v}_i \qquad \text{and} \qquad \mathbf{A}\mathbf{v}_{m+1} = \mathbf{A}^2\mathbf{v}_m - \sum_{i=1}^{m} \beta_{i,m}\mathbf{A}\mathbf{v}_i. \tag{2.7}$$

The resulting algorithm, Algorithm 2, then only requires one matrix-vector product per iteration. An even simpler implementation can be achieved by normalizing the vectors $(\mathbf{A}\mathbf{v})_m$ and scaling accordingly the vectors $\mathbf{v}_m$. This will simplify the computation of the coefficients y_m and $\beta_{i,m}$, but it will also lead to a more stable and robust implementation. Indeed, this will prevent the search directions to become too large or too small, and thus limiting the rounding errors. Moreover, the solver will be less sensitive to the problem scales.

The principal disadvantage of this class of algorithms is the need to store, orthogonalize and scale both the search directions $\mathbf{v}_m$ and their multiplication by the matrix $(\mathbf{A}\mathbf{v})_m$. This can lead to prohibitive storage and computational requirement as the number of iterations m becomes large. As explained, a simple way to limit the consumption of these methods is to restart them after a number of iterations k at the cost of a possible decrease of the convergence. The restarted version of GCR is also known as the Orthomin method because it was first derived in [33] as a variation of the (restarted) Orthodir method [34]. In practice, these methods are often implemented with restarting to limit the possibility of requiring more

memory than available. The implementation given by the Algorithm 2 then represents one cycle of the restarted methods. There are two possibilities for this cycle to end. The first one is that the maximum number of iterations in one cycle is reached, and this will lead to a another cycle with the improved approximate solution $\mathbf{x}_k$. The other situation is that a convergence criterion is reached such that the method ends. Here, the convergence criterion will be defined such that the relative residual norm with respect to the initial residual norm is smaller than a specified tolerance noted as tol.

Another possibility to limit the memory consumption of the GCR and Orthodir methods is also presented in [23]. Since the precedent search directions are only used in the orthogonalization step, it is possible to keep only the k last directions in these algorithms. This will also correspond to a limitation on the storage and computational requirement and the convergence can be similarly affected. The resulting methods are known as the truncated GCR and Orthodir, but they will not be studied in the current work since a truncated version of the other minimization methods can not be readily developed.

Algorithm 2 One cycle of the restarted GCR or Orthodir methods.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
for $m = 1, \ldots, k$ **do**
  if $\|\mathbf{r}_{m-1}\|/\|\mathbf{r}_0^{\text{init}}\| < tol$ **then** ▷ Check for convergence.
    Exit.
  if Orthodir and $m > 1$ **then** ▷ New search direction.
    $\mathbf{v}_m = (\mathbf{A}\mathbf{v})_{m-1}$
  else
    $\mathbf{v}_m = \mathbf{r}_{m-1}$
  $(\mathbf{A}\mathbf{v})_m = \mathbf{A}\mathbf{v}_m$
  for $i = 1, \ldots, m-1$ **do** ▷ Modified Gram-Schmidt orthogonalization.
    $\beta_{i,m} = ((\mathbf{A}\mathbf{v})_m, (\mathbf{A}\mathbf{v})_i)$
    $\mathbf{v}_m = \mathbf{v}_m - \beta_{i,m}\mathbf{v}_i$
    $(\mathbf{A}\mathbf{v})_m = (\mathbf{A}\mathbf{v})_m - \beta_{i,m}(\mathbf{A}\mathbf{v})_i$
  $\beta_{m,m} = \|(\mathbf{A}\mathbf{v})_m\|$
  $\mathbf{v}_m = \mathbf{v}_m/\beta_{m,m}$
  $(\mathbf{A}\mathbf{v})_m = (\mathbf{A}\mathbf{v})_m/\beta_{m,m}$
  $y_m = (\mathbf{r}_{m-1}, (\mathbf{A}\mathbf{v})_m)$ ▷ Update the solution and the residual.
  $\mathbf{x}_m = \mathbf{x}_{m-1} + y_m\mathbf{v}_m$
  $\mathbf{r}_m = \mathbf{r}_{m-1} - y_m(\mathbf{A}\mathbf{v})_m$

With this specific implementation, the storage and computational requirements of the GCR or Orthodir methods can be estimated for a single cycle of k iterations. In practice, the size of the problem, denoted as n, is anticipated to be notably large in comparison to the number of iterations k. Otherwise, resolving the problem with a direct solver would result in comparable memory usage and computational operations. Consequently, a direct solver would have been more suitable than iterative methods. In this context, only the memory and operations related to the vectors will be significant because one vector corresponds to n scalars.

In terms of storage requirements, the memory consumption related to the matrix $\mathbf{A}$, the right hand side $\mathbf{b}$ and the initial guess $\mathbf{x}_0$ will not be taken into account, as these elements

are consistently necessary regardless of the solver considered. Additionally, the final solution $\mathbf{x}_k$ will be stored in the same memory space as the initial guess $\mathbf{x}_0$, hence it will also not be considered. The memory consumption will then correspond to the storage of the residual vector $\mathbf{r}_m$, the search directions $\mathbf{v}_m$ and their multiplications with the problem matrix $(\mathbf{Av})_m$. All together, at least $(2k+1)n$ scalars must be stored during one cycle of k iterations.

In the context of computational needs, the main requirements will correspond to the matrix vector products as previously mentioned. In general, the matrix vector product can correspond to different processes depending of the problem considered. Therefore, all the operations due to the matrix vector product will be accounted separately from the other computational cost of the methods. One matrix vector product is needed to compute the initial residual at the beginning of the methods, except in cases where the initial guess is the null vector. Additionally, another matrix vector product is needed at each iteration. In total, the number of matrix vector products will be of $k+1$ for one cycle.

The other computational needs correspond to the multiplications and additions related to vector operations. At the m^{th} iteration, the computation of the residual norm correspond to n products and sums. Then, the orthogonalization and normalization of the vectors correspond to $3mn$ multiplications and $(3m-2)n$ additions. And finally, to update the solution and the residual, $3n$ more products and sums are needed. For one cycle of k iterations, the total number of products and additions are respectively of the order of $(3/2k^2+9/2k)n$ and $(3/2k^2+5/2k)n$. Even if the number of operation can be substantial for problem with a large size n, most of the work can be done in parallel thus reducing the computation time.

2.3 The generalized minimal residual method

2.3.1 Derivation of GMRES

The generalized minimal residual method is a minimization Krylov iterative solver which was first described in [26]. For this algorithm, the search directions are defined as orthonormal and the Arnoldi method [35] is used to build this orthonormal basis for the Krylov subspace. It consists in choosing the next search direction $\mathbf{v}_{m+1}$ from the current basis vector multiplied by the problem matrix $\mathbf{Av}_m$. Then, this vector is orthogonalized with respect to all precedent search directions and the orthogonalization coefficients are stored. One iteration of this method is described in Algorithm 3.

Algorithm 3 One iteration of the Arnoldi method.

1: $\mathbf{v}_{m+1} = \mathbf{Av}_m$
2: Compute $h_{i,m} = (\mathbf{v}_{m+1}, \mathbf{v}_i)$ for $i = 1, \ldots, m$.
3: $\mathbf{v}_{m+1} = \mathbf{v}_{m+1} - \sum_{i=1}^{m} h_{i,m}\mathbf{v}_i$
4: $h_{m+1,m} = \|\mathbf{v}_{m+1}\|$
5: $\mathbf{v}_{m+1} = \mathbf{v}_{m+1}/h_{m+1,m}$

All the coefficients are stored in a $m+1$ by m matrix denoted $\bar{\mathbf{H}}_m$. Since the orthonormalized only depends on precedent search directions, this matrix has the structure of an upper Hessenberg matrix, i.e. the structure of an upper triangular matrix with a non-zero subdiagonal. Also, these coefficients correspond to the decomposition of the vector $\mathbf{Av}_m$ in the Krylov subspace. The Arnoldi method allows then to express every search direction multiplied by the matrix $\mathbf{Av}_m$ as a linear combination of the the Krylov basis $\mathbf{V}_{m+1}$. These properties can be

summarized as

$$\mathbf{A}\mathbf{V}_m = \mathbf{V}_{m+1}\bar{\mathbf{H}}_m \qquad \text{with} \quad \bar{\mathbf{H}}_m = \begin{pmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,m} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,m} \\ & h_{3,2} & \ddots & \vdots \\ & & \ddots & h_{m,m} \\ & & & h_{m+1,m} \end{pmatrix}. \tag{2.8}$$

The Arnoldi iteration allows for the simultaneous computation of both the basis vectors $\mathbf{v}_{m+1}$ and the coefficient matrix $\bar{\mathbf{H}}_m$. This relation (2.8) can then be introduced in (2.3) to express the residual vector. Moreover, the first search direction is defined as the normalized initial residual: $\mathbf{v}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|$. Therefore, the initial residual can be expressed from the Krylov basis and the initial residual norm which is typically noted β: $\mathbf{r}_0 = \mathbf{V}_{m+1}\beta\mathbf{e}_1$ where $\mathbf{e}_1$ is a vector of size $m+1$ with the only non-zero value is the first component and which is equal to one. This allows to express the residual vector as

$$\mathbf{r}_m = \mathbf{r}_0 - \mathbf{A}\mathbf{V}_m\mathbf{y}_m = \mathbf{r}_0 - \mathbf{V}_{m+1}\bar{\mathbf{H}}_m\mathbf{y}_m = \mathbf{V}_{m+1}\left(\beta\mathbf{e}_1 - \bar{\mathbf{H}}_m\mathbf{y}_m\right). \tag{2.9}$$

To find the correction to the solution, the coefficient vector $\mathbf{y}_m$ must be determined such that the residual norm is minimized as stated in (2.4). The representation of the residual vector given in (2.9) can be introduced in this minimization and since orthonormal matrices, such as the basis $\mathbf{V}_{m+1}$, preserve the norm, it results in

$$\mathbf{y}_m = \underset{\tilde{\mathbf{y}}_m}{\arg\min} \|\mathbf{r}_0 - \mathbf{A}\mathbf{V}_m\tilde{\mathbf{y}}_m\| = \underset{\tilde{\mathbf{y}}_m}{\arg\min} \left\|\beta\mathbf{e}_1 - \bar{\mathbf{H}}_m\tilde{\mathbf{y}}_m\right\|. \tag{2.10}$$

To facilitate the resolution of this minimization problem, a possible approach is to transform the matrix $\bar{\mathbf{H}}_m$ into an upper triangular form. Indeed, a Hessenberg matrix is only a few plane rotations away from being transformed into an upper triangular matrix. These rotations are intended to zero out the values on the subdiagonal and as a result, they are referred to as Givens rotations. Then, by using m Givens rotation $\{\mathbf{\Omega}_i \ \forall i = 1, \ldots, m\}$, the Hessenberg matrix $\bar{\mathbf{H}}_m$ can be transformed in a $m+1$ by m upper triangular matrix noted $\bar{\mathbf{R}}_m$. The Givens rotation matrices are square matrices of size $m+1$ that are defined as

$$\mathbf{\Omega}_i = \begin{pmatrix} \mathbf{I}_{i-1} & & & \\ & c_i & s_i & \\ & -\bar{s}_i & \bar{c}_i & \\ & & & \mathbf{I}_{m-i} \end{pmatrix} \qquad \text{such that} \quad \mathbf{\Omega}_m \ldots \mathbf{\Omega}_1\bar{\mathbf{H}}_m = \bar{\mathbf{R}}_m, \tag{2.11}$$

where $\mathbf{I}_i$ represent the identity matrix of size i. The coefficients c_i and s_i must be computed such that the application of the i^{th} Given rotation will zero out the last component of the i^{th} column of $\bar{\mathbf{H}}_m$. Once these are computed, they can be introduced in the minimization problem (2.10):

$$\mathbf{y}_m = \underset{\tilde{\mathbf{y}}_m}{\arg\min} \left\|\bar{\mathbf{g}}_m - \bar{\mathbf{R}}_m\tilde{\mathbf{y}}_m\right\| \qquad \text{where} \quad \bar{\mathbf{g}}_m = \mathbf{\Omega}_m \ldots \mathbf{\Omega}_1\|\mathbf{r}_0\|\mathbf{e}_1. \tag{2.12}$$

This minimization is achieved by simply solving the triangular problem $\mathbf{R}_m\mathbf{y}_m = \mathbf{g}_m$ where the last row of $\bar{\mathbf{R}}_m$ and $\bar{\mathbf{g}}_m$ are removed since all the values $r_{m+1,i}$ are equal to zero for all i from 1 to m.

On top of simplifying the minimization problem, the Givens rotations allow to express and compute the residual norm without explicitly solving for $\mathbf{y}_m$. Indeed, the Givens rotations are unitary transformations such that the residual norm can be compute by

$$\|\mathbf{r}_m\| = \left\|\beta\mathbf{e}_1 - \bar{\mathbf{H}}_m\mathbf{y}_m\right\| = \left\|\bar{\mathbf{g}}_m - \bar{\mathbf{R}}_m\mathbf{y}_m\right\|. \tag{2.13}$$

As the coefficients $\mathbf{y}_m$ will be determined such that $\mathbf{R}_m\mathbf{y}_m = \mathbf{g}_m$, it becomes evident that the absolute value of the last component of the vector $\bar{\mathbf{g}}_m$ is equal to the residual norm: $|g_{m+1}| = \|\mathbf{r}_m\|$. This is essential since this norm will be used at each iteration in the convergence criterion. This enables the construction of the solution to be carried out just once at the end of the resolution. Every step of the GMRES method is summarized in Algorithm 4.

Algorithm 4 The GMRES method.

1: Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\beta = \|\mathbf{r}_0\|$ and set $\mathbf{v}_1 = \mathbf{r}_0/\beta$.
2: **for** $m = 1, \ldots$ until convergence **do**
3: $\quad\mathbf{v}_{m+1} = \mathbf{A}\mathbf{v}_m$ ▷ Next search direction.
4: $\quad$Compute $h_{i,m} = (\mathbf{v}_{m+1}, \mathbf{v}_i)$ for $i = 1, \ldots, m$. ▷ Arnoldi iteration.
5: $\quad\mathbf{v}_{m+1} = \mathbf{v}_{m+1} - \sum_{i=1}^{m} h_{i,m}\mathbf{v}_i$
6: $\quad h_{m+1,m} = \|\mathbf{v}_{m+1}\|$
7: $\quad\mathbf{v}_{m+1} = \mathbf{v}_{m+1}/h_{m+1,m}$
8: $\quad$Find $\mathbf{\Omega}_i$ for $i = 1, m$ such that $\mathbf{\Omega}_i \ldots \mathbf{\Omega}_i\bar{\mathbf{H}}_m = \bar{\mathbf{R}}_m$. ▷ Apply Givens rotations.
9: $\quad$Compute $\bar{\mathbf{g}} = \mathbf{\Omega}_i \ldots \mathbf{\Omega}_i\beta\mathbf{e}_1$.
10: Solve the triangular problem $\mathbf{R}_m\mathbf{y}_m = \mathbf{g}_m$. ▷ Find the solution.
11: $\mathbf{x}_m = \mathbf{x}_0 + \sum_{i=1}^{m} y_i\mathbf{v}_i$.

The GMRES method requires less storage and orthogonalization operations than the GCR method since the vectors $\mathbf{A}\mathbf{v}_m$ must not be stored or orthogonalized. Indeed, these vectors are instead represented as a linear combination of the basis vectors as represented in (2.8). Another difference with the GCR methods is that the solution is only given at the end of the GMRES method by solving a triangular system of size m. It is also important to note that for the GMRES method, the next seach direction must necessarily be defined from the last one multiplied by the problem matrix since it employs the Arnoldi method. Therefore, a variation of this method based on the residual vector is not possible.

2.3.2 Practical implementation of GMRES

Some precision on the Algorithm 4 must still be provided in order to obtain a practical implementation of the method corresponding to the problem considered. Similarly as for the GCR/Orthodir method, the orthogonalization of the search vector will be done with the modified Gram-Schmidt process instead of the classical one in order to enhance the numerical stability of the method. Again, the orthogonalization could also be computed using reorthogonalization or the Householder reflections to improve the numerical stability. Also, the overall memory consumption of the method can be limited by restarting the algorithm. Therefore, in Alorithm 5, one cyle of k iterations of the method is described.

The coefficients within the matrices $\bar{\mathbf{H}}_m$ and $\bar{\mathbf{R}}_m$ are never used simultaneously, therefore they can be store in the same memory space. This principle also applies to the vectors $\mathbf{y}_m$ and

$\mathbf{g}_m$. At the beginning of the m^{th} iteration, the first $m-1$ columns of $\bar{\mathbf{R}}_m$ have already been computed. Then, the last column of the Hessenberg matrix must be computed by the mean of the Arnoldi method and all the $m-1$ precedent Givens rotations must be applied to this column. The last step corresponds to compute the m^{th} Givens rotation matrix $\mathbf{\Omega}_m$ such that

$$\mathbf{\Omega}_m \begin{pmatrix} r_{1,1} & \cdots & r_{1,m-1} & r_{1,m} \\ & \ddots & \vdots & \vdots \\ & & r_{m-1,m-1} & r_{m-1,m} \\ & & & h^o_{m,m} \\ & & & h_{m+1,m} \end{pmatrix} = \begin{pmatrix} r_{1,1} & \cdots & r_{1,m-1} & r_{1,m} \\ & \ddots & \vdots & \vdots \\ & & r_{m-1,m-1} & r_{m-1,m} \\ & & & r_{m,m} \\ & & & \end{pmatrix}, \tag{2.14}$$

where $h^o_{m,m}$ refer to the corresponding coefficient of the matrix $\bar{\mathbf{H}}_m$ after the application of the $m-1$ first Givens rotations. Different ways of computing the complex rotations factor c_m and s_m exist. Following the convention described in [36], these factors are computed as

$$c_m = \frac{|h^o_{m,m}|}{\sqrt{|h^o_{m,m}|^2 + |h_{m+1,m}|^2}} \qquad \text{and} \qquad s_m = \frac{h^o_{m,m}}{|h^o_{m,m}|} \frac{\overline{h}_{m+1,m}}{\sqrt{|h^o_{m,m}|^2 + |h_{m+1,m}|^2}}. \tag{2.15}$$

The computation of the resulting coefficient $r_{m,m}$ is also provideed as

$$r_{m,m} = h_{m,m} \frac{\sqrt{|h_{m,m}|^2 + |h_{m+1,m}|^2}}{|h_{m,m}|^2}. \tag{2.16}$$

Furthermore, the component $h_{m+1,m}$ is a real value since it represent the norm of a vector. Some simplification can occur in these computations as it is illustrated in Algorithm 5.

In the GMRES method, only one set of vectors, the basis vectors $\mathbf{V}_{m+1}$, must be stored. The coefficient $\mathbf{g}_m$, the Givens rotation coefficients and the matrix $\mathbf{R}_m$ must also be stored but these will be negligible since the size of the problem n is expected to be much larger than the number of iterations k in one cycle. The memory requirement is then of the order of $(k+1)n$ scalars. Here again, $k+1$ matrix vector products are required in total. In term of the number of operation, for the m^{th} iteration the Arnoldi iteration requires approximately $(2m+2)n$ products and $(2m+1)n$ sums. To compute and apply the Givens rotations, a negligible amount of operation is required. For a full cycle of k iterations, the number of products and sums in the main loop corresponds to $(k^2+3k)n$ and $(k^2+2k)n$ respectively. Then, the resolution of the triangular system of size m can also be neglected. In addition, kn more products and sums are needed to update the solution. Eventually, the total number of product for the vector operations is roughly $(k^2+4k)n$ while the total number of addition is of the order of $(k^2+3k)n$. The GMRES method is then more advantageous both in term of storage and computational operation than GCR as expected.

2.4 The simpler GMRES method

2.4.1 Derivation of SGMRES and RB-SGMRES

The simpler GMRES method is, as its name suggests, is a method that shares some similarities with GMRES while leading to a simpler implementation. It was initially introduced in [37]. For the usual GMRES method, the Arnoldi iteration is applied to the vectors of the Krylov basis $\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^m\mathbf{r}_0\}$ thus leading to the search basis $\mathbf{V}_{m+1}$. In the case of the simpler GMRES method, the Arnoldi method is employed on a shifted Krylov subspace

Algorithm 5 One cycle of the restarted GMRES method.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $y_1 = \|\mathbf{r}_0\|$ and set $\mathbf{v}_1 = \mathbf{r}_0/y_1$.
for $m = 1, \ldots, k$ **do**
  if $|y_m|/y_1 < tol$ **then** ▷ Check for convergence.
    Exit.
  $\mathbf{v}_{m+1} = \mathbf{A}\mathbf{v}_m$ ▷ Next search direction.
  for $i = 1, \ldots, m$ **do** ▷ Arnoldi iteration with modified Gram-Schmidt.
    $h_{i,m} = (\mathbf{v}_{m+1}, \mathbf{v}_i)$
    $\mathbf{v}_{m+1} = \mathbf{v}_{m+1} - h_{i,m}\mathbf{v}_i$
  $h_{m+1,m} = \|\mathbf{v}_{m+1}\|$
  $\mathbf{v}_{m+1} = \mathbf{v}_{m+1}/h_{m+1,m}$
  for $i = 1, \ldots, m-1$ **do** ▷ Apply precedent Givens rotations.
    $h_{i,m} = c_i h_{i,m} + s_i h_{i+1,m}$ and $h_{i+1,m} = -\overline{s}_i h_{i,m} + \overline{c}_i h_{i+1,m}$
  $c_m = |h_{m,m}| \Big/ \sqrt{|h_{m,m}|^2 + h_{m+1,m}^2}$. ▷ Compute and apply the new Givens rotation.
  $s_m = (h_{m,m}/|h_{m,m}|) \left(h_{m+1,m} \Big/ \sqrt{|h_{m,m}|^2 + h_{m+1,m}^2} \right)$
  $h_{m,m} = h_{m,m} \left(\sqrt{|h_{m,m}|^2 + h_{m+1,m}^2} \Big/ |h_{m,m}|^2 \right)$
  $y_{m+1} = -\overline{s}_m y_m$ and $y_m = c_m y_m$
for $m = k, \ldots, 1$ **do** ▷ Find the solution.
  $y_m = \left(y_m - \sum_{i=m+1}^{k} h_{m,i}\, y_i \right) \Big/ h_{m,m}$
for $m = 1, \ldots, k$ **do**
  $\mathbf{x}_m = \mathbf{x}_{m-1} + y_m \mathbf{v}_m$

starting from $\mathbf{A}\mathbf{r}_0$ instead of $\mathbf{r}_0$: $\{\mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, \dots, \mathbf{A}^m\mathbf{r}_0\}$. The resulting set of vectors from this process will be denoted as $\mathbf{C}_m$ and will constitutes an orthonormal basis of the subspace $\mathbf{A}\mathcal{K}_m$. Consequentially, the representation of the vectos $\mathbf{A}\mathbf{V}_m$ will also deviate from the classical GMRES expression (2.8) and will take the form of

$$\mathbf{A}\mathbf{V}_m = \mathbf{C}_m\mathbf{T}_m \qquad \text{with} \quad \mathbf{T}_m = \begin{pmatrix} t_{1,1} & t_{1,2} & \cdots & t_{1,m} \\ & t_{2,2} & \cdots & t_{2,m} \\ & & \ddots & \vdots \\ & & & t_{m,m} \end{pmatrix}. \tag{2.17}$$

This time the matrix of coefficient $\mathbf{T}_m$ is already a square upper triangular matrix of dimension m. This is becomes evident when considering that the product of the search direction and the problem matrix, $\mathbf{A}\mathbf{v}_m$, can be represented as a linear combination of the basis of $\mathbf{C}_m$. The Arnoldi iteration enables to then jointly compute the basis vectors $\mathbf{c}_m$ and the coefficient matrix $\mathbf{T}_m$. Similarly to the GMRES method, the relation (2.17) is introduce in (2.3) to express the residual vector:

$$\mathbf{r}_m = \mathbf{r}_0 - \mathbf{A}\mathbf{V}_m\mathbf{y}_m = \mathbf{r}_0 - \mathbf{C}_m\mathbf{T}_m\mathbf{y}_m. \tag{2.18}$$

In addition, a vector $\mathbf{q}_m$ of size m can be defined such that it correspond to the components in the basis $\mathbf{C}_m$ of the orthogonal projection of the initial residual $\mathbf{r}_0$ in the subspace $\mathbf{A}\mathcal{K}_m$: $\mathbf{q}_m = \mathbf{C}_m^*\mathbf{r}_0$. Following that and since the basis $\mathbf{C}_m$ is orthonormal, the miminization (2.4) is expressed for the simpler GMRES method as

$$\mathbf{y}_m = \arg\min_{\tilde{\mathbf{y}}_m} \|\mathbf{r}_0 - \mathbf{C}_m\mathbf{T}_m\tilde{\mathbf{y}}_m\| = \arg\min_{\tilde{\mathbf{y}}_m} \|\mathbf{q}_m - \mathbf{T}_m\tilde{\mathbf{y}}_m\|. \tag{2.19}$$

Since the coefficent matrix $\mathbf{T}_m$ is already upper triangular, there will be no need to compute and used Givens rotations, and the minimization can be ensured by resolving the system $\mathbf{T}_m\mathbf{y}_m = \mathbf{q}_m$.

