Substructured non-overlapping DDM vs. ORAS for Large Scale Helmholtz Problems with Multiple Sources

B. Martin[1], P. Jolivet[2] and C. Geuzaine[1]

[1] University of Liège, Belgium [2]CNRS, Sorbonne University, France

TSIMF Workshop on Advanced Solvers for Frequency-Domain Wave Problems and Applications

January 21 2025, Sanya, China

Context: inverse problems

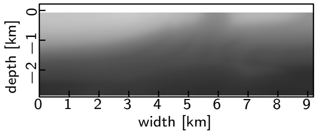
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**

Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**

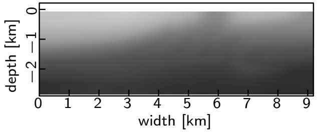
Choose an image



Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**

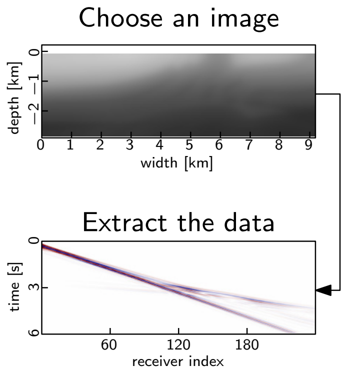
Choose an image



Simulate the propagation

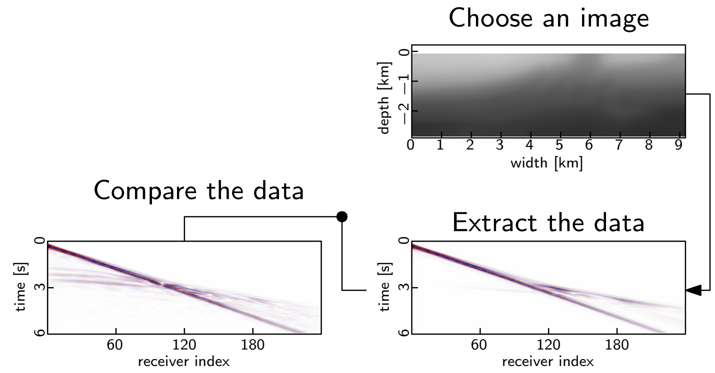
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



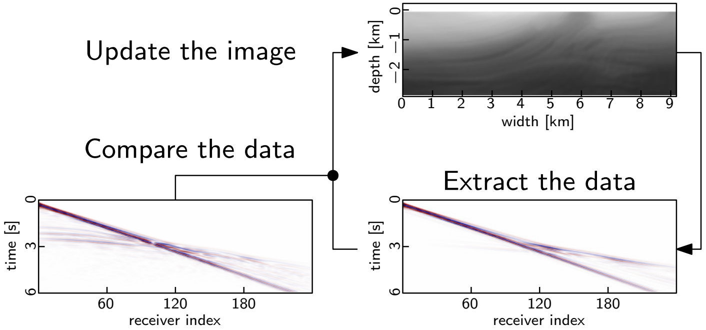
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



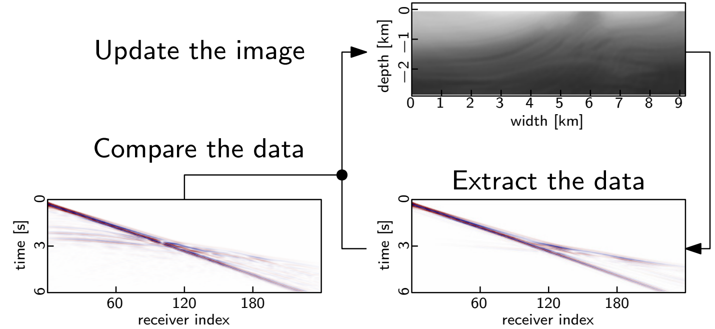
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



Full waveform inversion (FWI)

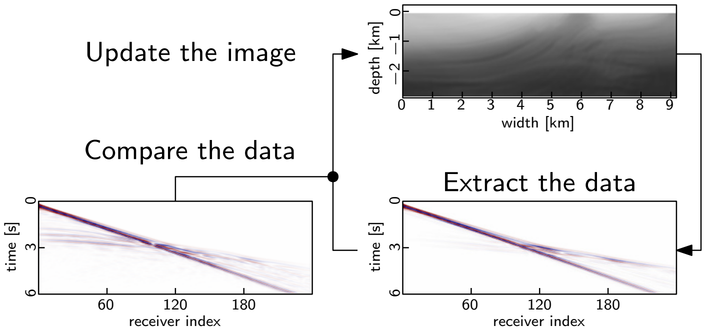
FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



Simulate the propagation

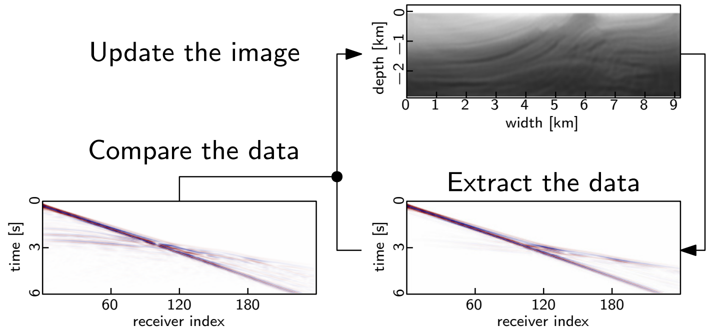
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



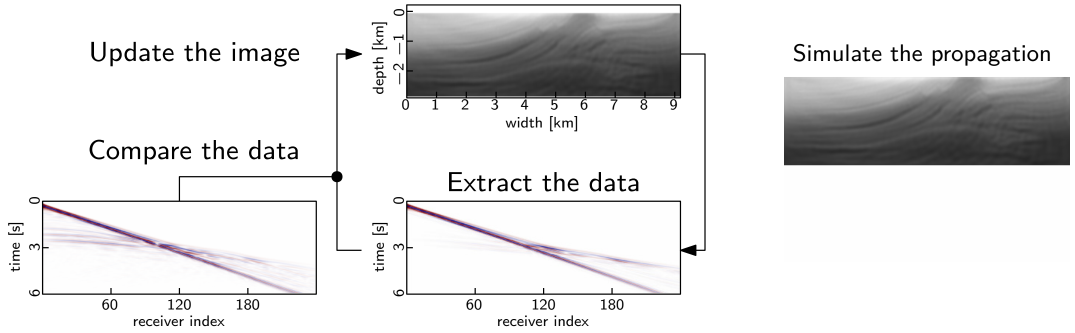
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



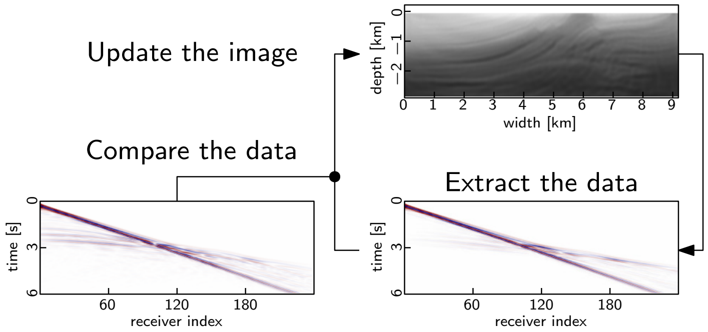
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



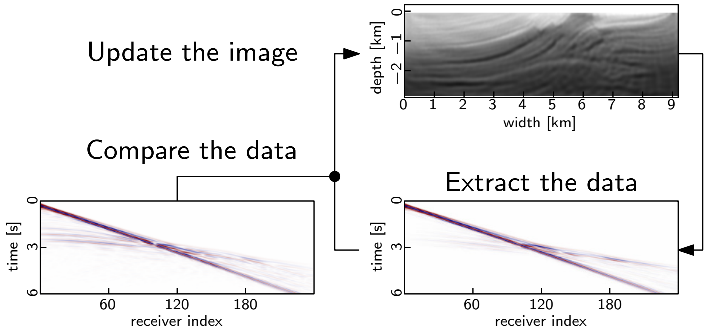
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



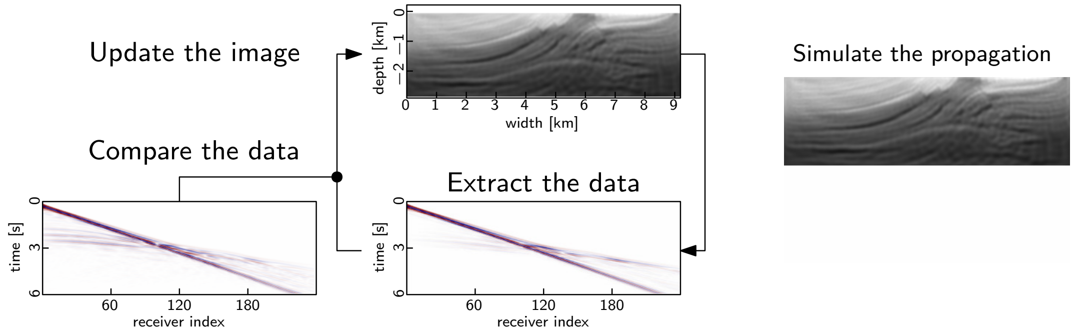
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



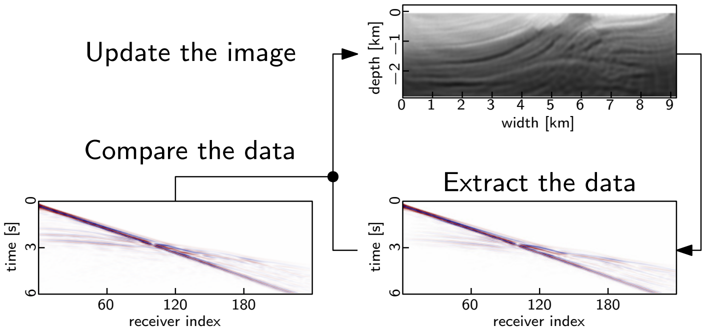
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



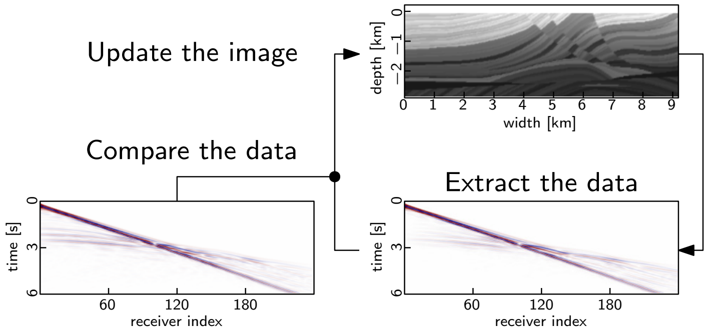
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