The residual norm is not readily available as it is the case in the classical GMRES method. Therefore, the residual vector must be computed and normalized at each step for the convergence crierion. However, in order to limit the number of matrix vector product, it will not be evaluated from its definition but rather from the relation (2.18). By taking advantage the fact that $\mathbf{y}_m$ will be determined such that $\mathbf{T}_m\mathbf{y}_m = \mathbf{q}_m$, an update rule for the residual can be developed:

$$\mathbf{r}_m = \mathbf{r}_0 - \mathbf{C}_m\mathbf{T}_m\mathbf{y}_m = \mathbf{r}_0 - \mathbf{C}_m\mathbf{C}_m^*\mathbf{r}_0 = \mathbf{r}_{m-1} - \mathbf{c}_m\mathbf{c}_m^*\mathbf{r}_0. \tag{2.20}$$

This update rule actually corresponds to the projection of $\mathbf{r}_0$ on the complement space of $\mathbf{A}\mathcal{K}_m$. Given that the complete residual vector $\mathbf{r}_m$ will be available at every iteration of the method, there exists an opportunity for a simplification the computation of the vector $\mathbf{q}_m$:

$$q_m = \mathbf{c}_m^*\mathbf{r}_0 = \mathbf{c}_m^*\left(\mathbf{r}_{m-1} + \mathbf{C}_{m-1}\mathbf{C}_{m-1}^*\mathbf{r}_0\right) = \mathbf{c}_m^*\mathbf{r}_{m-1}, \tag{2.21}$$

This will allow to not store the initial residual vector during resolution.

The final aspect to establish is the method for constructing the basis $\mathbf{V}_m$ of the Krylov subspace $\mathcal{K}_m$. This basis is not subject to any orthogonality constraints, and thus, to minimize the quantity of stored vectors, the basis $\mathbf{V}_m$ was initially formulated as follows:

$$\mathbf{v}_m = \begin{cases} \mathbf{r}_0 & \text{if } m = 1, \\ \mathbf{c}_{m-1} & \text{otherwise.} \end{cases} \tag{2.22}$$

Nevertheless, in practice this implementation of simpler GMRES is rarely used primarily because of stability concerns in comparison to the GMRES method. This is illustrated in [32] and a variation of the method is proposed to improve the numerical stability of the algorithm. It consist in defining the vector $\mathbf{v}_m$ from the residual vectors $\mathbf{r}_m$ instead. The resulting basis will still correspond the Krylov subspace as it was shown in Section 2.2.1 for the GCR method. This variation of the method is called the residual based simpler GMRES method (RB-SGMRES). Its main drawback is that the basis vector $\mathbf{v}_m$ must be stored on top of the vectors $\mathbf{c}_m$ since they do not share vectors anymore. Thus, it leads to a larger memory consumption for the RB-SGMRES method. The Algorithm 6 includes all the stages of both the SGMRES and RB-SGMRES methods.

Algorithm 6 The SGMRES (or RB-SGMRES) method.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, set $\mathbf{v}_1 = \mathbf{r}_0$ and $\mathbf{c}_1 = \mathbf{A}\mathbf{v}_1$.
for $m = 1, \ldots$ until convergence **do**
  $q_m = \mathbf{c}_m^* \mathbf{r}_0$ ▷ Update the residual.
  $\mathbf{r}_m = \mathbf{r}_{m-1} - q_m \mathbf{c}_m$
  $\mathbf{v}_{m+1} = \mathbf{c}_m$ (or $\mathbf{v}_{m+1} = \mathbf{r}_m$ for RB-SGMRES) ▷ Next search direction.
  $\mathbf{c}_{m+1} = \mathbf{A}\mathbf{v}_{m+1}$
  Compute $t_{i,m+1} = (\mathbf{c}_{m+1}, \mathbf{c}_i)$ for $i = 1, \ldots, m$. ▷ Arnoldi-like iteration
  $\mathbf{c}_{m+1} = \mathbf{c}_{m+1} - \sum_{i=1}^{m} t_{i,m+1} \mathbf{c}_i$
  $t_{m+1,m+1} = \|\mathbf{c}_{m+1}\|$
  $\mathbf{c}_{m+1} = \mathbf{c}_{m+1} / t_{m+1,m+1}$
Solve the triangular problem $\mathbf{T}_m \mathbf{y}_m = \mathbf{q}_m$. ▷ Find the solution.
$\mathbf{x}_m = \mathbf{x}_0 + \sum_{i=1}^{m} y_i \mathbf{v}_i$.

2.4.2 Practical implementation of SGMRES and RB-SGMRES

The Algorithm 6 is already close to the implementation of these methods used in practice. Once more, the orthogonalization will be executed using the modified Gram-Schmidt procedure to ensure numerical stability. In Algorithm 7 a restarting procedure will also be introduce to mitigate resource consumption of these methods. Furthermore, in this algorithm, the computation of the next basis vector is skipped when the algorithm has achieved convergence or is restarted. As the elements within vectors $\mathbf{y}_m$ and $\mathbf{q}_m$ are never employed concurrently, they can be stored in the same memory space. Also in order to improve the robustness of the implementation, the search vectors $\mathbf{v}_m$ will be normalized as it is usually the case.

Regarding memory utilization, there exists a distinction between SGMRES and RB-SGMRES methods. Indeed, the simpler GMRES approach almost mirrors the memory requirements of the classical GMRES algorithm in term of vectors. The only difference lies in the necessity to retain the first search vector $\mathbf{v}_1$, as it does not correspond to a vector within the basis $\mathbf{C}_k$. Consequently, during a cycle of k iterations, $k+2$ vectors must be stored in the SGMRES method. On the other hand, the memory consumption of the RB-SGMRES method is comparable to the GCR/Orthodir methods since two sets of vector bases must be stored. As a result, the storage requirement for the RB-SGMRES method involves a total of $2k+1$ vectors.

The computational requirements of the SGMRES and RB-SGMRES methods are however similar. Here again, the total number of matrix vector products in a full cycle of k iterations

Algorithm 7 One cycle of the restarted SGMRES or RB-SGMRES methods.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
for $m = 1, \ldots, k$ **do**
    if $\|\mathbf{r}_{m-1}\|/y_1 < tol$ **then** ▷ Check for convergence.
        Exit.
    if Residual based or $m = 1$ **then** ▷ New search direction.
        $\mathbf{v}_m = \mathbf{r}_{m-1}/\|\mathbf{r}_{m-1}\|$
    else
        $\mathbf{v}_m = \mathbf{c}_{m-1}$
    $\mathbf{c}_m = \mathbf{A}\mathbf{v}_m$
    for $i = 1, \ldots, m-1$ **do** ▷ Arnoldi-like iteration with modified Gram-Schmidt.
        $t_{i,m} = (\mathbf{c}_m, \mathbf{c}_i)$
        $\mathbf{c}_m = \mathbf{c}_m - t_{i,m}\mathbf{c}_i$
    $t_{m,m} = \|\mathbf{c}_m\|$
    $\mathbf{c}_m = \mathbf{c}_m/t_{m,m}$
    $q_m = \mathbf{c}_m^*\mathbf{r}_0$ ▷ Update the residual.
    $\mathbf{r}_m = \mathbf{r}_{m-1} - q_m\mathbf{c}_m$
for $m = k, \ldots, 1$ **do** ▷ Find the solution.
    $y_m = \left(y_m - \sum_{i=m+1}^{k} h_{m,i}\, y_i\right) \Big/ h_{m,m}$
for $m = 1, \ldots, k$ **do**
    $\mathbf{x}_m = \mathbf{x}_{m-1} + y_m\mathbf{v}_m$

is $k+1$. For the other computation needs, the number of operation in the m^{th} iteration is of approximately $(2m+3)n$ products and $(2m+2)n$ sums for the computation of the residual norm, the orthonormalization of the new vector $\mathbf{c}_m$ and the update of the residual. As a concequence, the main loop corresponds to $(k^2+4k)n$ products and $(k^2+3k)n$ sums for a cycle of k iterations. For the resolution of the triangular system of size k, a negligible amount of computation is needed as the problem size n is assumed much larger than the number of iterations k. And finally to the update of the solution, kn more products and sums are used. The total number of products and sums used in the SGMRES and RB-SGMRES method are of the order of $(k^2+5k)n$ and $(k^2+4k)n$ respectively. The computational requirements are then comparable to the GMRES method.

2.5 Comparison of the Krylov subspace methods

All the methods presented here correspond to the residual minimization over the Krylov subspace $\mathcal{K}_m$ for m iterations. Therefore, these methods are mathematically equivalent in exact arithmetic as previously mentioned. The memory and computational requirements are these methods can then be directly compared. Also, an important aspect related to these different implementations is the management of the error introduced be the finite precision arithmetic.

Derivation of the methods

The main attributes of these methods can be deduced by examining the selection of two critical properties inherent to these techniques. The different properties between the methods is resumed in the Figure 2.1.

Firstly, the selection of an orthonormal basis to represent the product between the search directions and the problem matrix $\mathbf{A}\mathbf{v}_m$ is required. This choice will determine the memory and computational requirements of the methods and will have a influence on the numerical stability of the methods. In the instance of the GCR and Orthodir methods, these vectors are directly orthogonalized and stored as part of this basis. Therefore, this leads to the deduction that the search directions $\mathbf{v}_m$ are $\mathbf{A}^*\mathbf{A}$-orthogonal. In contrast, for the GMRES method, the orthonormal basis corresponds to the search basis itself, incorporating the next search direction: $\mathbf{V}_{m+1}$. And for the (RB-)SGMRES method, the orthonormal basis is defined by a basis $\mathbf{C}_m$, which comprises the vectors $\mathbf{A}\mathbf{v}_m$ that have been orthonormalized with respect to each other. The key distinction from the GCR and Orthodir methods lies in the fact that the search directions $\mathbf{v}_m$ are not made $\mathbf{A}^*\mathbf{A}$-orthogonal to one another and therefore the vector $\mathbf{c}_m$ is not equal to $\mathbf{A}\mathbf{v}_m$.

Then, the second important property corresponds to the definition of the next search direction. Indeed, it might be possible to define it from the last search direction or from the residual vector. This choice will however not change the consumption of the method considered, but it will only influence its numerical stability. This property correspond to the difference between the Orthodir/GCR and SGMRES/RB-SGMRES methods respectively. However, for the GMRES method, the next search direction must necessarily be defined from the last direction.

Numerical stability

When the resolution employs a large number of directions, the orthogonalization process can accumulate round off errors in finite precision arithmetic. This can lead to a reduction of

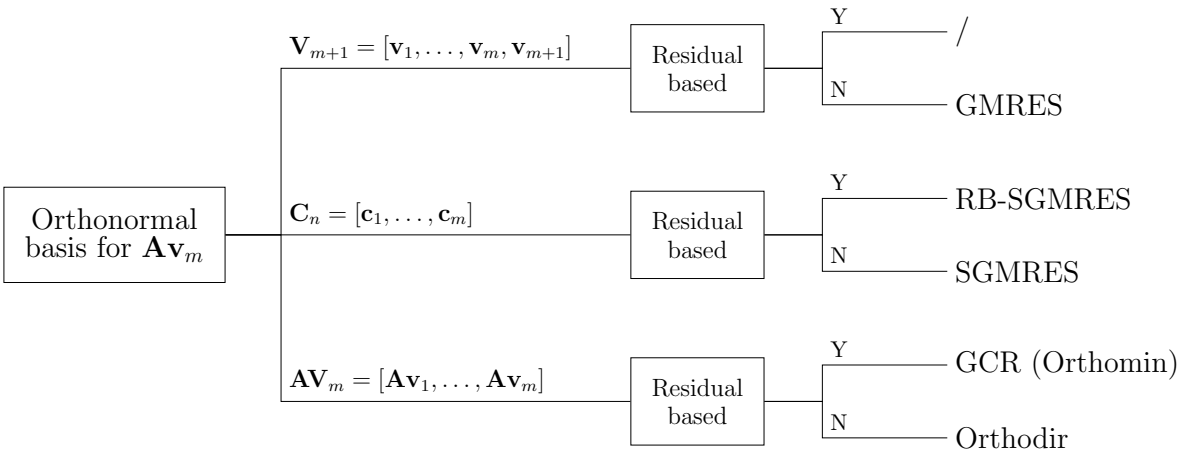


Figure 2.1: Properties comparison between GMRES, SGMRES and GCR.

the orthogonality between vectors and eventually to numerical instabilities. Such a situation may arise when a substantial number of iterations are needed for convergence. This can occur, for example, when dealing with a sizable problem or if the convergence tolerance, denoted as *tol*, is set to a low value. In practice, these instabilities are reduced by using the modified Gram-Schmidt orthogonalization, but they can still have an influence. Furthermore, enforcing the algorithms to restart after a small number of iterations will result in a reduced usage of directions, consequently mitigating the accumulation of error. However, it is important to note that this approach could potentially decrease the convergence of the resolution. The accumulation of errors is managed differently by each of the methods presented here, resulting in distinct numerical stability properties.

The solver exhibiting more favorable behavior in term of stability is the GMRES method as it was demonstrated in [38] and [39] for its implementation with the Householder reflections or the modified Gram-Schmidt process. Conversely, the Orthodir and SGMRES method may exhibit bad numerical behavior as emphasized in [32]. Indeed, this reference illustrates the limitations of these methods on specific examples and it highlights the superiority in term of stability of their residual based counterparts, specifically GCR and RB-SGMRES. Another observation from this reference is that the GCR and the RB-SGMRES methods behave almost equally to the GMRES method. This then justifies the use of either of these three methods when the number of directions used becomes large.

Memory and computational requirements

An essential aspect of these methods lies in their memory and computational consumption. In order to compare them, a complete cycle of k iterations is examined. Notably, it is assumed that the size of the considered problem n is significantly larger then the number of iteration k and therefore all the requirements not related to a vector or a vector operation can be neglected. For instance, the need to store and solve a k by k triangular system in the GMRES-like methods can be neglected. Consequently, the main contributions to the storage and operation count are resumed in the Table 2.1. It's important to highlight that the memory requirements associated to the matrix $\mathbf{A}$, the right-hand side $\mathbf{b}$, and the solution $\mathbf{x}_k$ are not taken into account in this analysis. Additionally, operations associated with matrix-vector products are treated separately from other operations.

Firstly, as indicated by Table 2.1, the GMRES method stands out as the most appealing

Number of	**GMRES**	**SGMRES**	**RB-SGMRES**	**GCR/Orthodir**
scalars stored	$(k+1)n$	$(k+2)n$	$(2k+1)n$	$(2k+1)n$
matrix vector products	$k+1$	$k+1$	$k+1$	$k+1$
scalar multiplications	$(k^2+4k)n$	$(k^2+5k)n$	$(k^2+5k)n$	$(\frac{3}{2}k^2+\frac{9}{2}k)n$
scalar additions	$(k^2+3k)n$	$(k^2+4k)n$	$(k^2+4k)n$	$(\frac{3}{2}k^2+\frac{5}{2}k)n$

Table 2.1: Approximate memory and computation requirements of one cycle of the restarted Krylov subspace methods for k iterations in one cycle.

option, accumulating all of the advantages. Indeed, it requires the least amount of memory storage and the fewest computational operations in comparison to all other methods. Moreover, since it exhibit the highest stability in the context of finite precision arithmetic, the GMRES solver is the method of choice for solving a single linear problem. Secondly, the requirements of the simpler GMRES are only slightly more expensive than those of GMRES, however it is more prone to lead to some numerical instabilities. In contrast, the RB-SGMRES method offers a better stability, albeit necessitating nearly twice the memory allocation of the classic GMRES method. Finally, the most resource demanding solvers, both in terms of storage and operation count, are the GCR and Orthodir methods. They require nearly twice the memory capacity of GMRES and roughly fifty percent more operations. Consequently, given the enhanced stability of GCR, there is no interest to opt for Orthodir over GCR.

2.6 Improvement techniques for Krylov subspace methods

As these methods find application across a wide range of application, various enhancements have been developed over the years to refine their effectiveness. The central objective has consistently been to improve the convergence of these methods. In this quest, the development of preconditioned solvers emerged as a valuable approach. The challenge is then to identifying an appropriate preconditioner. Some methods such a flexible GMRES (FGMRES) [40], GMRES recursive (GMRESR) [41] and GCR with inner orthogonalization (GCRO) [42] proposed to use another iterative methods as a preconditioner. This leads to the creation of nested algorithms, consisting of an outer and an inner iterative method. Numerous additional algorithms have also been developed. The main goal of these methods is to decrease the number of stored directions, consequently reducing the computational workload necessary for convergence, while achieving the highest possible level of convergence.

Another widely used category of algorithms aims to achieve similar objectives by improving the convergence of the restarted techniques. Indeed, the restarting of the algorithm can help mitigate consumption, however this might result in the degradation of the convergence of the method or even lead to stagnation. Subsequently, certain approaches were developed to reuse a carefully chosen subspace across cycles, with the intention of enhancing the convergence process. Certain techniques directly preserve specific search directions, as seen in GCRO with outer truncation (GCROT) [43]. Alternatively, some methods involve approximating a

few matrix eigenvectors using the precedent search space, and employing them to enhance the convergence of subsequent cycles. This strategy is applied in algorithms like restarted GMRES augmented with eigenvector (GMRES-E) [44], GMRES with deflated restarting (GMRES-DR) [45], simpler GMRES-DR [46], and GCRO with deflated restarting (GCRO-DR) [47].

This class of improved restarted methods was initially developed to enhance the convergence of the restarted cycles when solving a single linear problem. However, these techniques can also be exploited to improve the convergence of the methods when solving a sequence of linear problems with different right hand sides. The effectiveness of these approaches has been established in studies [47], [48], and [1]. Therefore, some recycling Krylov subspace methods will be investigated in the context of full waveform inversion, with the goal of enhancing the overall convergence of the process. Also, when solving multiple systems of equations that have different right hand sides, an opportunity arises to leverage the parallel computation and reuse of the search directions across these systems. This concept is effectively implemented within the block recycling Krylov subspace methods as described in [49] and [50]. These methods will not be covered here, although they would be correspond to a natural extension of the current work.

Chapter 3

Krylov subspace recycling

As it was mentioned in the previous chapter, variations of Krylov subspace methods where developed in order to accelerate the convergence rate of the solutions by recycling some information between the restarts of the methods or between systems with different right hand sides. These methods will be of a great interest in the context of a full waveform inversion since it consists in the resolution of multiple linear systems with different right hand sides following the explanation provided in Chapter 1. The idea behind these techniques is summarized in [51] and consists of selecting a portion of the Krylov subspace from a resolution and then reusing it in the subsequent resolution. If the recycled subspace is well chosen, then the resulting search space of the next resolution could provide a better approximation of the solution, and therefore the convergence of the method might be improved. Such methods have already been shown to be effective in many scientific and engineering applications, and they continue to be an area of active research [47], [48] and [1]. The recycled subspace can be represented in various ways. For instance, it can be depicted through search directions [43] or approximated eigenvectors [47]. In the current work, only the recycling of search directions without further processing will be examined. This simple way of choosing the augmentation space is already sufficient to compare the different recycling Krylov methods and if the whole Krylov subspace is recycled, it should provide the best results possible. Other, more complex, subspace augmentations exist such as the very popular deflated restarting [45] [46] [47], but they will not be considered here. The utilization of such methods might potentially reduce the consumption of the recycling methods and could be addressed in future research.

To harness the potential of the recycled subspace, some modifications are required in the Krylov iterative solvers. More generally, it is imperative to develop variations of the Krylov subspace methods to take advantage of supplied directions that are expected to be carefully chosen. Such a modification for the conjugate gradient method was proposed in [52]. This paper introduces a first method, InitCG, which improves the initial guess by using the provided directions. To further enhance the convergence, a second approach, AugCG, introduces an orthogonality constraint between the given directions and the computed ones. This constraint mirrors the one between the search directions themselves. As a result, the supplied directions are treated as if they were directions computed within this method. Consequently, the technique identifies the approximate solution through the exploration of a search subspace formed by augmenting the Krylov subspace with the provided directions. This explains why this method is called augmented CG. The extension of this technique to the minimization methods is possible and the resulting algorithms will then be referred as augmented methods.

The augmentation of the GCR method is rather straightforward due to the simple update rule and orthogonal property. The only additional requirements in comparison to the classical algorithm are to computation of the directions multiplied by the matrix, the enforcement of the orthogonality constraint between all the directions considered and the update of the solution and the residual at the beginning. For the simpler GMRES method a similar methodology could be used but with the properties of the SGMRES method. Then, the augmentation directions will undergo the Arnoldi-like process in order to form a basis that is capable of representing all augmentation directions multiplied by the problem matrix. Also, the update rule for the residual must also be used for each augmentation direction. In the end a larger triangular system will be solved to find the coefficient used to update the approximate solution with both the augmentation and the computed directions. An augmented method with the same properties as the GMRES method would be attractive, however this will not be possible. Indeed, the GMRES method is based on the fact that the multiplication of a search direction $\mathbf{A}\mathbf{v}_m$ by the problem matrix must be representable by a linear combination of the corresponding next search basis $\mathbf{V}_{m+1}$. However, if the search directions are augmented directions, this is no longer possible as it will be shown in Section 3.2.1. Therefore an augmentation process based on the classic GMRES method is not possible. However, building on what is done for SGMRES, an augmentation technique for GMRES can be provided. In the following, the augmentations methods for GCR, GMRES and simpler GMRES will first be presented. For these algorithms, a notation closely related to the one used in the derivation of the algorithms in Chapter 2 will be employed to facilitate the implementation.

Once the augmentation methods are described, they can make use of any provided search directions. For an improvement in the convergence behavior of the methods to be expected, these directions must be relevant with respect to the system that is aimed to be solved. The simplest way of choosing these augmentation directions is to reuse directions computed in precedent resolutions. On top of being a simple procedure, these directions will exhibit certain properties arising from their origin in the resolution of a Krylov iterative method. For instance, the recycled directions will already have the properties required by the resolution method. Also the multiplication of these directions with the matrix will be easily available. Subsequently, this will lead to some simplifications within the augmentation techniques and it will result in methods that will be referred to as methods with subspace recycling. Thus in term of matrix vector product, the recycling of directions will not require any additional matrix vector product. This is a very appealing property since the matrix vector product are expected to be expensive compared to all the other resolution operations (orthogonalizations, normalizations, updates, ..). When solving a single system, the augmentation can be used at each restart, then recycling some of the directions of precedent cycles. If all directions are recycled, then it would correspond to the same sequence of solutions as the resolution without restarting. On the other hand, when multiples systems with different right hand sides are solved, then the augmentation can be used at the beginning of every resolution. It will then be possible to use the recycled directions both at the beginning of each system resolution and at the beginning of each restart.

3.1 Augmented GCR method

3.1.1 Augmentation with arbitrary directions

The augmented GCR method with arbitrary directions can be perceived as a straightforward expansion of AugCG as delineated in [52]. The approach is detailed in [53] when augmenting directions are recycled from prior resolutions. However, some additional efforts

are necessary due to the arbitrary nature of the augmented directions. Let us represent these p augmentation directions as a matrix $\mathbf{W}_p$ were each of its columns is a vector. These augmentation vectors must be $\mathbf{A}^*\mathbf{A}$-orthogonal and scaled in order to have the same properties as the search directions of the GCR method. Once these properties are imposed, it will result in a basis that will be noted $\mathbf{V}_p$ similarly to the notation in Chapter 2. This will allow to use the same notation for the updates rules as defined in (2.5) and for the orthogonalization process, e.g. (2.7). In augmented GCR, the fist step is then to form the basis $\mathbf{V}_p$ from the augmentation directions $\mathbf{W}_p$. Then the residual is minimized over the augmentation space by using the basis and the update rules as defined in (2.5) for each direction. Once the residual and the approximate solution have been update with respect to the augmentation space, the standard GCR method can be employed with an additional orthogonality requirement. The m search directions computed by the GCR method will then be noted $\{\mathbf{v}_{p+1}, \dots, \mathbf{v}_{p+m}\}$. The overall basis representing the union of the augmentation subspace and the Krylov subspace is defined as the matrix $\mathbf{V}_{p+m}$, where the first p directions correspond to the augmentation directions and the m other correspond to the new computed search directions. Then any direction $\mathbf{v}_i$ must be $\mathbf{A}^*\mathbf{A}$-orthogonal to $\mathbf{V}_{i-1}$ in order to fit in the definition of the GCR method. Since the vectors $\mathbf{v}_i$ can represent either the augmentation directions or the new search direction, then all direction considered must be $\mathbf{A}^*\mathbf{A}$-orthogonal.

This augmentation method is described in the Algorithm 8. Its main disadvantage is that the augmentation of one vector corresponds to the same requirements as executing one iteration of the GCR method. Indeed, a matrix vector product is needed as well as the orthogonalization to all the orther directions. Therefore, the augmentation directions must have a large impact on the convergence in order to justify their use instead of one iteration of the GCR method.

Algorithm 8 Augmentation of the GCR method with arbitrary directions.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
for $j = 1, \dots, p$ **do**
    $\mathbf{v}_j = \mathbf{w}_j$ ▷ $\mathbf{A}^*\mathbf{A}$-orthogonalization of augmentation directions.
    $(\mathbf{A}\mathbf{v})_j = \mathbf{A}\mathbf{v}_j$
    for $i = 1, \dots, j-1$ **do**
        $\beta_{i,j} = ((\mathbf{A}\mathbf{v})_j, (\mathbf{A}\mathbf{v})_i)$
        $\mathbf{v}_j = \mathbf{v}_j - \beta_{i,j}\mathbf{v}_i$
        $(\mathbf{A}\mathbf{v})_j = (\mathbf{A}\mathbf{v})_j - \beta_{i,j}(\mathbf{A}\mathbf{v})_i$
    $\beta_{j,j} = \|(\mathbf{A}\mathbf{v})_j\|$
    $\mathbf{v}_j = \mathbf{v}_j/\beta_{j,j}$
    $(\mathbf{A}\mathbf{v})_j = (\mathbf{A}\mathbf{v})_j/\beta_{j,j}$
    $y_j = (\mathbf{r}_{j-1}, (\mathbf{A}\mathbf{v})_j)$ ▷ Update the solution and the residual.
    $\mathbf{x}_j = \mathbf{x}_{j-1} + y_j\mathbf{v}_j$
    $\mathbf{r}_j = \mathbf{r}_{j-1} - y_j(\mathbf{A}\mathbf{v})_j$
Classical GCR method with the $\mathbf{A}^*\mathbf{A}$-orthogonalization with respect to all $p+m$ directions.

3.1.2 Augmentation with subspace recycling

As it was described, when solving restarted cycles or a sequence of linear systems, one simple space augmentation strategy would be to reuse a part of the Krylov subspace of the precedent resolution. This offers two significant benefits: the directions forms already a $\mathbf{A}^*\mathbf{A}$-

orthogonal basis and their multiplication with the problem matrix are also readily available. The only requirement for the augmented directions is to form a $\mathbf{A}^*\mathbf{A}$-orthogonal basis, this implies that all directions or any subset of a Krylov subspace can be recycled. By storing both the vectors $\mathbf{v}_p$ and $\mathbf{A}\mathbf{v}_p$ from one resolution to another, no more operation is needed and the minimization over the augmentation subspace can be done right away.

The augmentation of the GCR method with a recycled subspace is described in Algorithm 9. The augmentation with a precedent Krylov subspace can then be done without any additional matrix vector product or orthogonalization of the recycled directions. Then, to proceed in the GCR method, the only additional cost is located in the orthogonalization of each new search direction. Indeed, one must make sure that the new direction is $\mathbf{A}^*\mathbf{A}$-orthogonal to all other directions including those originating from the recycled subspace. The limitations of this strategy are the same as for the classical GCR method. As the total number of stored directions grows, the memory demand increases linearly and the number of operation due to the orthogonalization increases in a quadratic manner. Fortunately, the workload linked to the matrix vector products is expected to remain unchanged. Since these operations correspond to a majority of the computation requirement, then the augmented GCR method with subspace recycling will not result in considerably higher costs than the classical GCR method.

Algorithm 9 Augmentation of the GCR method with subspace recycling.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
for $j = 1, \ldots, p$ **do** ▷ Update the solution and the residual.
    $y_j = (\mathbf{r}_{j-1}, (\mathbf{A}\mathbf{v})_j)$
    $\mathbf{x}_j = \mathbf{x}_{j-1} + y_j \mathbf{v}_j$
    $\mathbf{r}_j = \mathbf{r}_{j-1} - y_j (\mathbf{A}\mathbf{v})_j$
Classical GCR method with the $\mathbf{A}^*\mathbf{A}$-orthogonalization with respect to all $p+m$ directions.

3.2 Augmented GMRES-type methods

3.2.1 Augmentation with arbitrary directions

The augmentation of methods with a structure resembling that of the GMRES method is elaborated in [54] for any arbitrary set of directions. This augmentation approach will initially be introduced within the context of the simpler GMRES method. Using this as a foundation, the extension of this approach to the GMRES method will be detailed. Subsequently, these augmentation techniques will be simplified to enable the recycling of Krylov subspaces. However, selecting a subset of a Krylov subspace will result in additional operations, which will also be explored in the following.

Augmented simpler GMRES method

Again, the p augmentation directions are described as a basis $\mathbf{W}_p$. These basis vectors are then multiplied by the matrix $\mathbf{A}$ and using the Arnoldi-like process of the SGMRES method, a orthonormal basis $\mathbf{C}_p$ is formed. This will allow to express the vectors $\mathbf{A}\mathbf{w}_i$ from this basis $\mathbf{C}_i$ in the same manner as in the SGMRES method. Once all the augmentation directions are processed, one finds a relation similar to (2.17) but for the vectors $\mathbf{W}_p$. As it was done for the augmented GCR method in Section 3.1, a notation close to the one used in the derivation of the classical method will be used to simplify the implementation. Therefore, here the augmentation

vectors will also be denoted by $\mathbf{V}_p = \mathbf{W}_p$. The Arnoldi-like process will then produce a basis $\mathbf{C}_p$ and a matrix $\mathbf{T}_p$ such that

$$\mathbf{A}\mathbf{V}_p = \mathbf{C}_p\mathbf{T}_p, \tag{3.1}$$

where again $\mathbf{T}_p$ is a p by p upper triangular matrix of the orthogonalization coefficients. The use of the augmentation search directions will lead to a decrease of the residual norm since the augmented SMGES method will minimize the residual norm over the augmented Krylov subspace. This effect on the initial residual $\mathbf{r}_0$ can equivalently be expressed as is projection on the complement space of the basis $\mathbf{C}_p$ as it is the case in the classical implementation of simpler GMRES in (2.20). The updated residual with respect to the p augmentation direction is defined as

$$\mathbf{r}_p = \mathbf{r}_0 - \mathbf{C}_p\mathbf{C}_p^*\mathbf{r}_0. \tag{3.2}$$

The SGMRES method will then used this residual as the initial residual to start the resolution. In that way, the first search direction will be defined as: $\mathbf{v}_{p+1} = \mathbf{r}'_0/\|\mathbf{r}'_0\|$. Then, all the computed search direction $\mathbf{v}_{p+m}$ will undergo the Arnoldi-like process with respect to the basis $\mathbf{C}_{p+m}$. In other words, the corresponding vector $\mathbf{c}_{p+m}$ will be orthogonalized with respect to all the vectors of $\mathbf{C}_{p+m}$ thus leading to the last column of the matrix $\mathbf{T}_{p+m}$ composed of $p+m$ elements. The resulting matrix $\mathbf{T}_{p+m}$ will therefore still be upper triangular and the relation due to the Arnoldi-like process will still hold:

$$\mathbf{A}\mathbf{V}_{p+m} = \mathbf{C}_{p+m}\mathbf{T}_{p+m} \quad \text{with} \quad \mathbf{T}_{p+m} = \left(\begin{array}{ccc|ccc} t_{1,1} & \dots & t_{1,p} & t_{1,p+1} & \dots & t_{1,p+m} \\ & \ddots & \vdots & \vdots & \ddots & \vdots \\ & & t_{p,p} & t_{p,p+1} & \dots & t_{p,p+m} \\ \hline & & & t_{p+1,p+1} & \dots & t_{p+1,p+m} \\ & & & & \ddots & \vdots \\ & & & & & t_{p+n,p+m} \end{array}\right). \tag{3.3}$$

The rest of the algorithm stay the same as if the first p directions corresponds to search direction computed in iterations of the classical algorithm. The last step in the augmented SGMRES method correspond then to update the initial guess $\mathbf{x}_0$ by minimizing the residual vector $\mathbf{r}_m$ over the augmented Krylov subspace. Again, by using similar notation as in Charpter 2, this can be described as:

$$\text{Find} \quad \mathbf{x}_{p+m} = \mathbf{x}_0 + \mathbf{V}_{p+m}\mathbf{y}_{p+m} \qquad \text{such that} \quad \mathbf{y}_{p+m} = \underset{\tilde{\mathbf{y}}_{p+m}}{\arg\min} \|\mathbf{r}_{p+m}\|. \tag{3.4}$$

The minimization will be performed by exploiting the relation (3.1) and this will again correspond to the resolution of an upper triangular system:

$$\mathbf{T}_{p+m}\tilde{\mathbf{y}}_{p+m} = \mathbf{q}_{p+m} \quad \text{where} \quad \mathbf{q}_{p+m} = \mathbf{C}_{p+m}^*\mathbf{r}_0. \tag{3.5}$$

Then, in order to find the coefficient $\mathbf{y}_{p+m}$, one must first compute the vector $\mathbf{q}_{p+m} = \mathbf{C}_{p+m}^*\mathbf{r}_0$.

The augmentation of the simpler GMRES method with arbitrary direction is described in Algorithm 10. Similarly to the augmented GCR method, the requirements for handling one arbitrary augmentation direction are the same as those needed for one iteration of the SGMRES method.

Algorithm 10 Augmentation of the GMRES-type method with arbitrary directions.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
for $j = 1, \ldots, p$ **do**
    $\mathbf{c}_j = \mathbf{A}\mathbf{v}_j$ ▷ Compute the $\mathbf{C}_p$ basis.
    for $i = 1, \ldots, j-1$ **do**
        $t_{i,j} = (\mathbf{c}_j, \mathbf{c}_i)$
        $\mathbf{c}_j = \mathbf{c}_j - t_{i,j}\mathbf{c}_i$
    $t_{j,j} = \|\mathbf{c}_j\|$
    $\mathbf{c}_j = \mathbf{c}_j / t_{j,j}$
    $q_j = \mathbf{c}_j^* \mathbf{r}_0$ ▷ Update the residual.
    $\mathbf{r}_j = \mathbf{r}_{j-1} - q_j \mathbf{c}_j$

Classical GMRES-type method with the additional orthogonalization with respect to the basis $\mathbf{C}_p$ and the resolution of a triangular system of size $p + m$.

Augmented GMRES method

Unlike the augmented GCR and SGMRES methods, the augmentation of the GMRES method cannot be deduced solely from the attributes of its traditional implementation. If one aims to augment the Krylov subspace using p vectors $\mathbf{w}_p$, while imposing a relationship similar to the one established by the Arnoldi iteration (2.8), the following conditions must be satisfied:

$$\mathbf{A}[\mathbf{W}_p \mathbf{V}_m] = [\mathbf{W}_p \mathbf{V}_{m+1}]\bar{\mathbf{H}}_{p+m}. \tag{3.6}$$

Here, $\mathbf{H}_{p+n}$ is a matrix with dimensions $p + m + 1$ by $p + m$, following a Hessenberg structure similar to the classical GMRES method. This relation would impose significant constraints on the augmentation vectors, as each vector $\mathbf{w}_i$ multiplied by the matrix $\mathbf{A}$ must be representable using the basis $\mathbf{W}_{i+1}$ or the basis $[\mathbf{W}_p \mathbf{V}_1]$ for the vector $\mathbf{w}_p$. While this property may hold for the majority of vectors, such as when recycling vectors from a previous resolution, the final vector will not satisfy it in general. This is because the subsequent search vector must be defined based on the residual vector, and indeed it is unlikely that the vector $\mathbf{A}\mathbf{w}_p$ corresponds to a linear combination of the augmentation vectors $\mathbf{W}_p$ and the residual. Therefore, the augmentation of the GMRES method must be done in another way.

The GMRES method can then be augmented by using a methodology similar to the augmented simpler GMRES method. This approach will require to store a basis $\mathbf{C}_p$ and an upper triangular matrix $\mathbf{T}_p$ to express the multiplication of the augmentation vectors $\mathbf{V}_p = \mathbf{W}_p$ by the matrix $\mathbf{A}$ as it represented in (3.1). Then, the initial residual vector update with respect to the augmentations vectors. This is done by projection it in the complemented space of the basis $\mathbf{C}_p$ as described in (3.2). The GMRES method is initiated with this projected residual and the Arnoldi process will also orthogonalize the search direction with respect to the basis $\mathbf{C}_p$. If the set of computed search directions is represented by the basis $\mathbf{V}_{p+1:p+m}$, this will results in the relation:

$$\mathbf{A}\mathbf{V}_{p+m} = \mathbf{A}[\mathbf{V}_p \mathbf{V}_{p+1:p+m+1}] = [\mathbf{C}_p \mathbf{V}_{p+1:p+m+1}]\bar{\mathbf{H}}_{p+m} \tag{3.7}$$

where the matrix of coefficient $\bar{\mathbf{H}}_{p+m}$ is a $p + m + 1$ by $p + m$ matrix with an hybrid structure between an upper triangular and an Hessenberg matrix. This is due to the fact that the first

part of the matrix $\bar{\mathbf{H}}_{p+m}$ results from the relation (3.1) while the second is constructed by the Arnoldi iteration. This structure can then be represented as

$$\bar{\mathbf{H}}_{p+m} = \left(\begin{array}{ccc|ccc} t_{1,1} & \dots & t_{1,p} & h_{1,p+1} & \dots & h_{1,p+m} \\ & \ddots & \vdots & \vdots & \ddots & \vdots \\ & & t_{p,p} & h_{p,p+1} & \dots & h_{p,p+m} \\ & & & h_{p+1,p+1} & \dots & h_{p+1,p+m} \\ & & & h_{p+2,p+1} & \dots & h_{p+2,p+m} \\ & & & & \ddots & \vdots \\ & & & & & h_{p+m+1,p+m} \end{array}\right) \tag{3.8}$$

The subsequent application of the Givens rotations will not affect the first part of this matrix and therefore this matrix can still be transformed into an upper triangular matrix.

At the end of the GMRES method, the approximate solution is again found by minimizing the residual norm as in (3.4). Although, the derivation of the residual norm will be different since the relation (3.7) differ sightly:

$$\begin{aligned} \mathbf{r}_{p+m} = \mathbf{r}_0 - \mathbf{A}\mathbf{V}_{p+m}\mathbf{y}_{p+m} &= \mathbf{r}_0 - [\mathbf{C}_p \mathbf{V}_{p+1:p+m+1}]\bar{\mathbf{H}}_{p+m}\mathbf{y}_{p+m} \\ &= [\mathbf{C}_p \mathbf{V}_{p+1:p+m+1}](\mathbf{q}_{p+1} - \bar{\mathbf{H}}_{p+m}\mathbf{y}_{p+m}). \end{aligned} \tag{3.9}$$

The vector $\mathbf{q}_{p+m}$ is then defined from the the multiplication of the initial residual with the basis in the right hand side of (3.7): $\mathbf{q}_{p+m} = [\mathbf{C}_p \mathbf{V}_{p+1:p+m+1}]^* \mathbf{r}_0$. This vector is similarly defined as in the simpler GMRES method but with another basis. Furthermore, since this basis has an hybrid structure, then the vector $\mathbf{q}_{p+m}$ will also have a special structure. Its p first components will actually correspond to the orthogonalization coefficients used when the residual is projected. This is similar to the simpler GMRES method. Then, the first search direction computed $\mathbf{v}_1$ will be defined from the projected residual. All the following vector will then be orthogonalized with respect to this vector and also to all the augmentation vectors. This implies that all the computed search directions except the first one will be orthogonal to the initial residual vector. This time, this is similar to the GMRES method. This can be summarized as

$$q_i = \begin{cases} (\mathbf{c}_i, \mathbf{r}_0) & \text{for } i < p+1, \\ \|\mathbf{r}_p\| & \text{for } i = p+1, \\ 0 & \text{for } i > p+1. \end{cases} \tag{3.10}$$

In order to minimize the residual, Givens rotations will be used such as in the GMRES method. With m rotations, the matrix $\bar{\mathbf{H}}_{p+m}$ can be transformed into a fully upper triangular matrix $\bar{\mathbf{R}}_{p+m}$. The Givens rotations will also needed to be applied to the vector $\mathbf{q}_m$, thus resulting a triangular system that can be solved to minimize the residual:

$$\bar{\mathbf{g}}_{p+m} = \mathbf{\Omega}_m \dots \mathbf{\Omega}_1 \mathbf{q}_{p+m} \text{ and } \bar{\mathbf{R}}_{p+n} = \left(\begin{array}{ccc|ccc} t_{1,1} & \dots & t_{1,p} & h_{1,p+1} & \dots & h_{1,p+n} \\ & \ddots & \vdots & \vdots & \ddots & \vdots \\ & & t_{p,p} & h_{p,p+1} & \dots & h_{p,p+n} \\ & & & r_{p+1,p+1} & \dots & r_{p+1,p+n} \\ & & & 0 & \ddots & \vdots \\ & & & & \ddots & r_{p+n,p+n} \\ & & & & & 0 \end{array}\right). \tag{3.11}$$

The minimization of the residual norm can then be achieved by solving a triangular system $\mathbf{R}_{p+m}\mathbf{y}_{p+m} = \mathbf{g}_{p+m}$, where the last rows of $\bar{\mathbf{R}}_{p+m}$ and $\bar{\mathbf{g}}_{p+m}$ are removed. Additionally, similar to the approach used in the classic GMRES method, the residual norm will be given by the absolute value of the last element of $\bar{\mathbf{g}}_{p+m}$.

This description of the augmented GMRES method fits into the interpretation outlined in Algorithm 10. This augmentation technique is applicable to both the GMRES and the simplified GMRES methods and the slight distinctions between these approaches are hidden in line 11. However, these disparities essentially mirror the differences inherent in the classical versions of these methods, making them readily deducible.

3.2.2 Augmentation with subspace recycling

Once more, the recycling of the Krylov subspace offers the possibility of introducing simplifications into the augmentation methods. Indeed, if the basis $\mathbf{C}_p$ and the matrix $\mathbf{T}_p$ are inferred from a prior resolution, there is no requirement for their explicit computation. One necessity is however to compute the vector $\mathbf{q}_p$, as it must be derived from the initial residual of the specific problem. Subsequently, the residual is also updated before initiating the GMRES-type method. The resulting augmentation technique is resumed in Algorithm 11.

It remains necessary to setermine a method to acquire the basis $\mathbf{V}_p$ and $\mathbf{C}_p$, along with the matrix $\mathbf{T}_p$, at the end of a resolution process. In the case where the entire Krylov subspace is recycled, the task is straightforward for the SGMRES method, as these elements are immediately accessible. For the GMRES method, these elements are not directly provided. Nevertheless, they can be derived by combining the search basis vectors with the Givens rotations:

$$\mathbf{A}\mathbf{V}_m = \mathbf{V}_{m+1}\bar{\mathbf{H}}_m = \underbrace{\mathbf{V}_m\boldsymbol{\Omega}_1^*\dots\boldsymbol{\Omega}_m^*}_{\bar{\mathbf{C}}_m}\underbrace{\bar{\mathbf{R}}_m}_{\bar{\mathbf{T}}_m} \tag{3.12}$$

Consequently, an extra step becomes necessary in the context of the recycled GMRES approach: computing the basis $\mathbf{C}_m$ based on $\mathbf{V}_{m+1}$. Fortunately, this additional task is expected to be relatively cheap, as it involve only vector operations without the need for matrix vector products, making it efficient in terms of computational resources. The overall computational requirement for this step comprises $(4m+2)n$ products and $(2m+1)n$ additions. Furthermore, it's important to note that the additional storage of the basis $\mathbf{C}_m$ contribute to an increased need for memory capacity. This procedure is outlined at the end of Algorithm 11.

The last aspect that needs clarification regarding the recycling technique is the process of selecting only a subset of the Krylov subspace rather than using the entire subspace. This won't be a straightforward task, as the representation of the vector $\mathbf{A}\mathbf{v}_i$ necessitates the presence of all previous vectors, as illustrated in (3.1) or (3.7). Hence, this matter will be addressed in the subsequent section.

3.2.3 Partial subspace recycling

To selectively recycle a portion of a subspace, it becomes essential to devise an approach that eliminates the dependance on the directions not chosen. The upcoming section will introduce a technique designed to disentangle dependence from a single direction. While this method pertains to just one direction, its repeated application is sufficient to end up with the desired subspace.

Algorithm 11 Augmentation of the GMRES-type method with subspace recycling.

Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
for $j = 1, \ldots, p$ **do** ▷ Update the residual.
    $q_j = \mathbf{c}_j^* \mathbf{r}_0$
    $\mathbf{r}_j = \mathbf{r}_{j-1} - q_j \mathbf{c}_j$
Classical GMRES-type method with the additional orthogonalization with respect to the basis $\mathbf{C}_p$ and the resolution of a triangular system of size $p + m$.
if GMRES **then**
    $\mathbf{c}_{p+1} = \mathbf{v}_{p+1}$ ▷ Compute the basis $\mathbf{C}_{p+m}$.
    for $i = 1, \ldots, m$ **do**
        $\mathbf{c}_{p+i+1} = \mathbf{v}_{p+i+1}$
        $\mathbf{c}_{p+i} = \overline{c}_i \mathbf{c}_{p+i} + \overline{s}_i \mathbf{c}_{p+i+1}$ and $\mathbf{c}_{p+i+1} = -s_i \mathbf{c}_{p+i} + c_i \mathbf{c}_{p+i+1}$
    $\mathbf{c}_{p+m} = \overline{c}_m \mathbf{c}_{p+m} + \overline{s}_m \mathbf{c}_{p+m+1}$

For instance, if the j^{th} direction is not selected for the recycling, then all the directions after that one must be modified in order to be able to express their multiplication by the matrix $\mathbf{A}$ without that j^{th} direction. This will allow us to erase the j^{th} column and row of $\mathbf{T}_p$ in the relation describing the vectors $\mathbf{A}\mathbf{V}_p$. This dependance is clear when looking at the following relation:

$$\mathbf{A}\mathbf{V}_p = \mathbf{C}_p \mathbf{T}_p. = \mathbf{C}_p \begin{pmatrix} t_{1,1} & \cdots & t_{1,j} & \cdots & t_{1,p} \\ & \ddots & \vdots & \ddots & \vdots \\ & & t_{1,j} & \ddots & \vdots \\ & & & \ddots & \vdots \\ & & & & t_{p,p} \end{pmatrix}. \tag{3.13}$$

More specifically, the vector $\mathbf{A}\mathbf{v}_p$ can be explicitly expressed from the basis $\mathbf{C}_m$. It will allow us to remove the j^{th} vector in the representation of the vector $\mathbf{v}_p$ by using the following transformation:

$$\mathbf{A}\mathbf{v}_p = \sum_{i=1}^{p} t_{i,p} \mathbf{c}_i \qquad \Rightarrow \qquad \mathbf{v}'_p = \mathbf{v}_p - \frac{t_{m,p}}{t_{m,m}} \mathbf{v}_i. \tag{3.14}$$

This vector update will then modify the representation of the vector $\mathbf{A}\mathbf{v}_p$. To integrate this changes, the elements of the matrix $\mathbf{T}_p$ must be modify and this will indeed lead to nullify the j^{th} element of the p^{th} column:

$$\mathbf{A}\mathbf{v}'_p = \mathbf{A}\mathbf{v}_p - \frac{t_{j,p}}{t_{j,j}} \mathbf{A}\mathbf{v}_p = \sum_{i=1}^{j-1} \underbrace{\left[t_{i,p} - \frac{t_{j,p}}{t_{j,j}} t_{i,j} \right]}_{t'_{i,p}} \mathbf{c}_i. + \sum_{i=j+1}^{p} t_{i,p} \mathbf{c}_i. \tag{3.15}$$

These two updates must be applied to all the vectors from $j + 1$ to p. The representation of the vectors $\mathbf{A}\mathbf{V}_p$ is updated in such a way that it becomes possible to remove the vectors $\mathbf{v}_j$, $\mathbf{c}_j$ and the j^{th} column and row of the matrix $\mathbf{T}_p$ from the the relation (3.13):

$$\mathbf{T}'_p = \left(\begin{array}{ccccccc} t_{1,1} & \dots & t_{1,m-1} & t_{1,m} & t'_{1,m+1} & \dots & t'_{1,p} \\ & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & t_{1,m-1} & \vdots & t'_{m-1,m+1} & \dots & t'_{m-1,p} \\ \hline & & & t_{m,m} & & & \\ & & & & t_{m+1,m+1} & \dots & t_{m+1,p} \\ & & & & & \ddots & \vdots \\ & & & & & & t'_{p,p} \end{array}\right). \tag{3.16}$$

The removal of the j^{th} direction from the subspace is done in a manner that preserves the desirable properties of the other directions. Consequently, the vectors $\mathbf{c}_i$ will remain orthogonal to each other and maintain their normalization. This ensures that the remaining subspace remains suitable for subspace augmentation. The only expensive operation here is the modification of the search directions in (3.14). The vector updates will lead to additional operations of the order $(p-j)n$ product and sums to remove the j^{th} direction. On the other hand, the updates of the matrix $\mathbf{T}_p$ will only require a negligible amount of operation. When the total number of directions used is restricted, the worst case scenario arise when the recycled subspace is already full ($p+k$ directions) and when the first k directions must be erased in order to recycle the directions computed in the last cycle of k iterations. This will correspond to $(kp + k^2/2 - k/2)n$ additional products and sums. As it will be shown in the numerical experimentation, this worst case scenario is however unlikely and the additional operations due to the removing of search directions in the partial recycling strategy will remain limited.