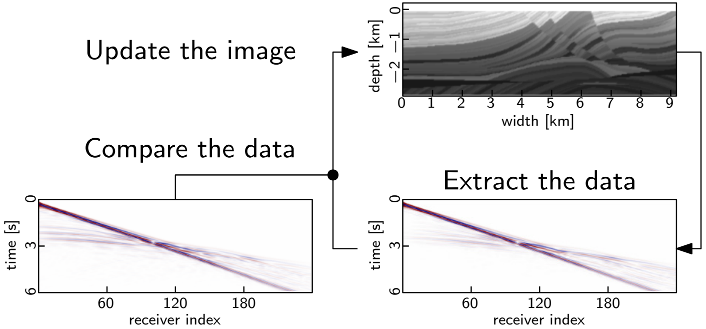
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**

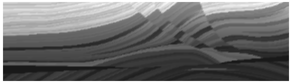


Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**

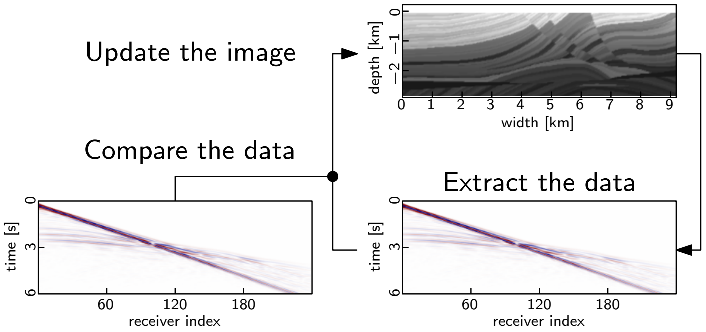


Simulate the propagation



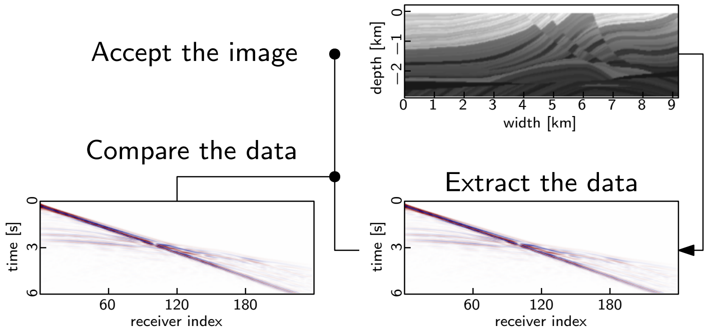
Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



Full waveform inversion (FWI)

FWI is an **imaging method** that reconstructs physical properties of a sample by **minimizing** the mismatch between measured wave scattering data on the boundary of the sample and data obtained by **full-wave simulations**



[Adriaens, Métivier & G., 2023]

This is FWI in the time domain: we will use it in the **frequency domain**, solving the Helmholtz equation instead of the wave equation

FWI in the frequency domain

Problem statement: For a model $m(x)$, a wavefield $u(x)$, data d, excitation f and a measurement operator R, find m that minimizes $J(m) = \|Ru(m) - d\|_2^2$ under constraint $A(m)u = f$

FWI in the frequency domain

Problem statement: For a model $m(x)$, a wavefield $u(x)$, data d, excitation f and a measurement operator R, find m that minimizes $J(m) = \|Ru(m) - d\|_2^2$ under constraint $A(m)u = f$

Setup for this talk:

- the model $m(x)$ is the local wave speed $c(x)$ in a domain Ω
- $A(m)$ is the Helmholtz operator, i.e. u satisfies the following Helmholtz problem

$$\begin{cases} -\Delta u - k^2 u = f \text{ in } \Omega \\ (\partial_{\boldsymbol{n}} u - \imath k u) = 0 \text{ on } \Gamma^{\infty} \end{cases}$$

  where $k = \frac{\omega}{c(x)}$, with ω the angular frequency
- the excitation f consists in (potentially many) sources located near the top of Ω

FWI in the frequency domain

Problem statement: For a model $m(x)$, a wavefield $u(x)$, data d, excitation f and a measurement operator R, find m that minimizes $J(m) = \|Ru(m) - d\|_2^2$ under constraint $A(m)u = f$

Setup for this talk:

- the model $m(x)$ is the local wave speed $c(x)$ in a domain Ω
- $A(m)$ is the Helmholtz operator, i.e. u satisfies the following Helmholtz problem

$$\begin{cases} -\Delta u - k^2 u = f \text{ in } \Omega \\ (\partial_{\boldsymbol{n}} u - \imath k u) = 0 \text{ on } \Gamma^\infty \end{cases}$$

  where $k = \frac{\omega}{c(x)}$, with ω the angular frequency
- the excitation f consists in (potentially many) sources located near the top of Ω

The minimization is carried out using a local, gradient-based optimization method (typically l-BFGS): computing $J(m)$ and $\nabla J(m)$ requires solving 2 Helmholtz problems, using an adjoint approach