3.3 Comparison between recycling methods

All the recycling techniques discussed in this context are expansions of well-established Krylov iterative methods, which have been extensively researched in existing literature. Specifically, the recycling approaches for the GCR and Orthodir methods are explicitly introduced in [53] and [1] respectively. However, the incorporation of these methods with a flexible set of directions was not addressed. In contrast, the augmentation of the GMRES method with a diverse set of directions was directly treated in [54]. Therefore, it can serve as a foundational element for recycling methods within the GMRES subspace framework. While numerous advanced recycling techniques for GMRES have been experimented with, as previously mentioned, the basic recycling of directions without deflation did not was experimented in the literature. The expansion of the recycling technique to the simpler GMRES method is also an original contribution since only the deflated restarted simpler GMRES has previously been covered. Lastly, the selective subspace recycling methods as presented in this work were not already covered in the literature for GMRES-type methods.

Once again, a direct comparison between the methods is possible and the computational requirements are resumed in Table 3.1. The numerical stability of these techniques will persist unchanged, as they rely on the foundation of the classical algorithm. Hence, opting for the GMRES method or the residual based variant of the other approaches is advisable in situations where the problem at hand could potentially lead to instabilities or when a substantial number of directions are employed.

In terms of memory requirements, the GMRES method loses its advantage over the others due to the necessity of storing an additional basis for subspace recycling. In the worst-case

scenario, the SGMRES method may also need to store two separate sets of vectors to represent the recycled subspace. However, these two sets can significantly overlap, as it is the case in the classical algorithm. Hence, the only method offering a storage advantage is the simpler GMRES approach that is not based on the residual. It is important to bear in mind, though, that this specific method carries the potential for numerical instabilities, as elucidated in Section 2.5.

This recycling techniques will avoid an increase in the count of matrix vector products, which offers a notable advantage since matrix vector products are typically the most resource-intensive operation across numerous applications, particularly in the present context. In terms of computational demands, all GMRES-type methods necessitate roughly equivalent amounts of operations. Depending on the utilization of the recycling techniques, either the count of recycled directions p or the number of iterations k will be relatively larger compared to the other. However, in both scenarios, the GCR or Orthodir method will again require approximately fifty percent more operations.

However, supplementary operations are required when the number of recycled directions is restricted and specific retained directions are chosen. In the most adverse situation, this can result in the additional operations outlined in Table 3.2. This additional cost exclusively affects GMRES-type methods, ultimately resulting in the GCR and Orthodir methods attaining a comparable level of efficiency to the GMRES-type techniques. Nevertheless, it's worth noting that this worst-case scenario isn't necessarily commonplace, and in practical applications, the supplementary cost might remains constrained.

In summary, the GMRES method maintains its preferred status due to its theoretically superior stability properties. Nevertheless, it doesn't exhibit lower storage consumption, as was the case with the classical algorithm. As a result, the RB-SGMRES approach might yield comparable outcomes in practical scenarios, while demanding roughly equivalent computational and memory resources. Lastly, when stability is not a concern, the simpler GMRES method could result in reduced storage demands compared to other methods, thanks to the potential sharing of vectors between its two stored bases.

Number of	**GMRES**	**SGMRES**	**RB-SGMRES**	**GCR/Orthodir**
scalars stored	$(2p+2k+1)n$	$(2p+k+2)n$	$(2p+2k+1)n$	$(2p+2k+1)n$
matrix vector products	$k+1$	$k+1$	$k+1$	$k+1$
scalar products	$(2kp+k^2+8k)n$	$(2kp+k^2+5k)n$	$(2kp+k^2+5k)n$	$(3kp+\frac{3}{2}k^2+\frac{9}{2}k)n$
scalar additions	$(2kp+k^2+5k)n$	$(2kp+k^2+4k)n$	$(2kp+k^2+4k)n$	$(3kp+\frac{3}{2}k^2+\frac{5}{2}k)n$

Table 3.1: Approximate memory and computation requirements of one cycle of the restarted Krylov subspace methods for k iterations in one cycle and p recycled directions.

Number of	**GMRES**	**SGMRES**	**RB-SGMRES**	**GCR/Orthodir**
scalar products	$(kp+\frac{1}{2}k^2-\frac{1}{2}k)n$	$(kp+\frac{1}{2}k^2-\frac{1}{2}k)n$	$(kp+\frac{1}{2}k^2-\frac{1}{2}k)n$	0
scalar additions	$(kp+\frac{1}{2}k^2-\frac{1}{2}k)n$	$(kp+\frac{1}{2}k^2-\frac{1}{2}k)n$	$(kp+\frac{1}{2}k^2-\frac{1}{2}k)n$	0

Table 3.2: Additional computation cost resuired to remove k direction from a recycled subspace of size $p+k$ in the worst case senario.

3.4 Selection strategies for partial subspace recycling

When dealing with a substantial number of systems or when the system's size necessitates numerous iterations for convergence, both memory and computational requirements can again become a limiting factor. The complete subspace recycling strategy encounters the same drawbacks as the non-restarted versions of these methods. Consequently, the necessity for partial subspace recycling becomes significant. This partial subspace recycling entails two distinct tasks. The first task involves determining the optimal directions of the subspace to be employed in recycling. Subsequently, the selected directions must be appropriately conditioned to align with the augmentation frameworks delineated in Section 3.1 and 3.2. No further adjustments are necessary for the GCR or Orthodir method, whereas for the GMRES-type approach, the corresponding operations were discussed in Section 3.2.3.

The most straightforward approach to restricting the number of recycled directions involves retaining only the first directions. In addition to the simple direction selection, no further operations are necessary for GMRES-type methods, as the initial directions inherently possess the right properties. This method, despite its simplicity, already yields satisfactory performance. More sophisticated methods could be derived from assessing the impact of each direction on the reduction of the residual norm or from considering the orthogonality properties of the directions. In [55], it was demonstrated that an effective strategy for selecting recycled directions involves retaining those directions that exert the most influence during orthogonalization. In other words, this involves selecting the directions associated with the largest orthogonalization coefficients. Some subspace recycling strategies will be studied in Section 4.1.4 and 4.2.5.

Chapter 4

Numerical experiments

In this chapter, we will delve into the resolution of wave scattering problems through the utilization of the domain decomposition method coupled with subspace recycling Krylov solvers. By considering various applications, we can elucidate the advantages and limitations of employing such methods within a this practical framework.

To numerically address the problems at hand, we will exploit some of the Gmsh related libraries. Consequently, the construction and meshing of models, along with their associated subdomains, will be executed using Gmsh. Subsequently, the GmshDDM library will facilitate the formulation of domain decomposition problems, utilizing the GmshFEM library for generating and solving volume-related problems. This collection of libraries, along with their source code, is readily accessible on the ONELAB repository[1]. Notably, the contribution of this work resides in the implementation of recycling iterative solvers to address the interface problem within the GmshDDM library. A version of this library incorporating these contributions can be found within a dedicated project[2].

The resolution of the small-scale problem can be managed by a laptop. However, for tackling larger scales of resolution, a high performance computing massively parallel cluster will become necessary. In this context, the NIC5 cluster, accessible to members of the University of Liège, will be used. Detailed specifications about the cluster can be found on the "Consortium des Équipements de Calcul Intensif" website[3].

4.1 Homogeneous case

As a starting point, the resolution of the Helmholtz equation can be investigated within the scope of wave scattering in a homogeneous medium. The simulation of such a problem will first be detailed for a single point source excitation. Despite its purely theoretical nature, this particular scenario already allows for the demonstration of some fundamental properties of the domain decomposition method and its resolution with iterative methods. Following this, the analysis will be expanded to incorporate multiple problems with a point source excitation, thereby revealing the potential of recycling schemes to accelerate multiple resolutions. Through this examination, valuable insights will be gained concerning the properties of recycling meth-

[1] `https://gitlab.onelab.info/gmsh`

[2] `https://gitlab.onelab.info/timgabriel/ddm-master-thesis-tim-gabriel`

[3] `https://www.ceci-hpc.be/clusters.html`

ods and to apply them in practical applications. Only after, a more concrete example will be explored

4.1.1 Analytical solution

In the context of a homogeneous infinite domain, it is possible to analytically compute the solution for a point source excitation. Leveraging the inherent symmetry of the problem, one can employ the polar coordinate system to deduce the analytical solution. When the source of excitation is positioned at the origin, the Helmholtz equation takes on the following form:

$$\Delta u(r,\theta) + k^2 u(r,\theta) = 0, \tag{4.1}$$

where r represents the distance from the origin, and θ indicates the angle formed with respect to the x axis. It is important to note that the source point under consideration can be visualized as an excitation originating from a line perpendicular to the plane defined by r and θ. As a result, the solution would remain unchanged along this third dimension. The intrinsic symmetry of the solution suggests that it can be structured as function of separable variables: $u(r,\theta) = R(r)\Theta(\theta)$, with R representing a function associated with the variable r, and Θ is associated with the variable θ. Subsequently, the Helmholtz equation (4.1) can be expressed using the polar representation of the Laplacian:

$$R''(r)\Theta(\theta) + \frac{1}{r}R'(r)\Theta(\theta) + \frac{1}{r^2}R(r)\Theta''(\theta) + k^2 R(r)\Theta(\theta) = 0 \tag{4.2}$$

The individual contributions of each function can be isolated such that

$$r^2\frac{R''(r)}{R(r)} + r\frac{R'(r)}{R(r)} + r^2k^2 = -\frac{\Theta''(\theta)}{\Theta(\theta)} = C^*. \tag{4.3}$$

Here, C^* represents a constant value. The anisotropic nature of the problem will enforce the function $\Theta(\theta)$ to adopt a constant value, therefore the constant C^* must inevitably equal zero. As a result, through the application of the variable substitution $\rho = kr$, the radial element $R(r)$ is expressed as the differential equation:

$$\rho^2 R_*''(\rho) + \rho R_*'(\rho) + \rho^2 R_*(\rho) = 0. \tag{4.4}$$

The solution to this equations are the well-known Bessel functions where the associate integer is zero. Overall, the solution of the Helmholtz equation is given by

$$u(r,\theta) = \frac{1}{4}\big[-Y_0(kr) + iJ_0(kr)\big], \tag{4.5}$$

where $J_0(r)$ and $Y_0(r)$ are the Bessel functions of respectively the first and the second kind [56].

4.1.2 Numerical solution

The homogeneous problem at hand can be effectively solved using a numerical approach, as outlined in Section 1.3. Within this framework, the propagation of a wave with a frequency of $f = 1\,\mathrm{Hz}$ will be investigated in a medium where the propagation velocity remains uniformly constant, with $c(\boldsymbol{x}) = 1\,\mathrm{m/s}$. The computational domain under consideration is a square region, featuring a side length of $2.5\,\mathrm{m}$. This domain is partitioned into nine subdomains, all of equal dimension. Additionally, a point of excitation is placed at the central location of this domain with a magnitude of 1. For the discretization, elements of size $l_c = 0.02\,\mathrm{m}$ and of the

first order are employed. The desired accuracy for the resolution of the interface problem is set at $tol = 10^{-6}$, and the GMRES method will be used. It is important to note, however, that alternative solvers could also be considered without changing the outcome.

The results of such a numerical simulation are illustrated in Figure 4.1, and a comparison is made with the exact solution of a wave propagating in an unbounded homogeneous space. The observed disparities in the solutions predominantly arise from the influence of boundaries and the application of imperfect absorbing boundary conditions. Indeed, however how much the mesh is refined, some inevitable disparities will remain between the numerical and the analytical solution. Consequently, the comparison to the analytical solution is not an optimal benchmark to assess the accuracy of the numerical solution. Therefore, in order to facilitate a relevant comparison between the solutions, an excessively refined solution will be computed and used as a reference to compare solutions.

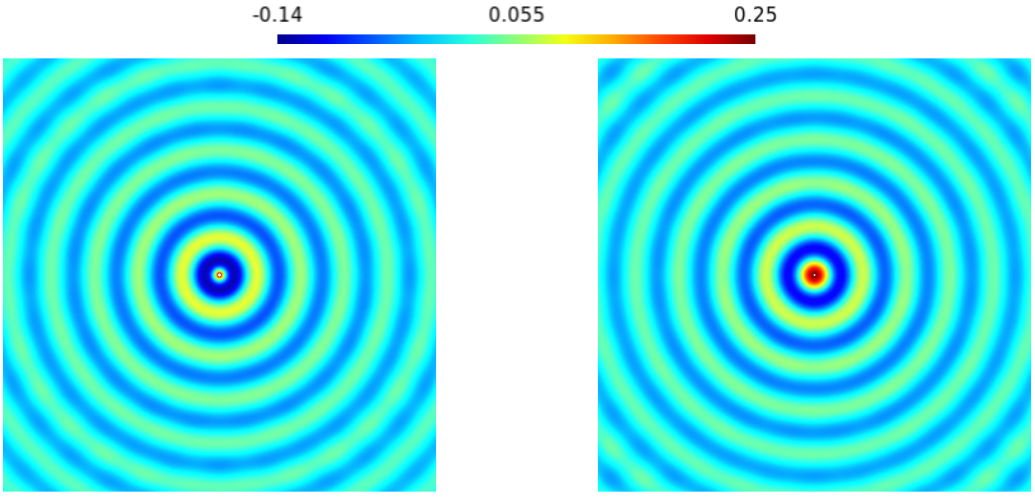
To obtain reliable solutions using numerical simulation, a study of the different parameters of the simulation must be carried out. First, the mesh will be progressively refined to minimize errors within the solution. The finite elements will here always be considered of the first order. After, the optimal tolerance on the iterative method will be found to limit le number of iterations required to reach this tolerance while keeping an accurate solution.

Mesh refinement

To investigate the impact of mesh refinement, the problem will be addressed without employing the domain decomposition method. For the purpose of establishing a basis for comparison, the discrepancies between the approximate solution fields and an over-refined solution ($lc = 0.002\,\mathrm{m}$ and first order) will be computed, as it was mentioned. It is important to note that the problem solution exhibits an asymptote in the real field at the excitation point. This characteristic can lead to significant variations in the solution when different meshes are employed. Consequently, only the imaginary field will be used to assess the accuracy of the solution. Here, the root mean square error will be used in order to aggregate the errors over the whole field into a scalar value. One drawback of the RMSE metric is that it is disproportionally influenced by outliers. This emphasizes the fact that the real field should not be considered for the estimation of the error on the solution. The error associated to the imaginary part of the solution $u^i(\boldsymbol{x})$ can then be defined from the most accurate solution available $u^i_*(\boldsymbol{x})$ such that

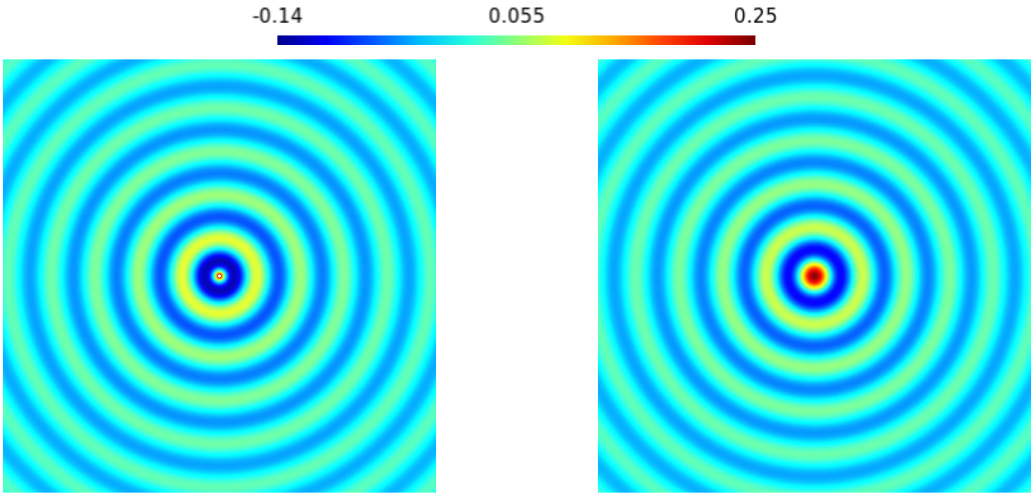
$$\mathrm{RMSE} = \sqrt{\frac{1}{\Omega}\int_\Omega \left(\frac{u^i_*(\boldsymbol{x}) - u^i(\boldsymbol{x})}{u^i_*(\boldsymbol{x})}\right)^2 d\Omega}. \tag{4.6}$$

The results associated with this analysis are depicted in Figure 4.2. It can be seen that at least a mesh size of $l_c = 0.08\,\mathrm{m}$ should be considered in order to obtain a not too large relative error in the solution, roughly 1.56% for elements of the first order. This mesh size correspond to having $\lambda/l_c \approx 12$ points along a wavelength. However, in practical application, a more refined mesh should be considered. Indeed, in [57], a study on the numerical simulation was conducted for the Helmholtz equation in one dimension with a different error metric more suited for the analysis. The main conclusion of the study is that the estimate of the error contains two contributions that varies with the nondimensional wavenumber defined as kL where L is the size of the domain. Then, for simulation with relatively small adimensional wavenumber ($kL < 100$) a constant number of points per wavelength can yield solutions with a sufficient resolution if the number of point is at least of 20. This is due to the fact that the leading term of the error will increase in that situation as the wave number multiplied by the



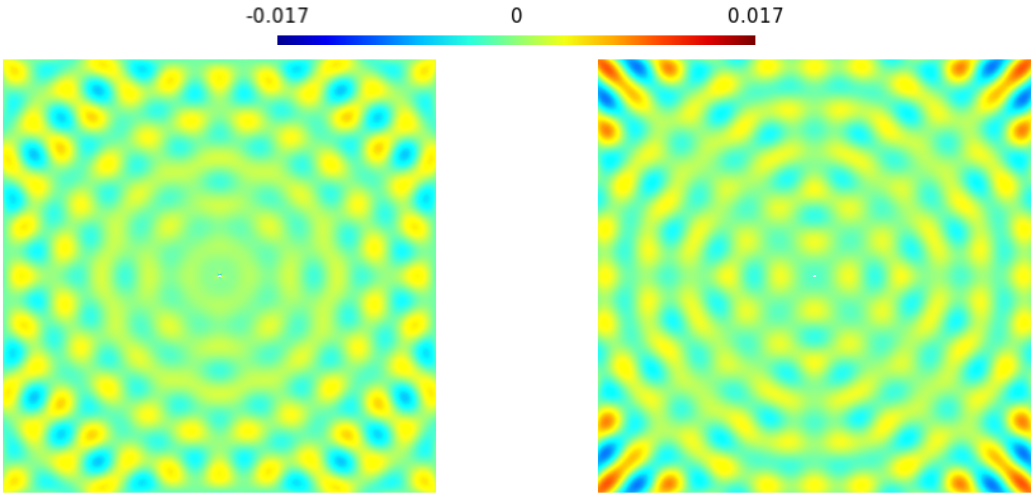
(a) Real part of the approximate solution.

(b) Imaginary part of the approximate solution.



(c) Real part of the exact solution.

(d) Imaginary part of the exact solution.



(e) Real part of the error.

(f) Imaginary part of the error.

Figure 4.1: Exact and approximate solution of a point source scattering wave in an homogeneous medium and the associated relative error.

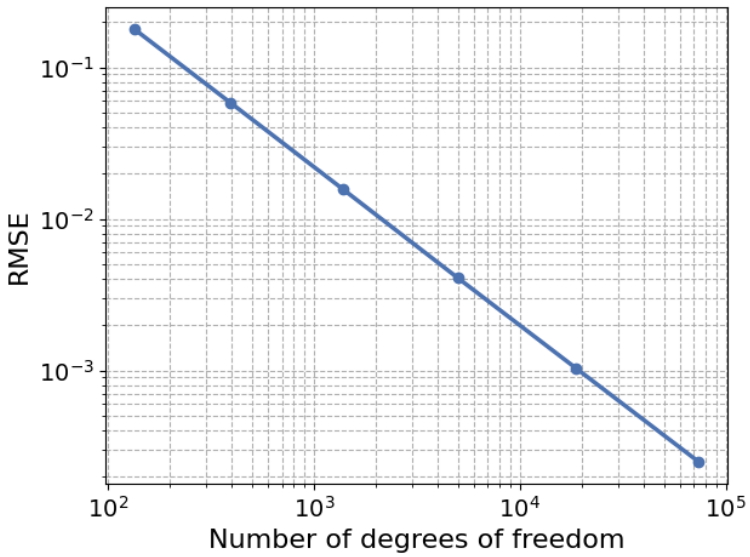
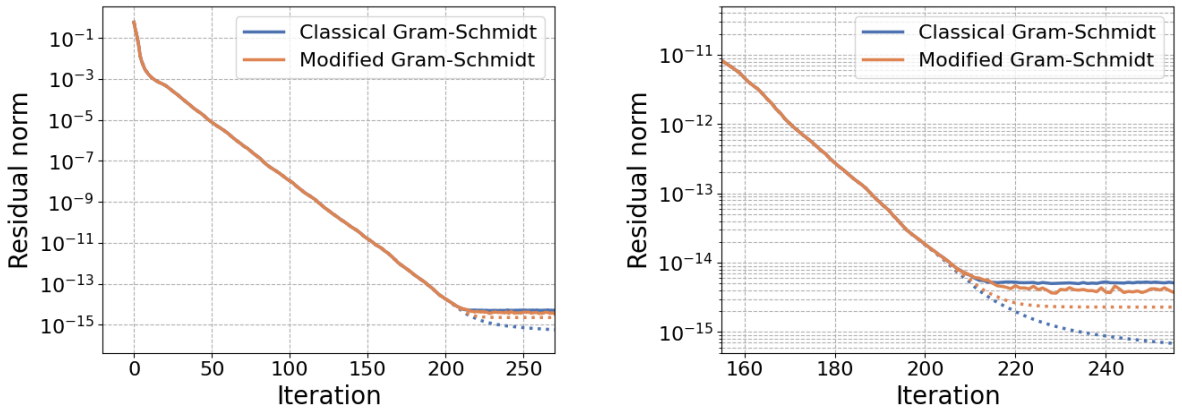


Figure 4.2: Root mean square error on the solution for element sizes l_c of 0.32, 0.16, 0.08, 0.04, 0.02 and 0.01 m compared to an overly refined simulation ($l_c = 0.002\,\text{m}$).

mesh size: error$\approx kl_c$. For large adimensional wavenumber values ($kL > 100$) however, this will not be sufficient since the leading term will scale as $l_c^2 k^3$. Thus the required mesh to obtain accurate solutions will need to be finer for large wavenumber. Following that and since one small wavenumbers will be considered, we will continue with a mesh size of $l_c = 0.04\,\text{m}$ since it will correspond to 25 nodes representing one wavelength. Nonetheless, it is worth highlighting that the choice of desired solution accuracy should be determined based on the specifics of the particular application being studied.

Resolution of the interface problem

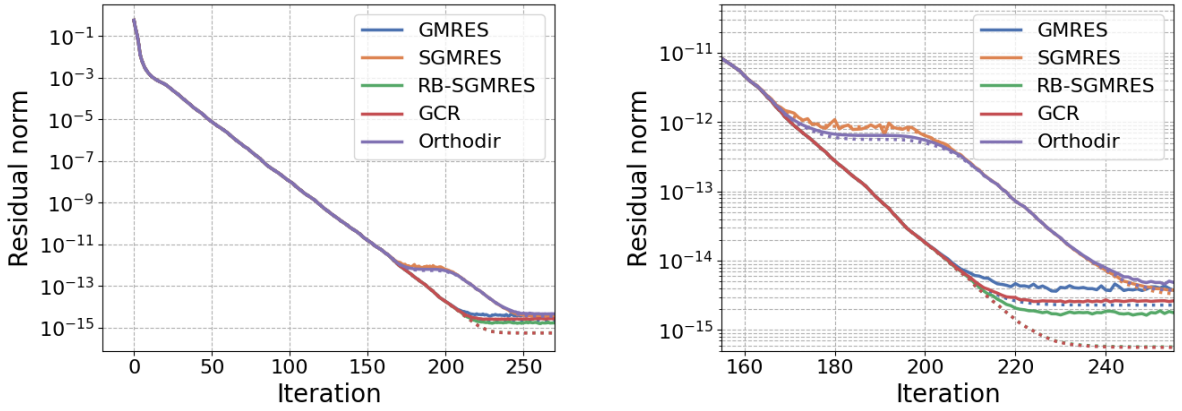
During the resolution of the interface problem, the results of the various studied methods would match in exact arithmetic. However, since computations are carried out in finite precision arithmetic, disparities will arise in the outcomes. In Figure 4.3, the relative residual norms are depicted as a function of the iterations in the resolution using the studied solvers without restarting. This figure reveals that once a sufficient number of iterations has been completed, the method ceases to exhibit further improvement. This phenomenon arises due to the computational precision limit, which is slightly less than 10^{-15} for double-precision floating-point variables. Additionally, this figure distinctly showcases the inherent numerical instabilities in the SGMRES and Orthodir methods. At some required tolerance, these methods demand more iterations to attain a comparable decrease of the residual norm. They are able to reaching residual norms as small as those attainable with other methods with more iterations in this particular example, but this is not always the case in general as demonstrated in [32]. Nevertheless, these differences only manifest when the residual norm is exceedingly tiny. This suggests that these methods could still be suitable when the tolerance on the resolution is not set too low, and therefore when the number of directions used remains moderate.



(a) For all the iterations.

(b) For the last iterations.

Figure 4.4: Evolution of the residual norm for the classical and modified Gram-Schmidt orthogonalization with the GMRES method. Solid lines represent the true residual norm while dotted lines represent is the residual norm approximated by the method.



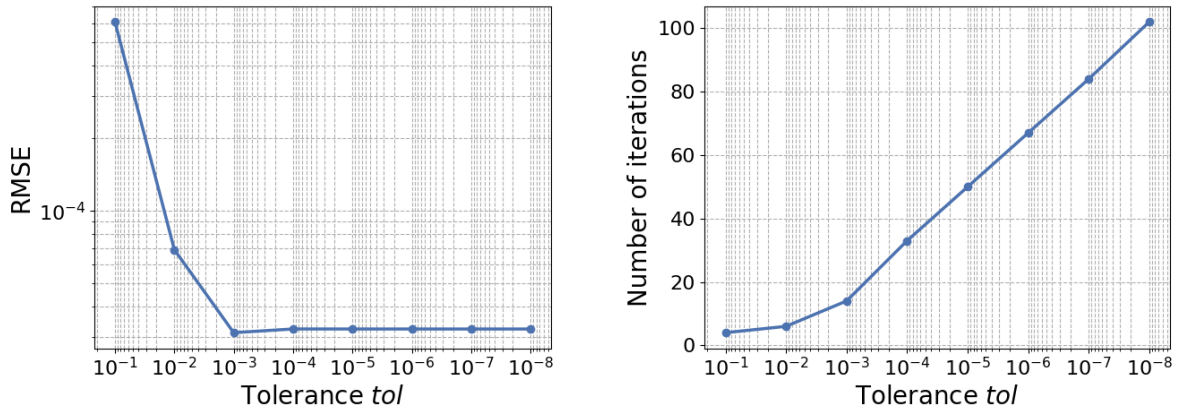
(a) For all the iterations.

(b) Zoom on the last iterations.

Figure 4.3: Evolution of the residual norm for different Krylov solvers without restarting. Solid lines represent the true residual norm while dotted lines represent is the residual norm approximated by the method.

The use of modified Gram-Schmidt orthogonalization can significantly impact the stability of the methods. Figure 4.4 illustrates the relative residual decrease for the GMRES method using the classical and modified processes. No major differences can be seen between the two methods for a single resolution of a problem of small size as considered here. This would justify the use of the classical Gram-Schmidt orthogonalization. However, as the problem becomes larger and as the number of iteration increases then differences may occur and since the classical process do provide only limited advantage, the modified process is preferred most of the time in practical implementations.

The resolution of the interface problem using an iterative method necessitates defining a tolerance that will serve as the stopping criterion. To compare the solutions, the error with the overly refined solution will be used once again. As it can be seen in Figure 4.5a, the error on the solution quickly decreases when the tolerance on the interface resolution gets smaller.



(a) Error on the whole solution.

(b) Number of iteration for the interface problem to converge.

Figure 4.5: Influence of the tolerance value of the interface problem on the accuracy of the solution and the number of iterations associated.

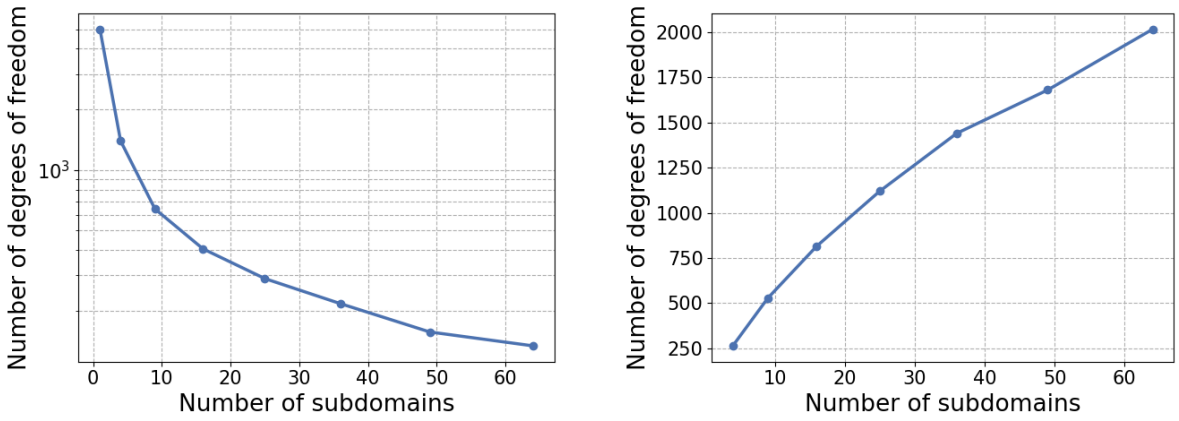
Then, it reaches a lower value and it does not decrease anymore. This minimum value actually corresponds to an error of the same order as the error due to the mesh and therefore it can not be decreased by decreasing the tolerance of the iterative method. The resolution of the interface problem will then be considered as having converged. This can then explain why the residual norm shown in Figure 4.3a decreases a lot for the few first iterations and then decreases smoothly. The first part of the curve correspond to large modification of the solution while the second does not affect the accuracy of the solution substantially.

The choice of the tolerance also affects directly the number of iterations required to reach this specified tolerance as it is depicted in Figure 4.5b. Hence it has a huge consequence on the overall memory and computational consumption of the iterative method. In the following, systems with a larger amount of degrees of freedom will be considered. Additionally, the domain will be decomposed in a larger number of subdomain, increasing even more the size of the interface problem. Therefore a rather conservative tolerance will be chosen. A value of 10^{-6} appears to be a favorable balance between ensuring the convergence of the error on the solution due to the domain decomposition, and the associated computational demands. Here again, the particular application considered and the requirements on the solution should enter into account when choosing the value of this tolerance.

Domain decomposition

The numerical solution for this wave scattering problem can be achieved using the domain decomposition method rather then classical finite element method but it comes with its advantages and disadvantages. While various domain decomposition strategies are possible, the focus here will be to share the computational workload within a distributed computing system. As a result, a straightforward domain decomposition approach will be adopted. In this method, the domain will be divided into subdomains with the same aspect ratio as the whole domain, ensuring uniform sizes across all subdomains. This implies the use of a square number of subdomains. Additionally, the chosen mesh consists of first order elements with a size of $l_c = 0.04\,\mathrm{m}$ and the tolerance of the interface resolution will be set to $tol = 10^{-6}$ as mentioned.

In Figures 4.6, the evolution of the number of degrees of freedom for both the volumic problems and the interface problem is depicted as the number of subdomains increases. As evident, the number of degrees of freedom for the resolution of each individual subdomain decreases significantly, since it corresponds to distribute the total number of degrees of freedom between all subdomains. Conversely, the interface problem experiences an increase in the number of degrees of freedom as the interface size gets larger. Nonetheless, this increase remains notably limited in comparison to the reduction in degrees of freedom observed for the subdomain problems. This observation shows the capacity of the domain decomposition method to segment a volumic problem into smaller subproblems, albeit at the expense of resolving an interface problem. This ability is particularly promising if it translates into reduced memory and computational consumption for the overall solution.



(a) For one volumic problem.

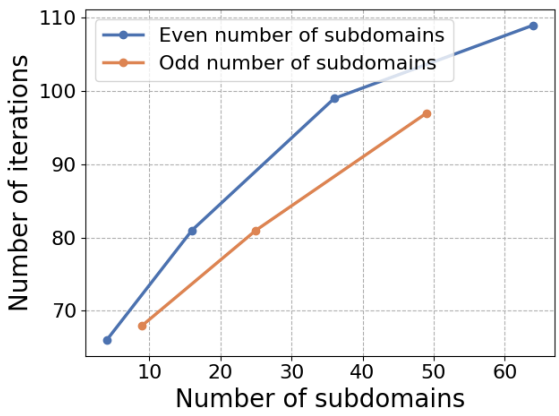
(b) For the interface problem.

Figure 4.6: Number of degrees of freedom for different numbers of subdomains.

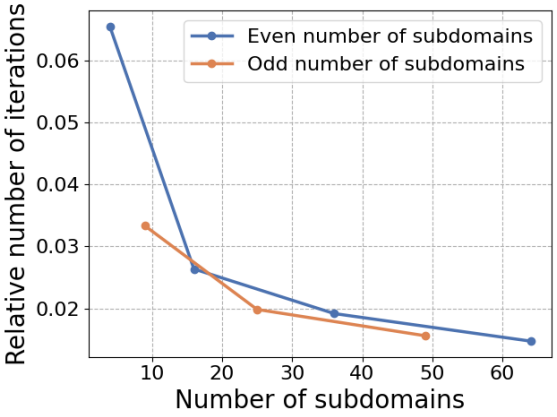
The limitations of the method will then be the increased storage and computation demands required to resolve the interface problem. Two significant factors come into play when evaluating the computation consumption of the interface problem: the size of the problem and the number of iterations. The actual size of the interface problem increases with the number of subdomains as represented in Figure 4.6b. Moreover, the increased size of the interface problem can lead to more iterations to achieve convergence of the resolution. This trend is depicted in Figure 4.21, although the increase in the number of iterations is not as significant as the growth in the number of degrees of freedom. This point is further emphasized by Figure 4.7b, which illustrates the ratio of the number of iterations required for convergence to the number of degrees of freedom in the corresponding problem. This suggests that the resolution of such interface problems for a large number of degrees of freedom might exhibit a favorable scalability.

Observing the data, it becomes apparent that the number of iterations required differs significantly based on whether the number of subdomains is even or odd. This variance could be attributed to the placement of the point source excitation. In cases where the number of subdomains is even, the point source is positioned at the intersection of four domains. Conversely, when the number of subdomains is odd, the point source resides at the center of a domain. Then, when a point is situated on an interface, it might require more iterations for the iteration method to converge. It is worth noting that when a source lies at the interface of domains, its value is distributed among the subdomains. For example, in situations where

the number of subdomains is even, one-fourth of the excitation is allocated to each involved subdomain. An alternative approach could have been to introduce the excitation in just one subdomain. While this would have ultimately resulted in the same solution, it would have necessitated more iterations for the method to converge. This is because of the uneven initialization caused by the excitation being concentrated in a single subdomain.

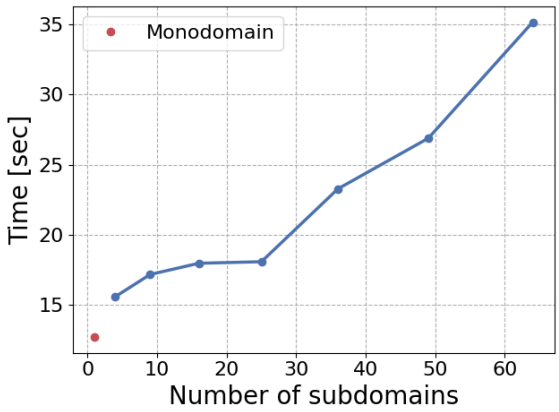


(a) Number of iterations to converge.

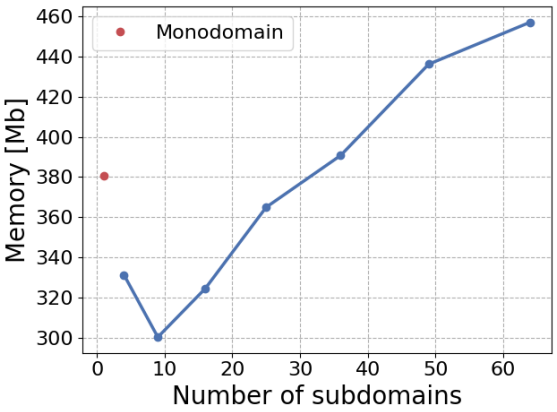


(b) Number of iterations to converge divided by the number of degrees of freedom.

Figure 4.7: Number of iteration required to converge for different numbers of subdomains.



(a) Computational time.



(b) Memory consumption (Maximum resident set size).

Figure 4.8: Consumption of the resolutions for different numbers of subdomains.

Given the relatively modest size of the problem under consideration, the domain decomposition method may not prove to be significantly advantageous in terms of memory usage limitations. This is demonstrated by the storage requirements illustrated in Figure 4.8b. Some gains can be achieved with four, nine or twenty five subdomains, yielding improvements of up to 21% on the storage consumption. However, since the size of the interface problem is not small in comparison to the volumic problem, its memory consumption quickly adds up thus leading to a memory demand comparable or larger to the monodomain resolution. Furthermore, in relation to computational resources, the scale of the current problem is insufficient to result in inefficient resolution using a direct solver for the monodomain. This is particularly

true given the two-dimensional nature of the problem, where direct solvers are very efficient. Therefore, the domain decomposition method will most of the time be less efficient in term of computational time in the simulations considered here. In the following, the number of subdomains used will be set to 9 for the simple homoneous problem.

4.1.3 Resolutions of multiple source points

The simple propagation of waves in a uniform field remains the same regardless of the location of the source point. Consequently, when disregarding the boundaries, if the excitation source changes position, the solution to the corresponding problem is simply the translated solution at the new location of the source point. Nevertheless, sequentially solving problems with varying source point positions will offer theoretical insights into evaluating the efficiency of recycling techniques for iterative solvers in the context of the domain decomposition method. This arises due to the fact that the directions employed in solving an interface problem can be reused in subsequent resolutions. Hence, it can also be anticipated that these directions will be advantageous when the solutions are not drastically different.

In this context, we will consider 32 distinct excitation sources, each separated by a quarter of the wavelength. The variations among these problems are solely attributed to differences in the source term, leading to variations in the right-hand side only. Moreover, the source points considered here will lie along the horizontal axis and will be treated in a left-to-right order. A 10,m square domain will be used and divided into nine smaller square subdomains. Ultimately, a similar mesh ($l_C = 0.04\,\text{m}$ and first order) and resolution tolerance ($tol = 10^{-6}$) will be used.

Figure 4.9 presents the relative residual norms for these thirty-two resolutions as a function of the total number of matrix vector products used. Notably, the recycling of directions already reduces the residual norm even before any iterations of the method, and prior to any matrix vector products. Additionally, the trend indicates that the rate of convergence intensifies with an increased number of recycled directions.

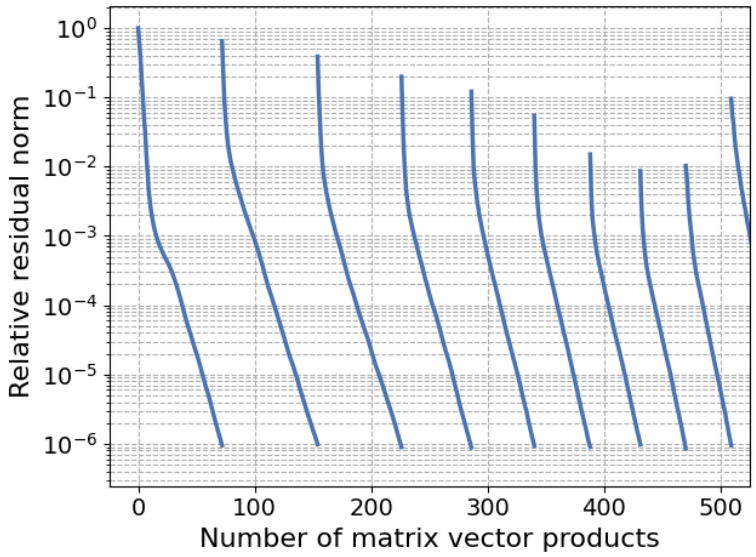


Figure 4.9: Relative residual norms as a function of the number of matrix vector product for multiple scattering problems.

One drawback of recycling, however, is that the initial resolutions using recycling tend to be less efficient than resolutions without recycling. The results presented here were computed using the GMRES method, yet using any other solver would have led to similar outcomes. The decrease in convergence efficiency during the initial resolutions must be attributed to the exploration of the search space by the method. When recycling a search space, the inherent subspace explored by the method will differ, as it begins with the residual vector projected out of the recycled space. Additionally, the search directions will be orthogonalized with respect to the augmentation space, leading to the exploration of even a different subspace than the Krylov subspace corresponding to the projected residual. Using this subsequent subspace might be less efficient than using the traditional one for solving this specific linear problem. Nevertheless, once a substantial number of directions are recycled, the resolution using recycling methods will require fewer iterations, leading to reduced computational requirements, particularly in terms of the overall number of matrix vector products. This can be clearly seen in Figure 4.10, which displays the number of iterations needed for convergence across different considered source points. Consequently, the total number of iterations in the entire resolution decreases from 2365 to 1067, representing a nearly 55% reduction in this simple scenario.

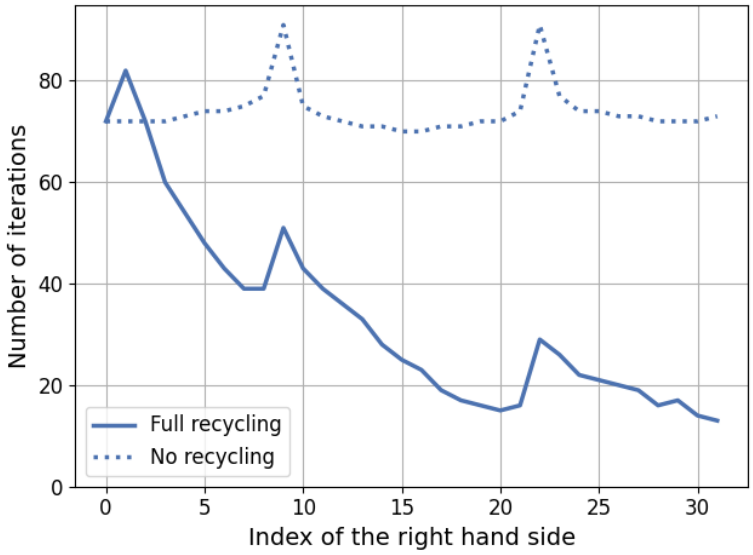


Figure 4.10: Number of iterations required to converge for different right hande sides when using the full subspace recycling or no recycling at all.

It is worth noting that the number of iterations required to solve a problem without restarting is not consistently the same. This variability is particularly evident in the example depicted in Figure 4.10, where two resolutions necessitate a larger number of iterations. This discrepancy is attributed to the proximity of the corresponding source points of these problems to a subdomain interface. In Figure 4.11, the solution for the ninth right-hand side is displayed. This reveals that the solution of the interface problem will exhibit more extreme values near this point. This then contributed to the increase in the number of iterations required for the convergence of the iterative method. The additional iterations are also noticeable in the iteration count when subspace recycling is employed. Moreover, this increase in the number of iterations influences the preceding and subsequent resolutions centered around these points. This phenomenon can be elucidated by considering the sequential exploration of problems from left to right. In proximity to the interfaces, the chosen excitation sources are positioned in

distinct subdomains. Consequently, their respective solutions will exhibit considerable variations on the interfaces. This discrepancy is especially pronounced for interfaces nearest to the excitation source. Such significant differences in solutions contribute to the less pronounced reduction in the iteration count in those cases. Collectively, these factors clarify the observed pattern in Figure 4.10 and provide insights into the reduced improvements provided by the recycling method when the considered excitation point are situated near another subdomain.

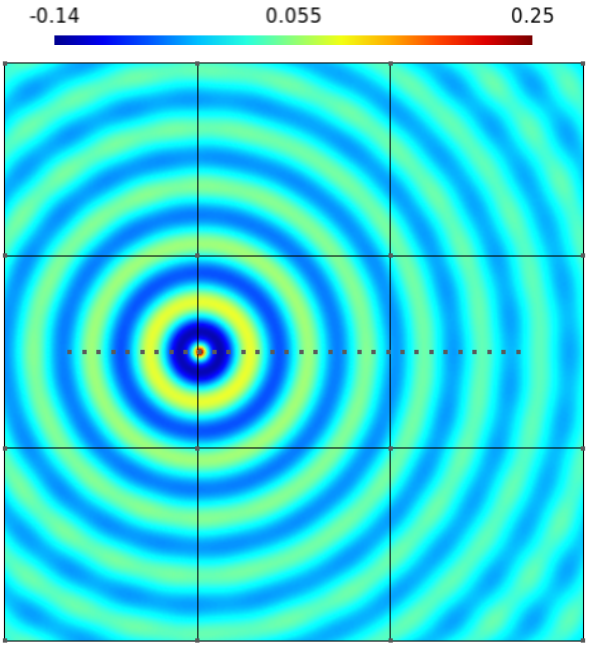


Figure 4.11: Solution for the ninth excitation point where all 32 excitation points are represented.

4.1.4 Influence of parameters on the recycling

Previously, the significant enhancements that recycling strategies can bring to the resolution of multiple scattering problems was demonstrated, particularly in a homogeneous field. Now, let us investigate how this method performs under different parameter selections for the problem. The frequency of the wave stands as a critical parameter in the study of wave propagation. However, due to our focus on a theoretical homogeneous medium with a predetermined wave propagation speed, conclusions regarding a specific frequency can be extrapolated by straightforwardly scaling the other parameters. As a result, in the subsequent analysis, we will maintain a consistent frequency for the wave of 1 Hz, ensuring the mesh remains unchanged throughout. All other relationships will be formulated with respect to this constant frequency.

Effect of the meshing

To initiate the exploration, one can begin by examining the influence of mesh size on the performance of the recycling strategy. While maintaining a separation of 0.25λ between 32 source points where $\lambda = 1\,\mathrm{m}$ is here the wavelength, the mesh size can be adjusted to gauge the impact on the recycling strategy. Of course, the mesh size must remain sufficiently small to ensure an accurate representation of the solution without introducing excessive errors. Therefore, we will consider a maximum mesh size of 0.08 m, although this might not be sufficient for practical applications. The outcomes of this investigation are illustrated in Figure 4.12.

As the grid becomes finer, more elements are introduced along the interfaces, causing the size of the interface to expand with the total problem size. Consequently, resolving a single problem will demand more iterations. Interestingly, this increase in iteration count does not adversely affect the benefits offered by the recycling strategy. The relative reduction in the iteration count remains relatively consistent as the mesh size is decreased. There is a noteworthy exception related to the coarser mesh. However, it is important to emphasized that a coarser mesh might fail to accurately represent the solution. Hence, any excessive reduction in the iteration count might be attributed to this lack of accurate representation. Furthermore, the finer mesh exhibits a more pronounced reduction in the iteration count compared to other meshes. However, this effect is mitigated by the increase in the number of iterations at the initial stages and near the interfaces.

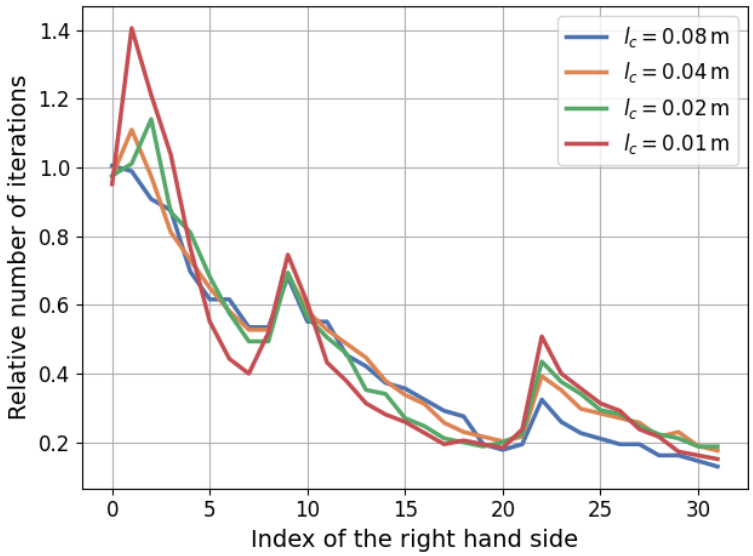


Figure 4.12: Relative number of iterations required to converge for different mesh sizes and for different right hande sides.

Spacing of source points

Another interesting variation would be to adjust the spacing between the different source points. This is motivated by the understanding that when excitation points are in close proximity, their corresponding solutions tend to be closely related. This suggests that sharing search directions between neighboring solutions could yield greater benefits. Additionally, given the periodic nature of the Helmholtz equation, there might be advantages in spacing the sources by multiples of half the wavelength. To explore different spacings, a smaller number of source points will be considered as this will allow for the investigation of larger spacings. However, the finite size of the domain and the limited size of elements impose constraints on the range of spacing available within the two extremes. The smaller spacing cannot be close to the mesh size ($l_c = 0.04$), and at least 8 points will be required to observe the effects of the recycling strategy. In the end, this results in an interval between 0.125 and 1 times the wavelength. The outcomes of these spacing variations are depicted in Figure 4.13. It is important to note that the domain decomposition remains the same and the points are positioned at the center of the domain to minimize boundary condition interference. Consequently, depending on the scenario, some points might be closer or farther from an interface, leading to potential increases in the iteration count at those points. This explains why the first and last points corresponding

to the 0.5λ spacing are higher than anticipated.