FWI in the frequency domain

Problem statement: For a model $m(x)$, a wavefield $u(x)$, data d, excitation f and a measurement operator R, find m that minimizes $J(m) = \|Ru(m) - d\|_2^2$ under constraint $A(m)u = f$

Setup for this talk:

- the model $m(x)$ is the local wave speed $c(x)$ in a domain Ω
- $A(m)$ is the Helmholtz operator, i.e. u satisfies the following Helmholtz problem

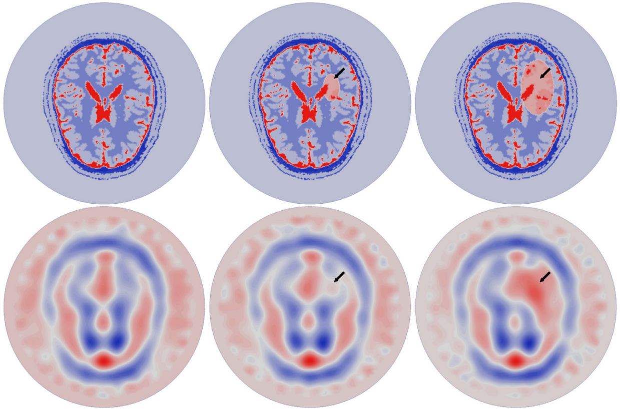
$$\begin{cases} -\Delta u - k^2 u = f \text{ in } \Omega \\ (\partial_{\boldsymbol{n}} u - \imath k u) = 0 \text{ on } \Gamma^\infty \end{cases}$$

  where $k = \frac{\omega}{c(x)}$, with ω the angular frequency
- the excitation f consists in (potentially many) sources located near the top of Ω

The minimization is carried out using a local, gradient-based optimization method (typically l-BFGS): computing $J(m)$ and $\nabla J(m)$ requires solving 2 Helmholtz problems, using an adjoint approach

Main cost: solve $A(m)u = f$ for different f and m

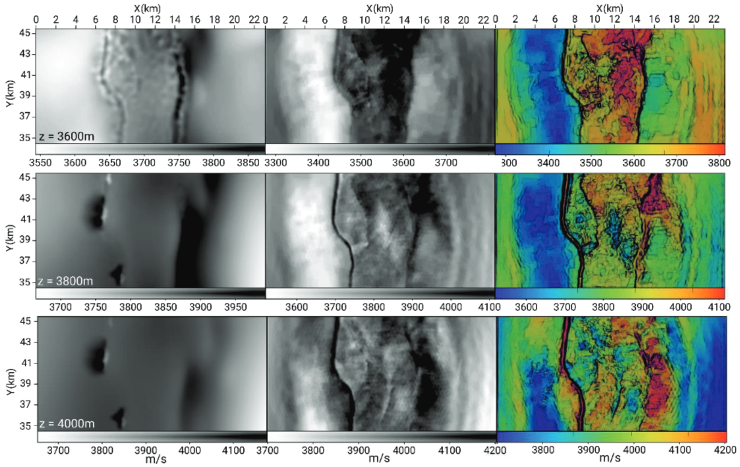
Frequency-domain FWI example in 2D



Imaginary part of permittivity in the brain

[Tournier et al., *Microwave tomographic imaging of cerebrovascular accidents by using high-performance computing*, Parallel Computing, 85, pp.88-97, 2019]

Frequency-domain FWI example in 3D



Slices of the *Gorgon* model before and after FWI

[Operto et al. *Is 3D frequency-domain FWI of full-azimuth/long-offset OBN data feasible? The Gorgon case study*. Leading Edge, 42 (3), pp.173-183, 2023]

State-of-the art

High-resolution FWI requires $\omega \gg$, leading to large sparse, complex and indefinite linear systems for which standard **iterative methods struggle**

State-of-the art

High-resolution FWI requires $\omega \gg$, leading to large sparse, complex and indefinite linear systems for which standard **iterative methods struggle**

Classical approach: perform sparse LU or LDL^T factorization (e.g. MUMPS)

State-of-the art

High-resolution FWI requires $\omega \gg$, leading to large sparse, complex and indefinite linear systems for which standard **iterative methods struggle**

Classical approach: perform sparse LU or LDL^T factorization (e.g. MUMPS)

- Factorization time $\mathcal{O}(N^2)$
- Memory hungry ($\mathcal{O}(N^{4/3})$)

State-of-the art

High-resolution FWI requires $\omega \gg$, leading to large sparse, complex and indefinite linear systems for which standard **iterative methods struggle**

Classical approach: perform sparse LU or LDL^T factorization (e.g. MUMPS)

- Factorization time $\mathcal{O}(N^2)$
- Memory hungry ($\mathcal{O}(N^{4/3})$)
- Efficient for a large number of right hand sides f
- Recent progress: Block Low Rank (BLR) compression, mixed precision

State-of-the art

High-resolution FWI requires $\omega \gg$, leading to large sparse, complex and indefinite linear systems for which standard **iterative methods struggle**