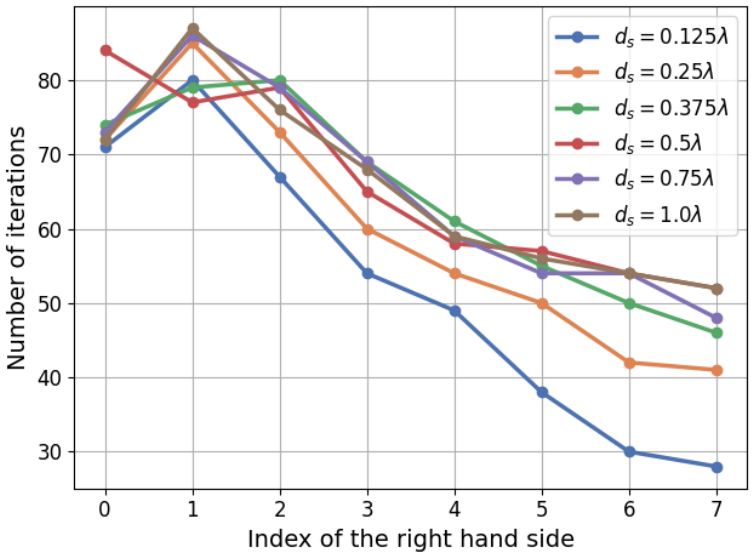


Figure 4.13: Number of iterations required to converge different spacing between the sources and for different right hande sides.

From this analysis, one can infer that the spacing between source points has the potential to enhance the recycling scheme, particularly when the spacing is comparably small relative to the wavelength. However, when the spacing aligns with the wavelength, no significant discrepancy in the effectiveness of the recycling method is observable. Unfortunately, spacings larger than the wavelength are not examined in this study due to the constraints imposed by the domain size. Nevertheless, slightly larger spacings will be investigated in Section 4.2.3, alongside a larger scale problem. It is important to consider that these findings were derived within the context of a particular domain decomposed into nine subdomains. Thus, an exploration of whether domain decomposition impacts the recycling method could also be of interest.

Relative size of the interface problem

When the number of subdomains is augmented, the number of degrees of freedom of the associated problem will correspondingly increase. Consequently, the relative reduction in the iteration count enabled by the recycling strategy for different domain decomposition will be examined here. Figure 4.14 illustrates the number of iterations required to achieve convergence with recycling for a given number of subdomains. This value is divided by the mean number of iterations necessary to solve the problems without recycling. To enhance clarity, this approach is chosen instead of directly dividing the recycling iterations count by the number of iterations in the non-recycling resolution of the same system. By following this method, the impact of recycling on interface points close to boundaries remains distinguishable, unlike the alternative scenario.

This approach allows us to discern that smaller interface problems modestly benefit from a better recycling performance. This could be attributed to the reduced number of degrees of freedom, causing recycled directions to be more likely to align favorably with subsequent directions due to the limited diversity of directions. However, more significantly, the observation reveals that as the number of domains increases, the enhancement attributable to the recycling

strategy appears to remain consistent and does not degrade as the size of the interface problem expands. If this pattern remains verified for larger problems, it will induce a good scalability of the recycling method.

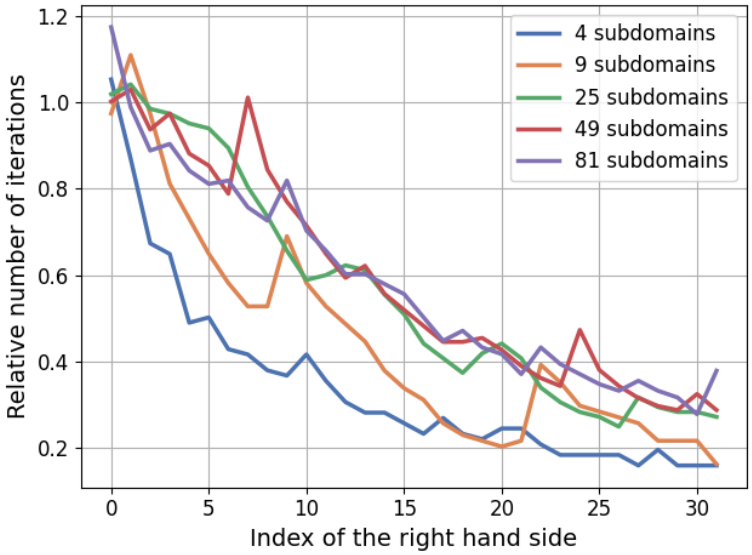


Figure 4.14: Relative number of iterations required to converge for different numbers of subdomains and for different right hande sides.

Recycled subspace selection strategies

As discussed in Section 3.4, various strategies for selecting the optimal directions to retain for recycling are feasible when the quantity of recycled directions is limited. In this context, we will explore four strategies, two of which are straightforward: retaining the first or the last direction. However, two additional strategies will also be examined. The first strategy involves retaining directions that contributed to the most substantial relative reduction of the residual norm upon their inclusion in the search space. This necessitates storing a measure of the relative residual decrease for all directions, which remains constant throughout all the resolutions. The second method entails selecting directions that have an important role in the orthogonalization process. This corresponds to retaining directions linked to the most significant coefficients in absolute value during the orthogonalization.

To study these selection strategies, the same set of problems with 32 excitation source points will be resolved, but with a constraint of storing a maximum of 800 recycled directions. The results for the various selection strategies are illustrated in Figure 4.15. It is evident from the figure that selecting the last direction yields the poorest performance. Conversely, the simple approach of retaining the first directions proves highly effective, delivering superior performance despite its uncomplicated principle. However, the two latter strategies do not exhibit an advantage over the other and particularly not over the "first direction" strategy.

4.1.5 Resolutions of a grid of source points

Instead of exclusively considering sources aligned along a single axis, the examination presented here above can be extended to the resolutions wave scattering for source points distributed across the entire two-dimensional plane. While this scenario may not directly

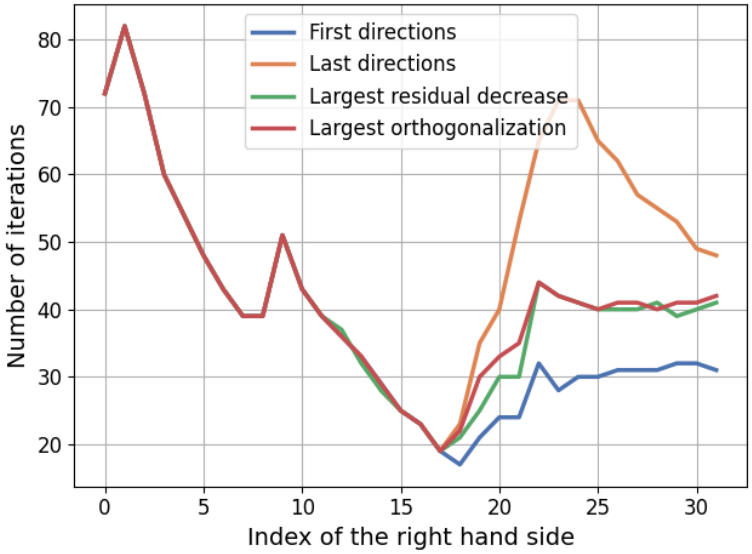


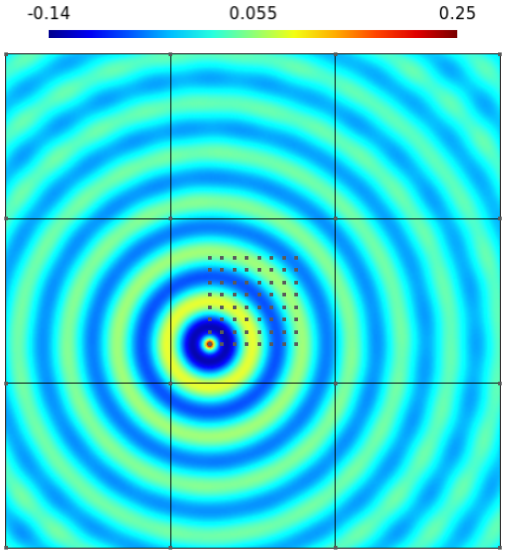
Figure 4.15: Number of iterations required to converge for different subspace recycling selection and for different right hande sides.

correspond to the upcoming practical application, it can provide valuable insights for related problems. For instance, this extension anticipates three-dimensional scenarios where grids of emitters and receivers are employed for full waveform inversion to image subsurface structures. By employing a grid of source points instead of a linear axis, a potential can be anticipated to be able to represent more intricate solutions through the combination of recycled directions. Consequently, a higher level of improvement in iteration count could be expected, as more information is available through solving problems with source terms varying across all spatial dimensions. In this analysis, we will consider an eight by eight grid of excitation points, each separated by the same distance as in previous cases (0.25λ). These points will be resolved in sequence, traversing from left to right and then from the bottom to the top. The setup is illustrated in Figure 4.16a, which shows the solution for the first points.

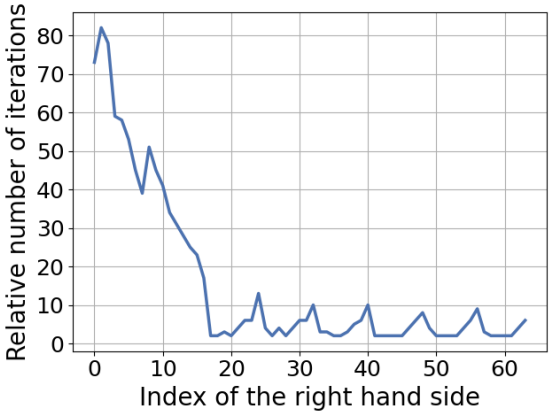
An interesting behavior is observed in the iteration counts, as shown in Figure 4.16b. The reduction in the iteration count is almost the same as the one observed for the case of 8 points separated by the same 0.25λ distance, as depicted in Figure 4.13. The only difference in the solutions is indeed a vertical shift due to the change in the spatial arrangement of the source points. However, when transitioning to a new row, an initial increase in iteration count is evident, as the new point is located at a vertical position that has not already be solved. Nevertheless, the overall iteration count continues to decrease rapidly in subsequent resolutions. This decrease is even more significant than what was experienced with point sources aligned along a single axis, as demonstrated in Figure 4.10.

In less than twenty iterations, the method exhibits a considerable reduction in iteration count, with only one or two iterations required for certain right-hand sides. This is in clear contrast to the approximately 70 iterations needed without recycling. This scenario exploiting an additional dimension, showcases the potential of combining the domain decomposition method with the subspace recycling technique. Consequently, this combination holds great promise for three-dimensional full waveform inversion.

Despite increasing the distance between sources, the benefits of recycling remain substan-



(a) Solution for the first excitation where all 64 excitation points are represented.



(b) Number of iteration per right hand side for the 64 different excitation points are represented.

Figure 4.16: Solution and iteration count for a grid of 8 by 8 source points where the sources are solved from left to right and from the bottom to the top.

tial, as demonstrated in Figure 4.17. Also, when the spacing becomes largely smaller than the size of the wavelength, some additional improvement are obtained, as elaborated in Section 4.1.4. Additionally, one can observe that the increments resulting from changing rows of source points are less pronounced for larger spacing. To rationalize this phenomenon, one could argue that in the case of a spacing of $d_s = 1\lambda$, the method accumulates more directions. This accumulation allows then for better approximation of solutions, even if the next considered excitation source correspond to a different row.

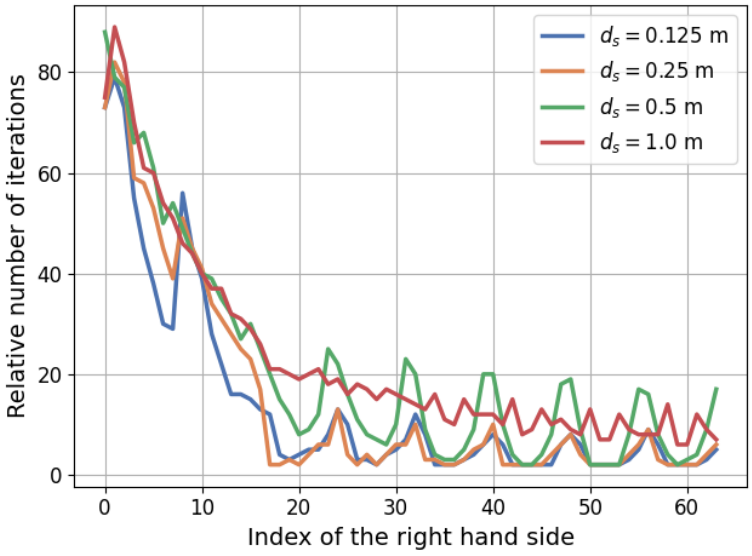


Figure 4.17: Number of iteration per right hand side for a grid of 8 by 8 source points and for different spacings of the sources.

4.2 Marmousi case

The Marmousi model is a synthetic subsurface model used in seismic imaging and exploration geophysics. It is often employed as a benchmark in the geophysical community for testing and evaluating seismic imaging algorithms and inversion techniques. The model was created to mimic the geological features and complexities of a real subsurface environment. The Marmousi model is particularly notable for its intricate and challenging subsurface structures that can pose difficulties for seismic imaging and interpretation. The synthetic nature of the Marmousi model allows researchers and geophysicists to compare the performance of different seismic processing and imaging algorithms in a controlled and well-understood setting. It has been an important tool for the development and validation of seismic imaging and inversion techniques.

Therefore, following the approach outlined in [2], we will use this model as a more practical context involving wave scattering and full waveform inversion. The Marmousi model serves as the practical scenario, with its velocity field portrayed in Figure 4.18. To simulate a more realistic environment, a water layer is included at the top of the model. The complete model spans 9192 m in width and 3116 m in height. In the work of [2], the acoustic wave emitters and receivers occupy identical positions. This strategic alignment simplifies the inversion process, as the adjoint problems can be derived as straightforward linear combinations of the direct problem solutions. Consequently, our focus will be confined to resolving the direct problems, specifically the scattering of waves through point source excitation. In [2], a total of 122 emitters/receivers were utilized, positioned 72 m apart and located 216 m beneath the upper boundary in the water layer. The frequencies considered in that study are 2, 4, and 6 Hz.

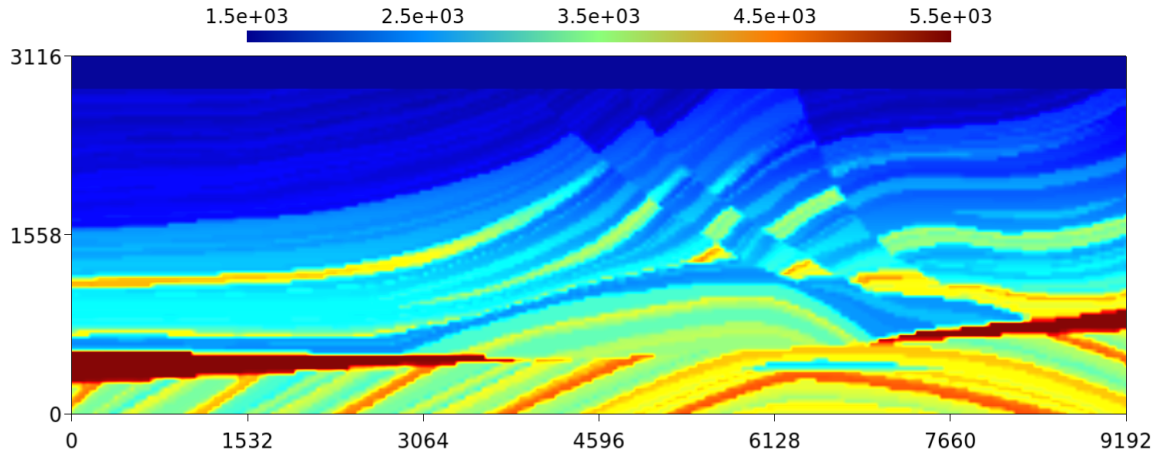


Figure 4.18: Velocity field of the Marmousi model.

As part of the research conducted in [2], the Marmousi model is employed for full waveform inversion without the domain decomposition method. In this study, the classical finite element method is employed for resolution. This approach yields favorable results due to the efficient resolution capabilities of direct solvers in two-dimensional scenarios. Moreover, these solvers can even leverage the parallel processing capabilities of modern computing systems. However, the ultimate goal of this method is to extend its applicability to three-dimensional scenarios. Practical applications often demand three-dimensional resolutions of the full waveform inversion, making it highly desirable. Nonetheless, three-dimensional finite element problems pose challenges for direct solvers, as even a relatively modest number of degrees of freedom can

lead to substantial memory and computational demands. As a result, this study focuses on the domain decomposition method in two dimensions, serving as an initial step towards tackling three-dimensional cases. This preliminary investigation aims to provide insights into the distinctive properties of decomposition method and how to improve it with subspace recycling.

As explored in Section 4.1.2, the decomposition of the domain significantly decreases the number of degrees of freedom of the problems handled by the direct solver. However, due to the two-dimensional context, this reduction does not necessarily translate to a decrease in computational time since this kind of problems are already efficiently managed by direct solvers. Nevertheless, memory requirements are diminished, although the reduction will not be substantial. The size of the interface problem in two dimensions will be smaller than volume problems when the subdomains considered are sufficiently large. This relationship emerges because the ratio of degrees of freedom between the interface and volume problems is approximately proportional to the ratio of the surface area to the perimeter of the subdomains. Consequently, the increase in memory consumption remains gradual, and it's only when a substantial number of subdomains are employed that the memory usage becomes comparable to that of the monodomain resolution. This situation arises when the size of the interface problem approaches the size of the volume problem, as observed here when the number of subdomains exceeds 25. The diverse characteristics associated with varying numbers of subdomains are condensed into separate plots within Figure 4.19. The conclusions drawn from these findings align with those reached for the homogeneous case.

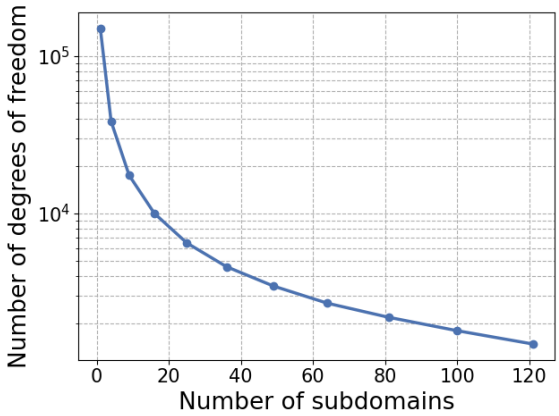
As this approach will be extended to three dimensions, it is foreseeable that both memory and computational demands for the monodomain case will exhibit unfavorable scalability with the number of degrees of freedom in the case of the monodomain resolution. Consequently, the disparity between the monodomain approach and the domain decomposition method is likely to become more pronounced. While the relative size of the interface problem compared to the volume problems might not be as small as in the two-dimensional case, the advantages gained from the factorization of smaller volume problems are anticipated to be significant. This contrast may become evident not only in terms of memory usage but also in the improved computational efficiency attainable through the domain decomposition approach.

Based on the distinct consumption patterns observed during the resolution of Marmousi scattering problems with varying numbers of subdomains, as illustrated in Figure 4.19, a suitable compromise must be found. On one side, the interface problem must be sufficiently large to capture its potential nuances, while on the other side the computational time must be maintained manageable. As a result of this consideration, resolutions will predominantly employ 25 subdomains in the following. The solutions for a single source at frequencies of 2, 4, and 6, Hz are depicted in Figure 4.20.

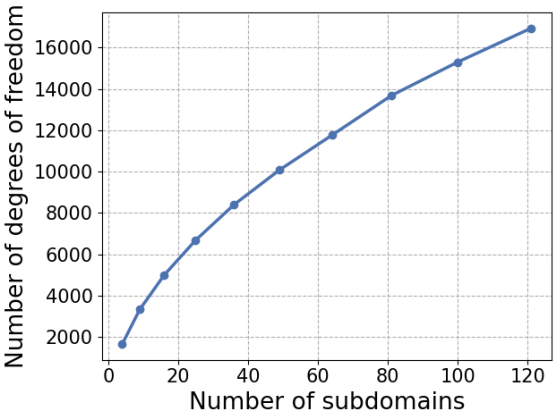
4.2.1 Recycling strategy

Moving away from the context of a homogeneous medium, the subspace recycling strategy will now be employed to accelerate the simulation of scattering waves in a more practical application. Here, the frequencies considered will range from 1 to 8 Hz, with each frequency utilizing a mesh that ensures approximately 25 nodes represent one wavelength. Consequently, different mesh sizes will be employed for different frequencies, allowing for comparisons between frequencies using either an appropriate mesh or a unique mesh.

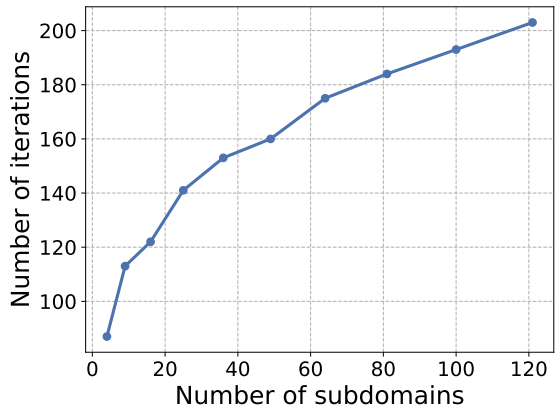
First, let us examine the simulation results for a frequency of 4 Hz with and without the recycling strategy, utilizing a domain decomposition into 25 smaller subdomains as illustrated in



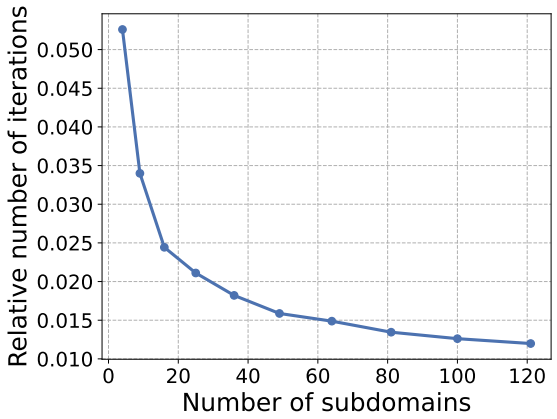
(a) Number of degree of freedom of one volume problem with respect to the number of domains.



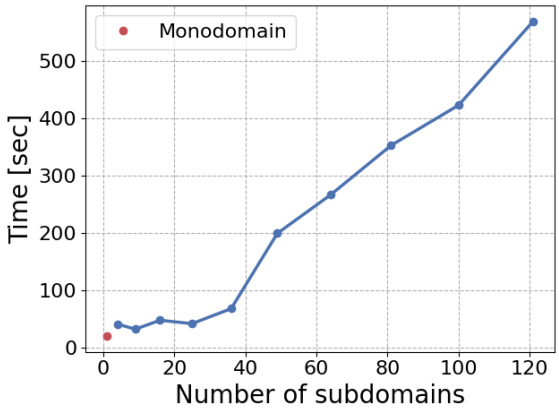
(b) Number of degree of freedom of the interface problem with respect to the number of domains.



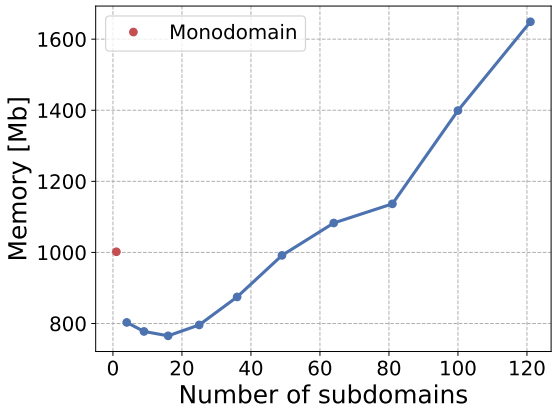
(c) Number of iteration for the interface to converge.



(d) Number of iterations to converge divided by the number of degrees of freedom.

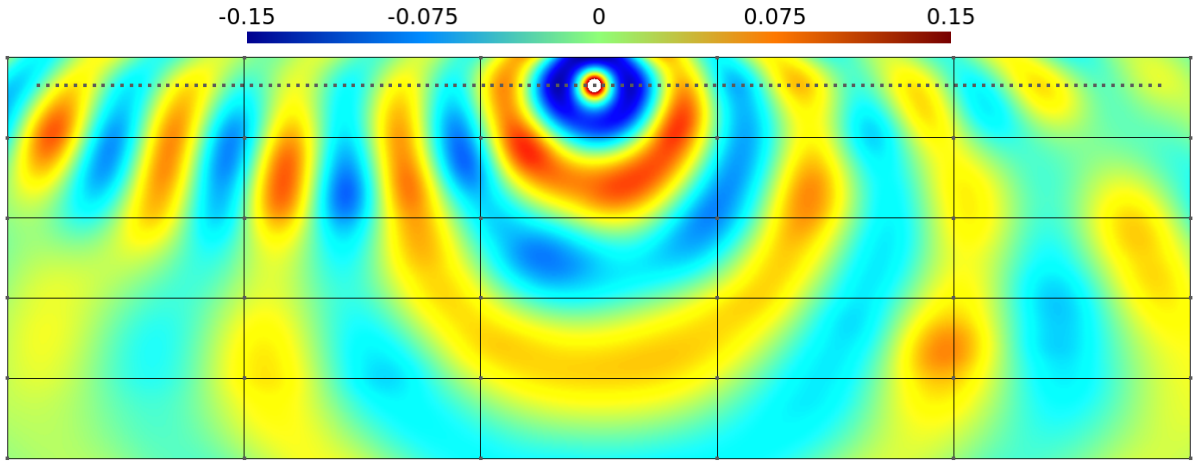


(e) Time taken by the resolution on a single node (without MPI).

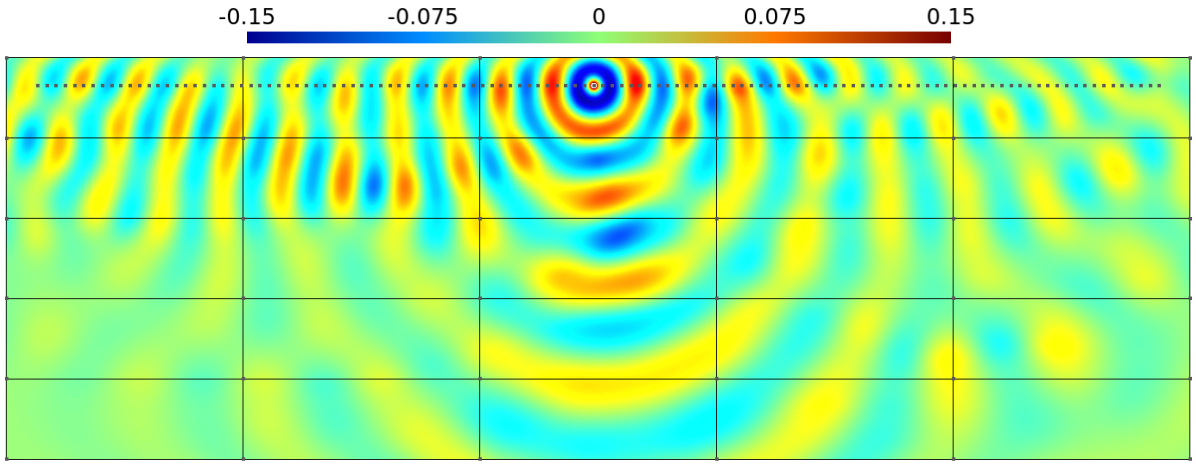


(f) Memory consumption (Maximum resident set size).

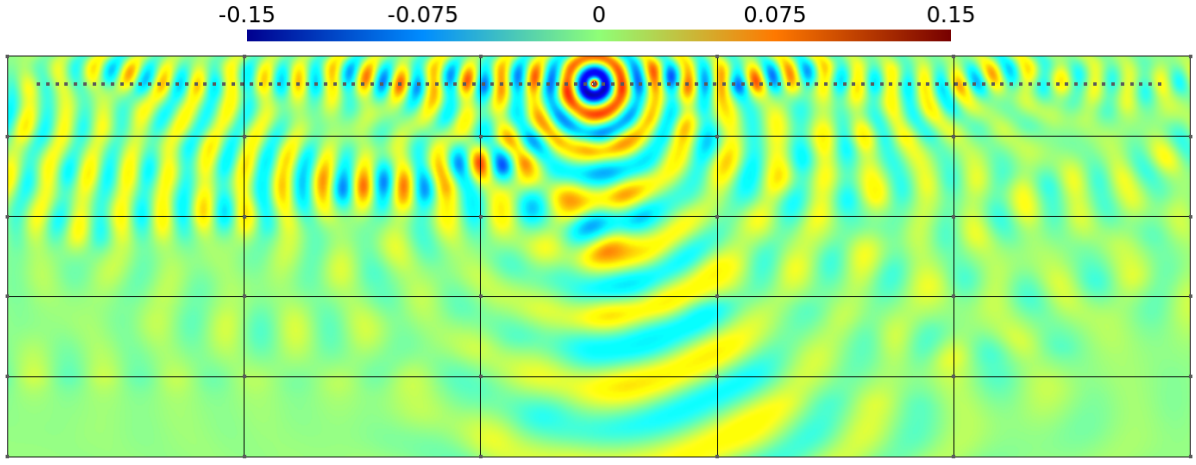
Figure 4.19: Consequences of the number of domains for the resolution of one system on the Marmousi model.



(a) For the frequency of 2 Hz.



(b) For the frequency of 4 Hz.



(c) For the frequency of 6 Hz.

Figure 4.20: Real part of the solution to the wave scattering in the Marmousi model using the domain decomposition method.

Figure 4.21. In order to obtain an accurate solution, a mesh size of 15 m will be required which leads to 16,1102 degrees of freedom for the entire domain considered. The related volume problems will have on average 6444 degrees of freedom while the interface problem correspond to 6680 degrees of freedom. This comparison clearly highlights once again the substantial potential of recycling methods in significantly reducing the number of iterations. In the absence of recycling, the total iteration count is approximately 17,000, whereas the recycling strategy reduces it to only around 3000, corresponding to less than 18% of the matrix vector products. This represents a significant reduction in the overall number of operations and, consequently, simulations employing the recycling strategy require just 35% of the computational time compared to classical resolution. For instance, on a single NIC5 node with 32 threads (OpenMP threads), the recycling-based simulation takes only 28 minutes, whereas the classical resolution would take 1 hour and 22 minutes.

An important observation must be done on the increase in the number of iterations as source points near interfaces are considered. Unlike in the homogeneous case discussed in Section 4.1, these additional iterations are not mainly due to proximity of the excitation point to an interface. This increase is present in non-recycling simulations as well, but its magnitude is far smaller compared to the sharp rise observed in the recycling-based simulations. Furthermore, this augmentation is not a localized phenomenon, the iteration count begins to rise even before reaching the point closest to the interface. Moreover, this increase does not immediately drop back after resolving the problem related to point the closest to the interface. Instead, it decreases again only after several subsequent right-hand sides. This rise in the iteration count can then be primarily attributed to the change of domain of the solutions. The recycled directions might not be adequate to improve the resolutions of points in or close to another domain. Therefore, more iterations will be required in comparison to the other resolutions. However, as iterations accumulate and new directions are recycled, the recycled subspace becomes better equipped to capture the next solutions, resulting in a subsequent decrease in iteration count.

Another noteworthy observation is the diminished impact of the relative initial iteration increase in recycling simulations for the first few resolutions. This is in contrast to the smaller homogeneous medium scenario detailed in Section 4.1. Additionally, in the extreme points of the domain, a high iteration count is required without recycling, attributed to the largely varying amplitude of the related solution across the whole domain. The magnitude disparity leads to substantially different solutions on the interface, causing an additional computational cost for extreme points. When recycling is employed to accelerate the convergence of the method, the initial iteration increase remains evident for the first resolution due to the absence of recycling directions. However, it is worth noting that for the last resolutions, corresponding to the rightmost points, the slight increase observed in the non-recycling case is no longer present. Even if this increase is limited, the recycling method might contribute to reducing the augmentation of the number of iteration associated with solutions exhibiting large magnitude differences.

4.2.2 Influence of frequency

The first point of the following analysis will be to examine how variations in frequency impact the resolution. This holds particular significance in practical scenarios where multiple frequencies are used for full waveform inversion. To eliminate mesh-dependence, a consistent mesh is employed across all simulations. Given that the 8 Hz simulation necessitates a maximum mesh size of $l_c = 7.5\,\text{m}$, this mesh size is adopted for all frequency simulations.

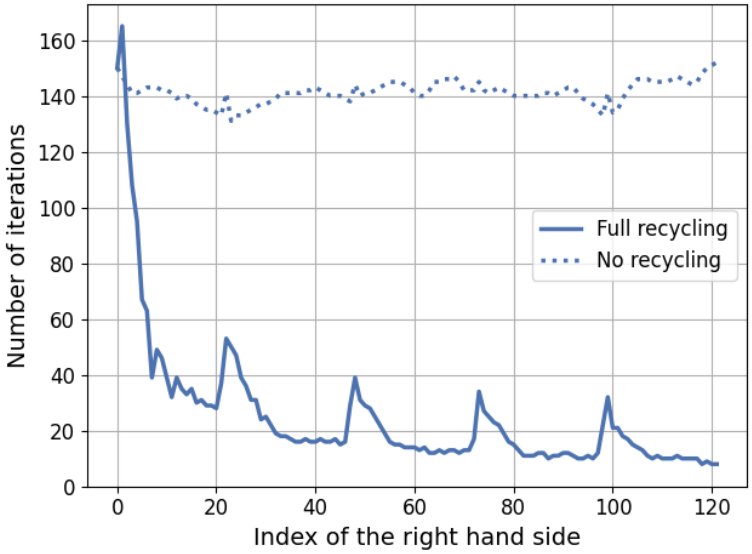


Figure 4.21: Number of iteration required for each right hand side with total recycling or without recycling for a frequency of 4 Hz.

The outcomes regarding different frequencies are illustrated in Figure 4.22. Notably, a trend emerges: as frequency decreases, the improvement in resolution through recycling becomes more pronounced. Despite this trend, the recycling approach remains highly advantageous even at higher frequencies. For instance, with the highest frequency under consideration, the total iteration count decreases from around $19,000$ to just $5,000$ iterations, representing roughly a quarter of the total iterations.

It is important to emphasize that the problem remains identical across all frequencies, resulting in a different spacing between source points relative to the wavelength. This minor distinction might account for the varied effectiveness of the recycling approach. Further exploration involving problem resolution while maintaining a relatively constant spacing relative to the wavelength will then provide valuable insights.

Additionally, it is noteworthy that when source points near the interfaces are considered, the iteration count increases for all frequencies, albeit more significantly for lower frequencies than for higher ones. This phenomenon could be attributed to the fact that simulations at lower frequencies tend to accumulate fewer directions on average. Consequently, when a substantial change in the solution arises, these do not provide a large improvement to the resolution and a larger number of iteration are required to solve the interface problem. In contrast, simulations at higher frequencies amass a greater number of recycled directions as points close to the interfaces are addressed. Consequently, a larger space is described by the recycled directions which can then enhance the convergence of simulations.

An interesting observation when using a single mesh and varying the considered frequency is that the number of iterations required to solve the interface problem without recycling increases for lower frequencies, as evident in the first column of Table 4.1. However, the recycling strategy demonstrates greater efficiency for lower frequencies, as illustrated in Figure 4.22. As a consequence, the number of iterations decreases rapidly for low frequencies due to the recycling approach. Without the recycling method, resolving low frequencies with the domain decomposition method on a fine mesh turns out to be more computationally expensive than

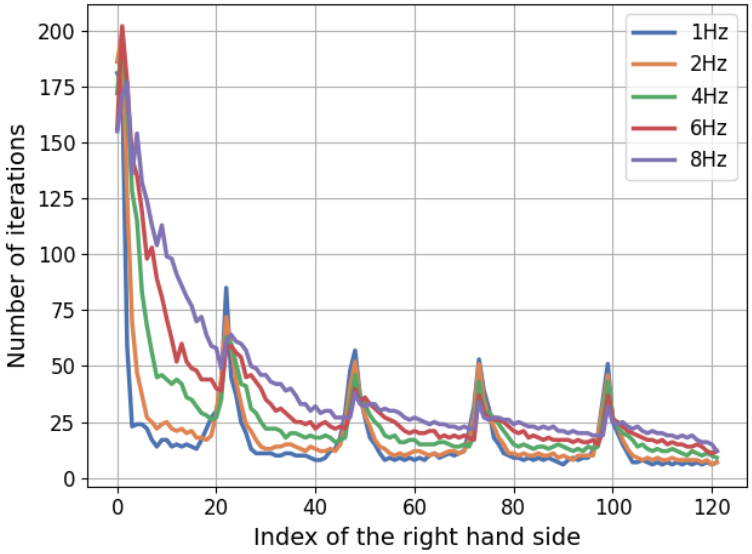


Figure 4.22: Number of iteration required for each right hand side with total recycling and for different frquencies. Also the mesh considered here is the same for all frequencies considered.

resolving high frequencies (assuming the mesh is always sufficient to represent the solutions accurately). This contradicts the intuitive expectation that high frequencies, which capture more complex solution structures, would require more computation. Nevertheless, with the recycling technique, resolving low frequencies becomes more advantageous. This phenomenon can be explained by the fact that solutions for low frequencies exhibit simpler structures, as depicted in Figures 4.20 and therefore two distinct solution are more prone to share similarities.

Frequency	**Total**	**System**		
		First	**Second**	**Third**
$f = 1$	24686	181	182	184
	With recycling: 2287	181	176	59
$f = 2$	22009	186	187	186
	With recycling: 2682	186	200	126
$f = 4$	19764	172	169	166
	With recycling: 3506	172	196	168
$f = 6$	18947	156	158	160
	With recycling: 4317	156	202	177
$f = 8$	18943	155	171	177
	With recycling: 5093	155	157	158

Table 4.1: Number of iterations required to solve the Marmousi scattering problem with different frequencies on a single mesh ($l_c = 7.5\,\mathrm{m}$).

4.2.3 Influence of the spacing of the sources

As it was shown in Figure 4.22, the performance of the recycling method vary might with the frequency. To investigate whether these variations in iteration counts across different frequencies are attributed to the differences in the spacing of the excitation sources or not, the change of frequency will be explored while maintaining a constant spacing relative to the

wavelength. This scenario is illustrated in Figure 4.23. Remarkably, when the spacing between the sources is kept constant in relation to the wavelength, the differences between frequencies in terms of recycling improvements disappear. This suggests that the frequency of the wave being considered does not significantly impact the efficiency of recycling. Instead, it appears that the spacing between the sources holds a more substantial influence.

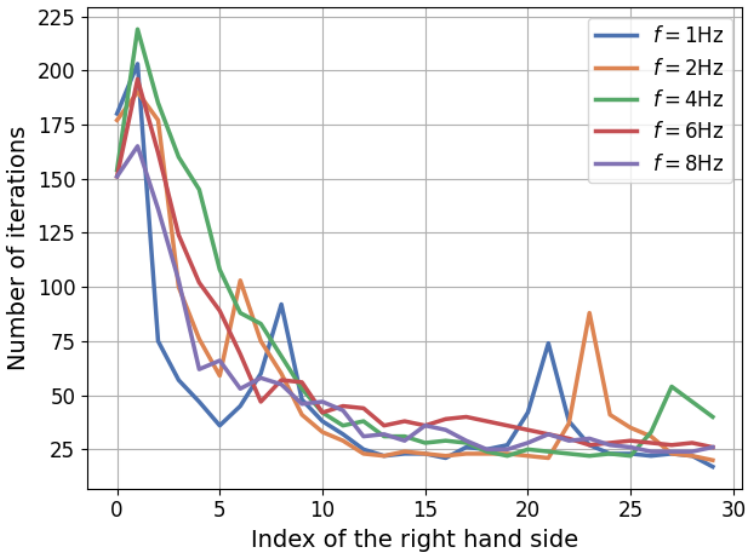


Figure 4.23: Number of iteration required for each right hand side with total recycling and for different frequencies while keeping the spacing at a constant wavelength length (0.192λ).

The impact of source spacing on recycling improvements can be observed in Figure 4.24. When this spacing is relatively small compared to the wavelength, the reduction in iteration count becomes more pronounced as the spacing decreases. This observation can help explain the differences seen in Figure 4.22. In that case, while the spacing remains constant at $d_s = 72\,\mathrm{m}$, the spacing in terms of the wavelength varies from $0.048\,\lambda$ for $1\,\mathrm{Hz}$ to $0.384\,\lambda$ for $8\,\mathrm{Hz}$. Nevertheless, it is worth noting that as the spacing approaches the size of the wavelength, the diminishing effect on the iteration reduction becomes less significant, and there comes a point where the recycling improvements are no longer influenced by the spacing. It is important to highlight that recycling still provides substantial benefits even for larger spacings. As a result, this method remains effective across various spacings between the source points. Additionally, there might be a slight enhancement in the performance of the recycling method by using small spacings relative to the wavelength.

4.2.4 Influence of the mesh size

An important facet of the method lies in its adaptation to various mesh configurations. This is a significant aspect to explore, as it allows us to observe how the method behaves when applied to more refined meshes. This analysis is presented in Figure 4.25, where the number of iterations required for convergence is depicted for different right-hand side problems using the recycling method. To comprehensively cover a range of mesh conditions, the simulations with a frequency of $1\,\mathrm{Hz}$ are considered.

Notably, without the recycling method, as the mesh is refined, the number of iterations required to solve a problem increases as the mesh size becomes smaller. This trend is evident

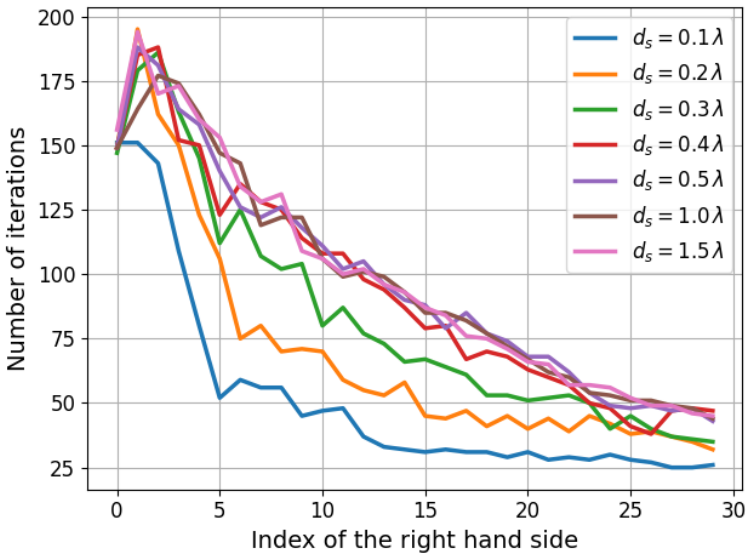


Figure 4.24: Number of iteration required for each right hand side with total recycling and for different spacing in term of wavelength (6[Hz]).

in the initial system shown in the figure and is explain by the fact that smaller mesh sizes lead to a larger number of degree of freedom and therefore to a larger number of iterations to converge. However, a noteworthy observation is that the recycling method greatly enhances the resolution for all mesh configurations. Moreover, it consistently achieves a significantly low number of iterations across different mesh sizes. Consequently, the resolution outcomes for all considered meshes become highly similar, with only slight discrepancies at the beginning and near interfaces can be distinguished.

As anticipated, there is an augmentation in the iteration count near interfaces. However, this increase is also influenced by the refinement of the mesh. Specifically, the finer the mesh, the larger the increase in iteration count. This behavior can be attributed to the transition of the excitation point from one subdomain to another, causing a substantial change in the solution behavior along the interface. Given this variability, the current recycled subspace fails to accurately capture parts of the solution, leading to an increase in the iterations. The mesh size dependence of this phenomenon can be explained by the fact that as the mesh becomes finer, the number of degrees of freedom along the interface increases. Therefore, if the solution exhibits substantial changes, more directions are required to account for the variations in the refined mesh scenarios.

4.2.5 Recycled subspace selection strategies

In this context, the partial subspace recycling approach can also be employed with all the solvers. Also, the same selection strategies as those discussed in Section 4.1.4 can be applied. However, it is important to note that the total number of right-hand sides considered here is significantly larger, therefore an effective selection strategy will be even more important.

In the first plot of Figure 4.26, the various selection strategies are applied to a the simulation of waves with a frequency of 4 Hz, resulting in an interface problem with 6680 degrees of freedom. The limits on the maximum number of stored directions is set to a very small number of 500 directions. In this plot, it becomes evident that the strategy involving the largest

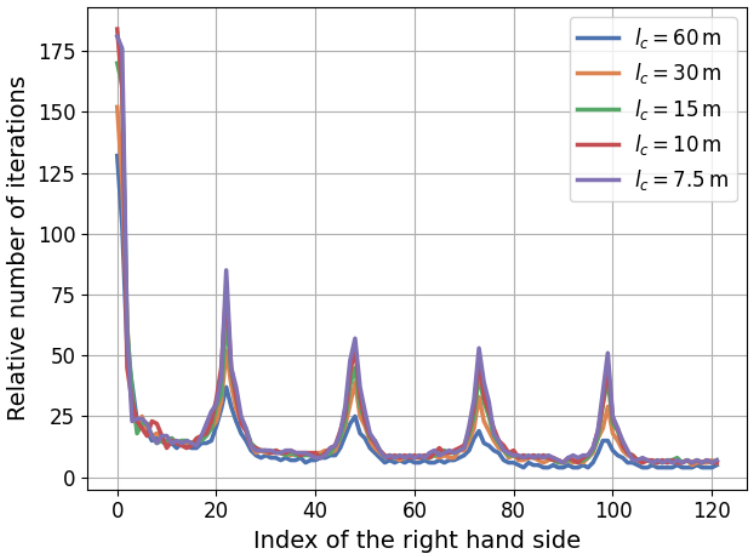


Figure 4.25: Number of iteration required for each right hand side with total recycling and for different mesh size l_c (4[Hz]).

orthogonalization coefficient outperforms the others. However, the two other strategies, the one retaining the initial directions and the one selecting the directions that led to the most substantial decrease in the residual norm, are not far behind in terms of performance. The only strategy that clearly performs poorly is the one that retains the last directions.

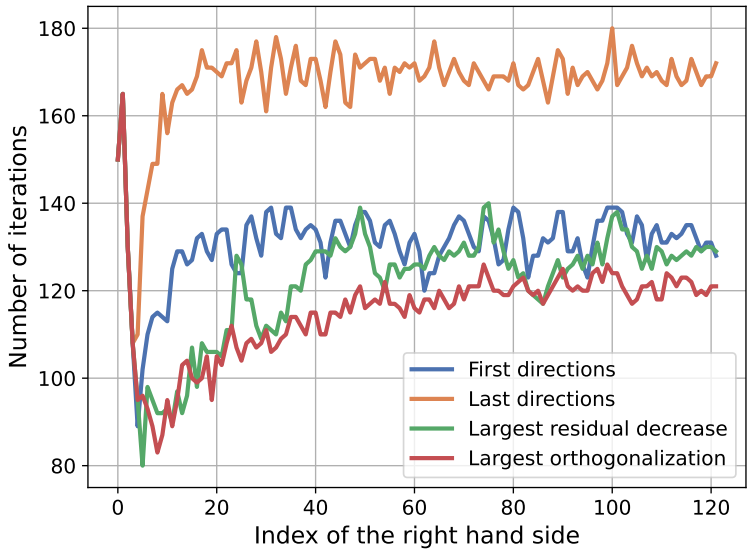
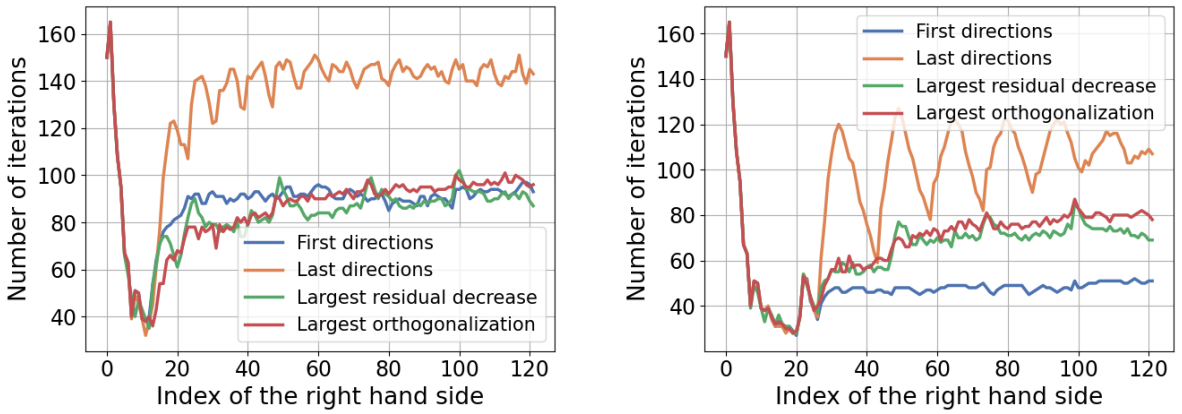


Figure 4.26: Number of iteration required for each right hand side with different recycling strategies and with a maximum of 500 recycled directions.

The strategy keeping the directions corresponding to the largest orthogonalization coefficients might be a good strategy when a small number of directions are recycled. However, as the number of recycled direction increases, the performances of the differents strategies does not remain the same. A similar simulation will be considered, but this time with a maximum of

1000 and 1500 directions are recycled. The results obtained with different selection strategies are represented in Figure **??**. Interestingly, the strategy based on selecting the largest orthogonalization coefficient exhibits a less favorable behavior in this scenario. The strategy that focuses on the largest residual norm decrease also performs comparatively less effectively. And as before, the strategy of keeping the last direction continues to show suboptimal results. On the other hand, the strategy of retaining the first direction maintains a consistent performance. This strategy appears to be more resilient and robust compared to the others. Additionally, the slight disadvantage of choosing the first direction strategy in the previous scenario indicates that, in practical applications, opting for the first direction recycling strategy remains a solid choice. Nevertheless, further comparisons and experiments need to be conducted to confirm this assertion.