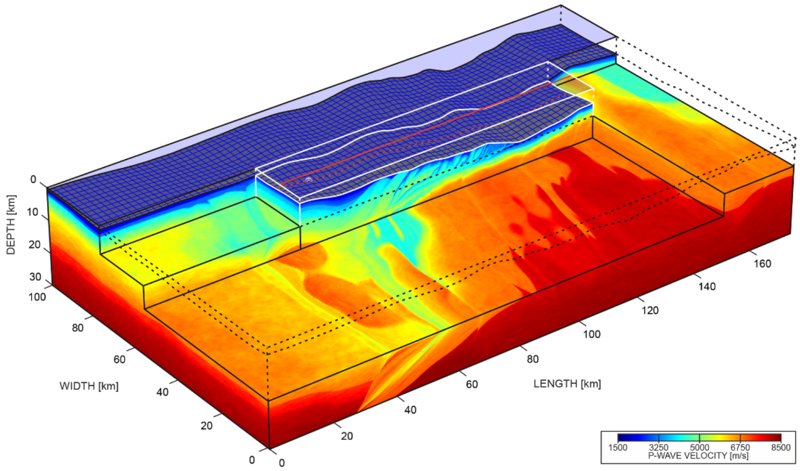
Classical approach: perform sparse LU or LDL^T factorization (e.g. MUMPS)

- Factorization time $\mathcal{O}(N^2)$
- Memory hungry ($\mathcal{O}(N^{4/3})$)
- Efficient for a large number of right hand sides f
- Recent progress: Block Low Rank (BLR) compression, mixed precision

If refinement proportional to frequency ω, we have $\mathcal{O}(\omega^3)$ unknowns:

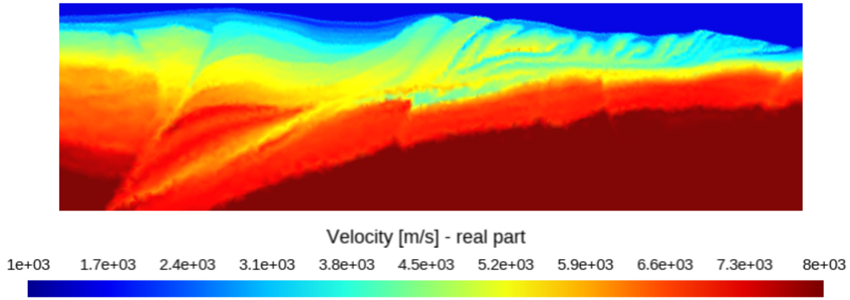
- Factorization time: $\mathcal{O}(\omega^6)$
- Extra cost per RHS: $\mathcal{O}(\omega^4)$
- Storage: $\mathcal{O}(\omega^4)$

Test case for this talk



[Górszczyk, A. and Operto, S.: GO_3D_OBS: the multi-parameter benchmark geomodel for seismic imaging method assessment and next-generation 3survey design (version 1.0), Geosci. Model Dev., 14, 17731799, https://doi.org/10.5194/gmd-14-1773-2021, 2021]

Test case for this talk

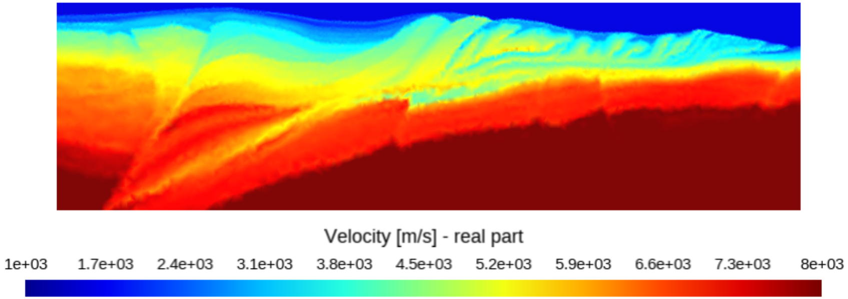


P-wave speed in one slice of the model

Wave speed varies by a factor close to 6 in the model: this requires

- mesh adaptation

Test case for this talk

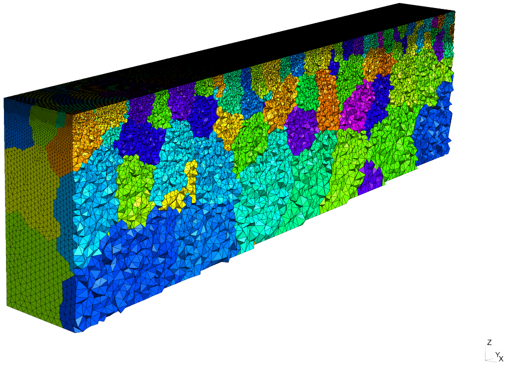


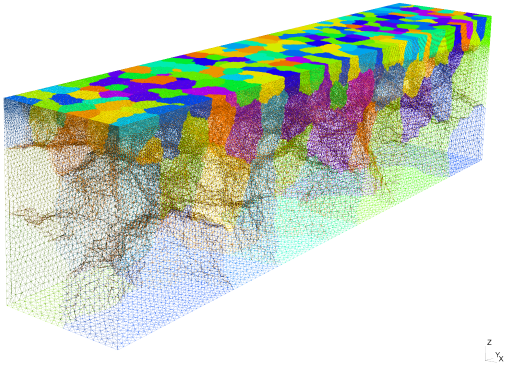
P-wave speed in one slice of the model

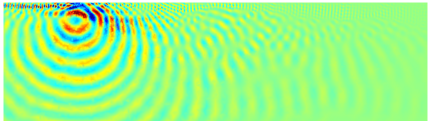
Wave speed varies by a factor close to 6 in the model: this requires

- mesh adaptation
- balanced mesh partitioning

Test case for this talk







Typical mesh (here with 256 partitions) and sample solution on a slice for a single source at 2Hz

Test case for this talk

What is the cost of a sparse direct solver for a problem with 10M Dofs and 68 sources?