A last interpretation can be done on the strategy keeping the last directions. Indeed, on the Figure 4.27b, this strategy leads to periodic fluctuations in the required number of iterations. One plausible explanation for this phenomenon is that the initial directions play a crucial role in recycling as proved by the performance of the corresponding strategy. When these initial directions are discarded, the method quickly loses its recycling improvement. As the simulation progresses, the augmentation space regains different directions that might contribute more effectively to recycling. Consequently, the performance of the method improves again. However, as these crucial directions are replaced with new ones, the performance diminishes once more, leading to a recurring cycle of periodic fluctuations.



(a) For a maximum of 1000 recycled directions.

(b) For a maximum of 1500 recycled directions.

Figure 4.27: Number of iteration required for each right hand side with different recycling strategies and with large numbers of recycled directions.

4.2.6 Computational time of the solvers

One important consideration when selecting a solver to address the interface problem lies in the computational demands. As demonstrated in Section 3.3, the number of operations should be comparable for GMRES-type methods, while GCR and Orthodir methods require about fifty percent more operations, excluding matrix vector products. However, given that matrix vector products are known to be computationally intensive, they will constitute a significant portion of the computational time. Since the number of matrix vector products remains consistent across all methods, the discrepancies among the methods will contribute only to a small portion to the overall computational load. To validate this, the computational times of the various methods are depicted in Figure 4.28. This visualization clearly illustrates that

there are negligible differences in computational time among the methods. Any minor variations are attributed to measurement noise, as even between GMRES-type methods and GCR or Orthodir, no discernible differences can be observed. In most scenarios, the predominant computational workload will stem from matrix vector products. Consequently, the disparities in computational requirements between the methods are generally negligible. In cases where differences are detectable, they typically exist between GMRES-type methods and the alternatives. As no discernible benefits arise from using methods other than GMRES-type methods, the latter should be the preferred choice in most situations.

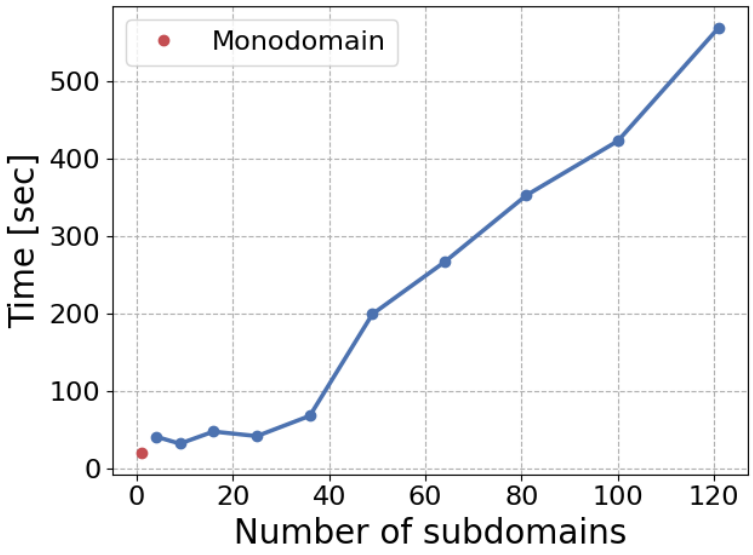
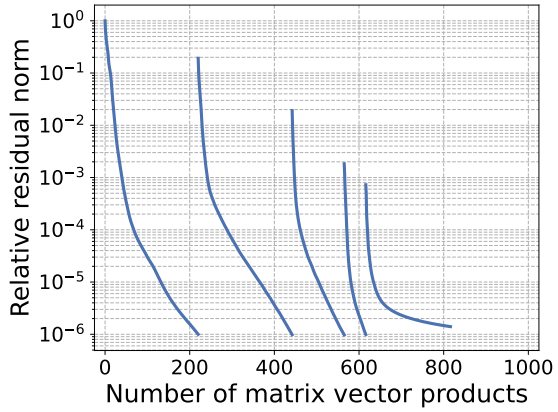


Figure 4.28: Computational time of the different solvers with recycling.

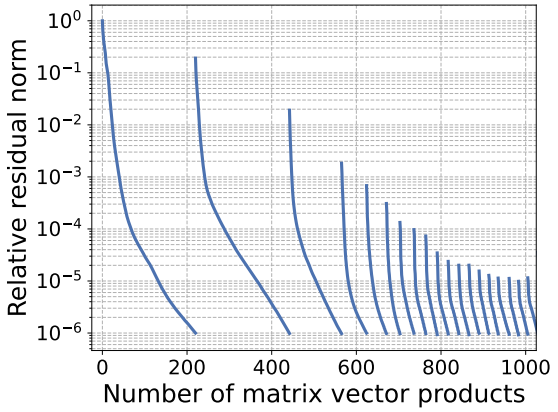
4.2.7 Stability of the solvers

As a final study, we will highlight the limitations of the methods in terms of numerical stability by exploring problems with an even higher number of degrees of freedom in this section. The focus will be on the scattering of waves with a frequency of 2 Hz and a mesh size of $l_c = 10\,\text{m}$. Similar to previous cases, the domain will be divided into 25 subdomains. However, a change will be made in the element order, employing second-order elements to further increase the number of degrees of freedom. Consequently, the volumetric problems will have 54899 degrees of freedom, while the interface problem will involve 19840 degrees of freedom. In this scenario, the GMRES method will be tested using both the classical and the modified Gram-Schmidt orthogonalization techniques. Similar observations could be drawn with all solvers. The outcomes are depicted in Figure 4.29.

Upon examination, it is evident that the classical approach fails to provide a solution for the interface problem after only the fifth right-hand side is considered. The solution stagnates at a value exceeding the tolerance threshold, therefore the method is unable to reach a satisfactory solution. This starkly highlights the potential pitfalls of using the classical Gram-Schmidt orthogonalization, particularly when confronted with sizable interface problems. Furthermore, adopting the modified Gram-Schmidt process incurs no additional costs, except for a potential loss of parallelization opportunity. However, given that the benefits of parallelization would not substantially outweigh other associated costs, such as the matrix vector product computations, the practical consensus favors the use of the modified Gram-Schmidt method.



(a) Using the classical Gram-Schmidt orthogonalization.



(b) Using the modified Gram-Schmidt orthogonalization.

Figure 4.29: Residual decrease for multiple right hand sides of a large scale problem using recycling methods with GMRES. More precisely, the plotted values represent the estimated residuals obtained from the method iterations.

As said, in the context of the problems considered here, there are no noticeable differences in the numerical stability between the different methods. Even if the numerical instability of the SGMRES and Orthodir methods were proven in [**?**] for some specific examples, no reductions in the convergence was notices in all the different cases studied along this work. Therefore, in the context of the problems resolved here and considering problems in the same range of sizes, the choice of the method in not crucial from that perspective. However, as it was shown in the last example, as the size of the problem is increased, the numerical stability of the solvers becomes more essential to find the solution to the problem. Then, more exploration at significantly larger scales should be studied to find where the numerical stability of the different methods start to show its limits. This can correspond both to a large the size of the problem and to the nature of the problem considered. In [32], concretes example of problem causing numerical instabilities are given.

4.3 Conclusion on the practical use of recycling Krylov solvers

The current chapter explore the practical use of Krylov iterative solvers paired with recycling strategies. Specifically applications within the context of domain decomposition methods for scenarios in two dimensions are considered. These applications involve tackling a sequence of linear problems with different right-hand sides. Notably, it is demonstrated that these techniques yield significant advantages by considerably reducing the necessary iteration count to achieve convergence for the interface problem. Considering the inherent suitability of these methods for a possible expansion into three-dimensional scenarios, future works should be considered to address such cases with subspace recycling methods.

An analysis of the different parameters of the application considered was conducted, yielding several insightful conclusions on the performance of the subspace recycling strategy. The number of subdomains was found to exert limited influence on the recycling performance, as long as the interface problem possesses a certain minimum size. In the context of full waveform inversion, it was revealed that the primary factor impacting the efficiency of the recycling method was the spacing of source points. Proximity between excitation points correlated with

an enhanced recycling strategy performance when the spacing was small relative to the wavelength. However, as the spacing approached approximately half the wavelength and exceeded it, the influence of spacing on recycling performance ceased to exist. Additionally, the study indicated that frequency had negligible impact on recycling performance when the spacing was held constant in terms of wavelength. Another noteworthy observation was that varying mesh densities led to an increased number of iterations being necessary to solve one problem with finer meshes. However, the recycling method effectively mitigated this difference in iteration counts. Throughout all the examined test cases, there was no instance that demonstrated the ineffectiveness of the total subspace recycling method. On the contrary, this method consistently led to significant reductions in the total number of iterations required by the iterative approach.

Moreover, a concise demonstration illustrated the feasibility of partial subspace recycling coupled with judiciously selected search directions. This presentation underscored the substantial impact of the search direction selection strategy on the performance of the recycling method. Notably, the findings consistently underscored that retaining the initial directions proved most effective, producing satisfactory outcomes across a majority of scenarios.

The final observation of this study indicates the absence of significant differences among the various solvers considered in terms of computational requirements. This can be attributed to the substantial operation count and memory demands inherent in the domain decomposition framework. As a result, these demands overshadowed any variations arising from the iterative resolution of the interface problem. Notably, the most resource-intensive operation within the iterative methods is the matrix vector product. However, all the considered methods required approximately the same amount of this particular operation, resulting in similar performance across these methods. Additionally, the memory consumption associated with the recycled directions will be negligible in comparison to the storage required by the volume problems when large subdomains are considered. Furthermore, the analysis determined that the application of the modified Gram-Schmidt orthogonalization technique within the iterative methods ensured the absence of numerical stability issues across the examined problems. Therefore, in this specific context, selecting a recycling Krylov solver over another is unlikely to result in substantial discrepancies in outcomes.

Conclusion and perspectives

In the Chapter 1, the full waveform inversion framework is introduction in the context of subsurface tomography. This method require the resolution of large scale finite element simulations, i.e. large linear problems, of wave scattering in time-harmonics. This then lead to interested of using the domain decomposition method in order to handle large scale problem and therefore the need to use iterative methods to solve the problem at the interfaces. On top of that the full waveform inversion also require the resolution of a sequence of linear problems with varying right hand sides. This then form an examples of the need to efficiently resolve a sequences of linear problems with varying right hand sides using iterative methods.

Then in the Chapter 2, an in depth presentation of Krylov subspace iteration methods is presented with a particular focus on minimization solvers. This concern three kind of solvers: the GCR, GMRES and SGMRES methods. Also the concept of residual based algorithms is presented and lead to the variation of some algorithms. From that study, it was outlined that the main characteristics of the solvers are the computation and memory consumption as well as there numerical stability. It is outlined that the GMRES method stand out as the most advantageous method on all characteristic. This is do to the fact that method like GCR require almost twice as much storage and fifty percent more operation than most of GMRES-type methods. Also, GMRES possesses the more stable numeric behavior in theory than all the others even if in practice no clear difference is measured with residual based methods.

In the Chapter 3, the extension of the Krylov iteration method is presented notably to enable the efficient resolution of sequence of linear problem with different right hand sides. There, the augmentation technique is derived for all solvers and its particular application to the subspace recycling is presented. This enables then to reuse the directions used in each resolutions for the next resolutions. From that, chapter one can conclude that almost all the methods will have roughy the same storage consumption. However, the additional cost associate to method like GCR combined with recycling still remains and therefore GMRES-type method are still the most advantageous. This time the GMRES method is still vary advantageous but other methods have close characteristics. Also, it is noted that the SGMRES method can provide a reduction in memory consumption with respect to the other methods. Lastly, the partial recycling strategy is presented for each methods and selection strategy are briefly proposed.

In Chapter 4, the practical application of subspace recycling iterative solvers is explored in the context of two-dimensional wave scattering problems using domain decomposition. Significant improvements achieved through subspace recycling is demonstrated across different problem scales. The impact of problem parameters on the performance of the recycling strategy is also studied. The findings show that the recycling methods remain effective even for a large number of subdomains. Moreover, certain parameters, such as a small spacing between

sources, enhance recycling performance in this context. Interestingly, we also observe that the recycling method remains effective even with larger spacing. Among all the tests conducted, the total recycling method consistently proves highly effective, leading to a substantial reduction in the total iteration count of the iterative solver. Throughout the study, various aspects indicate the potential extension of this method to the three-dimensional case, suggesting the feasibility of such an extension and its potential for significant improvements. Furthermore, the feasibility of partial subspace recycling was observed and the selection of search directions is discussed. This demonstration emphasizes the considerable influence of the search direction selection strategy on the performance of the recycling method. The final conclusion of this chapter is that, in the specific example of resolution using the domain decomposition method, no major differences between the iterative solvers can be discerned. Consequently, any recycling method can be employed without causing a measurable impact on performance.

There are several potential extensions of the current work that could be explored. Regarding the specific application presented here, which is full waveform inversion, the ultimate goal would involve extending the methods to a three-dimensional case. This extension would primarily necessitate adaptations in the domain decomposition example presented in this study. However, no more adaptation of the recycling iterative solvers are necessary. Numerous indicators strongly suggest that such an extension could yield compelling results, making it a worthy subject for future research. Also, in the specific application studied here, the right-hand sides are available simultaneously. Therefore, it is possible that promising methods like the Block Krylov method could lead to even greater improvements in resolution [1], [50]. Comparing these methods with the ones presented in this work would be interesting. Additionally, it is worth highlighting that the recycled Krylov solvers detailed in this study are not confined to domain decomposition methods. Exploring their application in other practical scenarios could be intriguing, potentially unveiling diverse performance outcomes among various solvers. Finally, one limitation of the recycling Krylov methods presented here is their applicability only to sequences of linear problems with different right-hand sides but with the same matrix. It could be intriguing to consider extending this method to linear problems where the matrix is slightly perturbed, as it is the case in [47].

Bibliography

[1] F.-X. Roux and A. Barka, "Block krylov recycling algorithms for feti-2lm applied to 3-d electromagnetic wave scattering and radiation," *IEEE Transactions on Antennas and Propagation*, vol. 65, no. 4, pp. 1886–1895, 2017.

[2] X. Adriaens, *Inner product preconditioned optimization methods for full waveform inversion*. Phd thesis, Université de Liège, December 2022.

[3] R. G. Pratt, W. J. McGaughey, and C. H. Chapman, "Anisotropic velocity tomography: A case study in a near-surface rock mass," *GEOPHYSICS*, vol. 58, no. 12, pp. 1748–1763, 1993.

[4] R. G. Pratt, "Seismic waveform inversion in the frequency domain, part 1: Theory and verification in a physical scale model," *GEOPHYSICS*, vol. 64, no. 3, pp. 888–901, 1999.

[5] R. Feynman, R. Leighton, M. Sands, and E. Hafner, *The Feynman Lectures on Physics; Vol. I*, vol. 33. AAPT, 1965.

[6] S. H. Schot, "Eighty years of sommerfeld's radiation condition," *Historia Mathematica*, vol. 19, no. 4, pp. 385–401, 1992.

[7] X. Antoine, H. Barucq, and A. Bendali, "Bayliss–turkel-like radiation conditions on surfaces of arbitrary shape," *Journal of Mathematical Analysis and Applications*, vol. 229, no. 1, pp. 184–211, 1999.

[8] X. Antoine, M. Darbas, and Y. Y. Lu, "An improved surface radiation condition for high-frequency acoustic scattering problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 33, pp. 4060–4074, 2006.

[9] E. Turkel and A. Yefet, "Absorbing pml boundary layers for wave-like equations," *Applied Numerical Mathematics*, vol. 27, no. 4, pp. 533–557, 1998. Special Issue on Absorbing Boundary Conditions.

[10] A. Bermúdez, L. Hervella-Nieto, A. Prieto, and R. Rodrı´guez, "An optimal perfectly matched layer with unbounded absorbing function for time-harmonic acoustic scattering problems," *Journal of Computational Physics*, vol. 223, no. 2, pp. 469–488, 2007.

[11] X. Adriaens, F. Henrotte, and C. Geuzaine, "Adjoint state method for time-harmonic scattering problems with boundary perturbations," *Journal of Computational Physics*, vol. 428, p. 109981, 2021.

[12] J.-F. Debongnie, *Fundamentals of finite elements*. Les Editions de l'Université de Liège, 2003. D/2003/8886/5. Tous droits de reproduction, d'adaptation et de traduction réservés pour tous pays.

[13] P. Solin, K. Segeth, and I. Dolezel, *Higher-order finite element methods*. CRC press, 2003.

[14] O. G. Ernst and M. J. Gander, *Why it is Difficult to Solve Helmholtz Problems with Classical Iterative Methods*, pp. 325–363. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[15] V. Dolean, P. Jolivet, and F. Nataf, *An Introduction to Domain Decomposition Methods*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2015.

[16] B. Thierry, A. Vion, S. Tournier, M. El Bouajaji, D. Colignon, N. Marsic, X. Antoine, and C. Geuzaine, "Getddm: An open framework for testing optimized schwarz methods for time-harmonic wave problems," *Computer Physics Communications*, vol. 203, pp. 309–330, 2016.

[17] M. J. Gander and H. Zhang, "A class of iterative solvers for the helmholtz equation: Factorizations, sweeping preconditioners, source transfer, single layer potentials, polarized traces, and optimized schwarz methods," *SIAM Review*, vol. 61, no. 1, pp. 3–76, 2019.

[18] Y. Erlangga, C. Vuik, and C. Oosterlee, "On a class of preconditioners for solving the helmholtz equation," *Applied Numerical Mathematics*, vol. 50, no. 3, pp. 409–425, 2004.

[19] Y. A. Erlangga, C. W. Oosterlee, and C. Vuik, "A novel multigrid based preconditioner for heterogeneous helmholtz problems," *SIAM Journal on Scientific Computing*, vol. 27, no. 4, pp. 1471–1492, 2006.

[20] B. Engquist and L. Ying, "Sweeping preconditioner for the helmholtz equation: hierarchical matrix representation," *Communications on pure and applied mathematics*, vol. 64, no. 5, pp. 697–735, 2011.

[21] P. R. Amestoy, I. S. Duff, J.-Y. L'Excellent, and J. Koster, "A fully asynchronous multifrontal solver using distributed dynamic scheduling," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 1, pp. 15–41, 2001.

[22] S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. Gropp, *et al.*, "Petsc users manual," 2019.

[23] Y. Saad, *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second ed., 2003.

[24] M. R. Hestenes, E. Stiefel, *et al.*, "Methods of conjugate gradients for solving linear systems," *Journal of research of the National Bureau of Standards*, vol. 49, no. 6, pp. 409–436, 1952.

[25] H. C. Elman, *Iterative methods for large, sparse, nonsymmetric systems of linear equations*. Yale University, 1982.

[26] Y. Saad and M. H. Schultz, "Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*,

vol. 7, no. 3, pp. 856–869, 1986.

[27] W. Joubert, "On the convergence behavior of the restarted gmres algorithm for solving nonsymmetric linear systems," *Numerical linear algebra with applications*, vol. 1, no. 5, pp. 427–447, 1994.

[28] A. Baker, E. Jessup, and T. Kolev, "A simple strategy for varying the restart parameter in gmres(m)," *Journal of Computational and Applied Mathematics*, vol. 230, no. 2, pp. 751–761, 2009.

[29] R. Fletcher, "Conjugate gradient methods for indefinite systems," in *Numerical Analysis: Proceedings of the Dundee Conference on Numerical Analysis, 1975*, pp. 73–89, Springer, 1975.

[30] H. A. van der Vorst, "Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 2, pp. 631–644, 1992.

[31] M. H. Gutknecht, "Deflated and augmented krylov subspace methods: A framework for deflated bicg and related solvers," *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 4, pp. 1444–1466, 2014.

[32] P. Jiránek, M. Rozložník, and M. Gutknecht, "How to make simpler gmres and gcr more stable," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, pp. 1483–1499, 12 2008.

[33] *Orthomin, an Iterative Method for Solving Sparse Sets of Simultaneous Linear Equations*, vol. All Days of *SPE Symposium on Numerical Simulation of Reservoir Performance*, 02 1976.

[34] D. M. Young and K. C. Jea, "Generalized conjugate-gradient acceleration of nonsymmetrizable iterative methods," *Linear Algebra and its Applications*, vol. 34, pp. 159–194, 1980.

[35] W. E. Arnoldi, "The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Quarterly of applied mathematics*, vol. 9, no. 1, pp. 17–29, 1951.

[36] D. Bindel, J. Demmel, W. Kahan, and O. Marques, "On computing givens rotations reliably and efficiently," *ACM Transactions on Mathematical Software (TOMS)*, vol. 28, no. 2, pp. 206–238, 2002.

[37] H. F. Walker and L. Zhou, "A simpler gmres," *Numerical Linear Algebra with Applications*, vol. 1, no. 6, pp. 571–581, 1994.

[38] J. Drkošová, A. Greenbaum, M. Rozložník, and Z. Strakoš, "Numerical stability of gmres," *BIT Numerical Mathematics*, vol. 35, no. 3, pp. 309–330, 1995.

[39] A. Greenbaum, M. Rozložnik, and Z. Strakoš, "Numerical behaviour of the modified gramschmidt gmres implementation," *BIT Numerical Mathematics*, vol. 37, no. 3, pp. 706–719, 1997.

[40] Y. Saad, "A flexible inner-outer preconditioned gmres algorithm," *SIAM Journal on Scientific Computing*, vol. 14, no. 2, pp. 461–469, 1993.

[41] H. A. Van der Vorst and C. Vuik, "Gmresr: a family of nested gmres methods," *Numerical linear algebra with applications*, vol. 1, no. 4, pp. 369–386, 1994.

[42] E. de Sturler, "Nested krylov methods based on gcr," *Journal of Computational and Applied Mathematics*, vol. 67, no. 1, pp. 15–41, 1996.

[43] E. D. Sturler, "Truncation strategies for optimal krylov subspace methods," *SIAM Journal on Numerical Analysis*, vol. 36, no. 3, pp. 864–889, 1999.

[44] R. Morgan, "A restarted gmres method augmented with eigenvectors," *SIAM Journal on Matrix Analysis and Applications*, vol. 16, 10 1995.

[45] R. B. Morgan, "Gmres with deflated restarting," *SIAM Journal on Scientific Computing*, vol. 24, no. 1, pp. 20–37, 2002.

[46] Y. Lin, L. Bao, and Q. Wu, "Simpler gmres with deflated restarting," *Mathematics and Computers in Simulation*, vol. 82, no. 11, pp. 2238–2252, 2012.

[47] M. L. Parks, E. De Sturler, G. Mackey, D. D. Johnson, and S. Maiti, "Recycling krylov subspaces for sequences of linear systems," *SIAM Journal on Scientific Computing*, vol. 28, no. 5, pp. 1651–1674, 2006.

[48] D. Darnell, R. B. Morgan, and W. Wilcox, "Deflated gmres for systems with multiple shifts and multiple right-hand sides," *Linear Algebra and its Applications*, vol. 429, no. 10, pp. 2415–2434, 2008.

[49] M. L. Parks, K. M. Soodhalter, and D. B. Szyld, "A block recycled gmres method with investigations into aspects of solver performance," *arXiv preprint arXiv:1604.01713*, 2016.

[50] P. Jolivet and P.-H. Tournier, "Block iterative methods and recycling for improved scalability of linear solvers," in *SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 190–203, IEEE, 2016.

[51] K. M. Soodhalter, E. de Sturler, and M. E. Kilmer, "A survey of subspace recycling iterative methods," *GAMM-Mitteilungen*, vol. 43, no. 4, p. e202000016, 2020.

[52] J. Erhel and F. Guyomarc'h, "An augmented conjugate gradient method for solving consecutive symmetric positive definite linear systems," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1279–1299, 2000.

[53] P. Benner and L. Feng, "Recycling krylov subspaces for solving linear systems with successively changing right-hand sides arising in model reduction," in *Model Reduction for Circuit Simulation*, pp. 125–140, Springer, 2011.

[54] J. Baglama and L. Reichel, "Augmented gmres-type methods," *Numerical Linear Algebra with Applications*, vol. 14, no. 4, pp. 337–350, 2007.

[55] C. Jackson and P. C. Robinson, "A numerical study of various algorithms related to the preconditioned conjugate gradient method," *International journal for numerical methods in engineering*, vol. 21, no. 7, pp. 1315–1338, 1985.

[56] C. Linton, "The green's function for the two-dimensional helmholtz equation in periodic domains," *Journal of Engineering Mathematics*, vol. 33, pp. 377–401, 1998.

[57] I. Babuška, F. Ihlenburg, T. Strouboulis, and S. K. Gangaraj, "A posteriori error estimation for finite element solutions of helmholtz'equation. part i: The quality of local indicators and estimators," *International Journal for Numerical Methods in Engineering*, vol. 40, no. 18, pp. 3443–3462, 1997.