Test case for this talk

What is the cost of a sparse direct solver for a problem with 10M Dofs and 68 sources?

Standard MUMPS on 8 HPC nodes ($8 \times 2 \times 64$ AMD Epyc Milan cores) requires

- 2TB of RAM
- 1250s (21 minutes) of compute time
  - $\rightarrow$ 140s symbolic factorization
  - $\rightarrow$ 960s numerical factorization
  - $\rightarrow$ 150s triangular solves

Test case for this talk

What is the cost of a sparse direct solver for a problem with 10M Dofs and 68 sources?

Standard MUMPS on 8 HPC nodes ($8 \times 2 \times 64$ AMD Epyc Milan cores) requires

- 2TB of RAM
- 1250s (21 minutes) of compute time
  - $\rightarrow$ 140s symbolic factorization
  - $\rightarrow$ 960s numerical factorization
  - $\rightarrow$ 150s triangular solves

MUMPS-BLR in mixed precision is about 20 % faster

Can we do better?

Domain Decomposition Methods for Helmholtz

Domain Decomposition Methods

Main idea of Domain Decomposition Methods (DDM): split the N unknowns in N_{dom} subdomains...

- Factorization is N_{dom}^2 times faster (but one needs to factorize N_{dom} times)
- We use less memory
- But we must iterate!

Domain Decomposition Methods

Main idea of Domain Decomposition Methods (DDM): split the N unknowns in N_{dom} subdomains...

- Factorization is N_{dom}^2 times faster (but one needs to factorize N_{dom} times)
- We use less memory
- But we must iterate!

We can use DDM to either

- build a **preconditioner** made of local solves for the original problem (e.g. ORAS)
- solve an **interface problem** to glue local solutions together (e.g. OSM)

Non-overlapping substructured optimized Schwarz DDM (OSM)

Partition Ω into non-overlapping subdomains Ω_i, $i = 1, \ldots, N_{\text{dom}}$, with interface $\Sigma_{i,j}$ between Ω_i and Ω_j; for every i, $\Gamma_i^\infty = \Gamma^\infty \cap \partial\Omega_i$

Non-overlapping substructured optimized Schwarz DDM (OSM)

Partition Ω into non-overlapping subdomains Ω_i, $i = 1, \dots, N_{\text{dom}}$, with interface $\Sigma_{i,j}$ between Ω_i and Ω_j; for every i, $\Gamma_i^\infty = \Gamma^\infty \cap \partial\Omega_i$

In a subdomain Ω_i with neighboring subdomain Ω_j, solve

Non-overlapping substructured optimized Schwarz DDM (OSM)

Partition Ω into non-overlapping subdomains Ω_i, $i = 1, \dots, N_{\text{dom}}$, with interface $\Sigma_{i,j}$ between Ω_i and Ω_j; for every i, $\Gamma_i^\infty = \Gamma^\infty \cap \partial\Omega_i$

In a subdomain Ω_i with neighboring subdomain Ω_j, solve

Local problem

$$\begin{cases} -\Delta u_i - k^2 u_i = f_i \text{ in } \Omega_i, & \text{(Helmholtz equation)} \\ (\partial_{\mathbf{n}_i} u_i - \imath k u_i) = 0, \text{ on } \Gamma_i^\infty & \text{(radiation condition)} \\ (\partial_{\mathbf{n}_i} u_i - \mathcal{S} u_i) = (\partial_{\mathbf{n}_i} u_j - \mathcal{S} u_j)\,, \text{ on } \Sigma_{ij} & \text{(transmission condition)} \end{cases}$$

with $k = \frac{\omega}{c(x)}$ the wave number and $\mathcal{S}$ a well-chosen interface operator (simplest: $\mathcal{S} = ik$)

Non-overlapping substructured optimized Schwarz DDM (OSM)

Partition Ω into non-overlapping subdomains Ω_i, $i = 1, \dots, N_{\text{dom}}$, with interface $\Sigma_{i,j}$ between Ω_i and Ω_j; for every i, $\Gamma_i^\infty = \Gamma^\infty \cap \partial\Omega_i$

In a subdomain Ω_i with neighboring subdomain Ω_j, solve

Local problem

$$\begin{cases} -\Delta u_i - k^2 u_i = f_i \text{ in } \Omega_i, & \text{(Helmholtz equation)} \\ (\partial_{\mathbf{n}_i} u_i - \imath k u_i) = 0, \text{ on } \Gamma_i^\infty & \text{(radiation condition)} \\ (\partial_{\mathbf{n}_i} u_i - \mathcal{S} u_i) = (\partial_{\mathbf{n}_i} u_j - \mathcal{S} u_j)\text{ , on } \Sigma_{ij} & \text{(transmission condition)} \end{cases}$$

with $k = \frac{\omega}{c(x)}$ the wave number and $\mathcal{S}$ a well-chosen interface operator (simplest: $\mathcal{S} = ik$)

The solution u_i in Ω_i depends on

- the sources in Ω_i
- the solution u_j in Ω_j through the transmission condition

Non-overlapping substructured optimized Schwarz DDM (OSM)

Pose $g_{ij} = \partial_{\mathbf{n_i}} u_i - \mathcal{S} u_i$ and introduce the corresponding interface unknown $g_{ji} = \partial_{\mathbf{n_j}} u_j - \mathcal{S} u_j$ for Ω_j

Since $\boldsymbol{n_i} = -\boldsymbol{n_j}$, we obtain the interface problem

$$\begin{aligned} g_{ij} &= \partial_{\mathbf{n_i}} u_j - \mathcal{S} u_j \\ &= -(\partial_{\mathbf{n_j}} u_j - \mathcal{S} u_j) - 2\mathcal{S} u_j \\ &= -g_{ji} - 2\mathcal{S} u_j \end{aligned}$$

Iterative method (Jacobi) to solve the global problem

For each subdomain i:

- Compute u_i^n from g_{ij}^n and f_i
- Update: $g_{ij}^{n+1} = -g_{ji}^n - 2\mathcal{S} u_j^n$.

Non-overlapping substructured optimized Schwarz DDM (OSM)

Split by linearity $u_i = v_i + \tilde{u}_i$ into its contribution from the physical sources f_i and the interface sources g_{ij}, and define the transmission operators

$$\mathcal{T}_{ij} g_{ij} := -g_{ij} - \mathcal{S}\tilde{u}_i$$
$$\mathcal{T}_{ji} g_{ji} := -g_{ji} - \mathcal{S}\tilde{u}_j$$

For two subdomains i, j we obtain the system

$$\underbrace{\begin{pmatrix} g_{ij} \\ g_{ji} \end{pmatrix}}_{g} = \underbrace{\begin{pmatrix} 0 & \mathcal{T}_{ji} \\ \mathcal{T}_{ij} & 0 \end{pmatrix}}_{\mathcal{A}} \underbrace{\begin{pmatrix} g_{ij} \\ g_{ji} \end{pmatrix}}_{g} \underbrace{-2\mathcal{S} \begin{pmatrix} v_j \\ v_i \end{pmatrix}}_{b} \qquad (1)$$

Non-overlapping substructured optimized Schwarz DDM (OSM)

Split by linearity $u_i = v_i + \tilde{u}_i$ into its contribution from the physical sources f_i and the interface sources g_{ij}, and define the transmission operators

$$\begin{aligned} \mathcal{T}_{ij} g_{ij} &:= -g_{ij} - \mathcal{S}\tilde{u}_i \\ \mathcal{T}_{ji} g_{ji} &:= -g_{ji} - \mathcal{S}\tilde{u}_j \end{aligned}$$

For two subdomains i, j we obtain the system

$$\underbrace{\begin{pmatrix} g_{ij} \\ g_{ji} \end{pmatrix}}_{g} = \underbrace{\begin{pmatrix} 0 & \mathcal{T}_{ji} \\ \mathcal{T}_{ij} & 0 \end{pmatrix}}_{\mathcal{A}} \underbrace{\begin{pmatrix} g_{ij} \\ g_{ji} \end{pmatrix}}_{g} \underbrace{-2\mathcal{S} \begin{pmatrix} v_j \\ v_i \end{pmatrix}}_{b} \tag{1}$$

The global update of the interface variables $g = (g_{ij}, g_{ji})^T$ thus takes the form of a linear system

$$(\mathcal{I} - \mathcal{A})g = b, \tag{2}$$

which can be solved with a matrix-free **Krylov solver** such as GMRES or GCR

Non-overlapping substructured optimized Schwarz DDM (OSM)

Properties of the interface problem:

- Significantly smaller number of unknowns than the volume problem
- Simplest transmission condition $\mathcal{S} = ik$ [Després 1991]

Non-overlapping substructured optimized Schwarz DDM (OSM)

Properties of the interface problem:

- Significantly smaller number of unknowns than the volume problem
- Simplest transmission condition $\mathcal{S} = ik$ [Després 1991]
- Clustering of the eigenvalues of $(\mathcal{I} - \mathcal{A})$ around 1 for "optimized" $\mathcal{S}$:
  - $\rightarrow$ $\mathcal{S} := (a + b\Delta_\Sigma)$ [Gander, Magoules & Nataf 2002], rational DtN approximations [Boubendir, Antoine & G. 2012], PMLs [Stolk 2013], [Vion & G. 2014], [Royer, G. Béchet & Modave 2022], non-local operators [Parolin 2020], ...

  Leads to fast convergence of the iterative Krylov solver
- One matrix-vector product involves solving each subproblem once

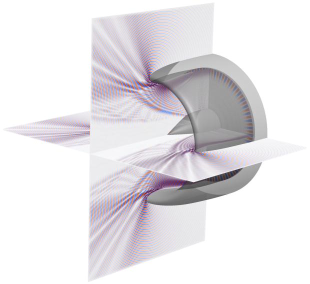
Solving the subproblems using a sparse direct solver is expected to be the most computationally expensive part

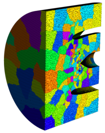
OSM in practice

- Open source implementation: Gmsh [G. & Remacle 2009], GmshFEM [Royer, Béchet & G. 2021] and GmshDDM

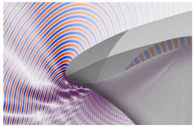
OSM in practice

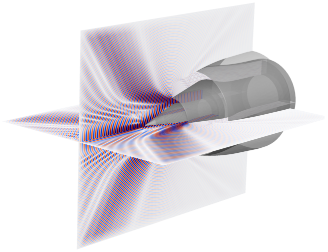
- Open source implementation: Gmsh [G. & Remacle 2009], GmshFEM [Royer, Béchet & G. 2021] and GmshDDM
- Extended to Maxwell [Dolean, Gander & Gerardo-Giorda 2009], [El Bouajaji, Thierry, Antoine, G. 2015], elastic waves [Mattesi, Darbas & G. 2020], convected Helmholtz [Marchner, Beriot, Antoine & G. 2024]

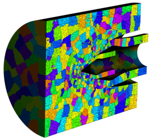




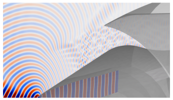
(b)







(b)



Cores (MPI×threads)	unknowns	nnz	peak memory	pre-pro	GMRES	It
1024×20	1.1B	167B	70Gb	24min	3h24min	1293

Cores (MPI×threads)	unknowns	nnz	peak memory	pre-pro	GMRES	It
4096×16	1.3B	96B	18.4Gb	1min	14min	555

(peak memory here is per MPI rank, i.e. per subdomain: see [Marchner, Beriot, Antoine & G. 2024])

512 node HPC cluster allows to typically resolve about $100 \times 100 \times 100$ wavelengths with high-order FEM

Gmsh FEM DDM

Optimized Restricted Additive Schwarz (ORAS)

Partition Ω in **overlapping** subdomains Ω_i, $i = 1, \ldots, N_{\text{dom}}$; denote R_i the restriction operator from Ω to Ω_i and D_i a partition of unity s.t. $\sum_i^{N_{\text{dom}}} R_i^T D_i R_i = I$

Optimized Restricted Additive Schwarz (ORAS)

Partition Ω in **overlapping** subdomains Ω_i, $i = 1, \dots, N_{\text{dom}}$; denote R_i the restriction operator from Ω to Ω_i and D_i a partition of unity s.t. $\sum_i^{N_{\text{dom}}} R_i^T D_i R_i = I$

RAS and ORAS preconditionners

Let A be the system matrix resulting from the discretization of the Helmholtz problem and $A_{\text{loc},i} = R_i A R_i^T$ the local matrix for subdomain i

The RAS preconditionner is defined as:

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N_{\text{dom}}} R_i^T D_i A_{\text{loc},i}^{-1} R_i$$

Optimized Restricted Additive Schwarz (ORAS)

Partition Ω in **overlapping** subdomains Ω_i, $i = 1, \ldots, N_{\text{dom}}$; denote R_i the restriction operator from Ω to Ω_i and D_i a partition of unity s.t. $\sum_i^{N_{\text{dom}}} R_i^T D_i R_i = I$

RAS and ORAS preconditionners

Let A be the system matrix resulting from the discretization of the Helmholtz problem and $A_{\text{loc},i} = R_i A R_i^T$ the local matrix for subdomain i

The RAS preconditionner is defined as:

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N_{\text{dom}}} R_i^T D_i A_{\text{loc},i}^{-1} R_i$$

Replacing $A_{\text{loc},i}$ with a local matrix $A_{\mathcal{S},i}$ obtained by assuming an impedance boundary condition on $\partial\Omega_i$ (i.e. the same as in OSM), we obtain the ORAS preconditionner:

$$M_{\text{ORAS}}^{-1} = \sum_{i=1}^{N_{\text{dom}}} R_i^T D_i A_{\mathcal{S},i}^{-1} R_i$$

Optimized Restricted Additive Schwarz (ORAS)

Partition Ω in **overlapping** subdomains Ω_i, $i = 1, \ldots, N_{\text{dom}}$; denote R_i the restriction operator from Ω to Ω_i and D_i a partition of unity s.t. $\sum_i^{N_{\text{dom}}} R_i^T D_i R_i = I$

RAS and ORAS preconditionners

Let A be the system matrix resulting from the discretization of the Helmholtz problem and $A_{\text{loc},i} = R_i A R_i^T$ the local matrix for subdomain i

The RAS preconditionner is defined as:

$$M_{\text{RAS}}^{-1} = \sum_{i=1}^{N_{\text{dom}}} R_i^T D_i A_{\text{loc},i}^{-1} R_i$$

Replacing $A_{\text{loc},i}$ with a local matrix $A_{\mathcal{S},i}$ obtained by assuming an impedance boundary condition on $\partial\Omega_i$ (i.e. the same as in OSM), we obtain the ORAS preconditionner:

$$M_{\text{ORAS}}^{-1} = \sum_{i=1}^{N_{\text{dom}}} R_i^T D_i A_{\mathcal{S},i}^{-1} R_i$$

Then apply a Krylov iterative solver (e.g. GMRES) to $M_{\text{ORAS}}^{-1} A u = M_{\text{ORAS}}^{-1} f$

OSM vs. ORAS comparison

Setup

- Adapted meshes generated by Gmsh, solutions obtained with GmshFEM+GmshDDM (OSM) and GmshFEM (ORAS)
- FEM order 3, 2nd order transmission conditions, ORAS with 1-element overlap

Setup

- Adapted meshes generated by Gmsh, solutions obtained with GmshFEM+GmshDDM (OSM) and GmshFEM (ORAS)
- FEM order 3, 2nd order transmission conditions, ORAS with 1-element overlap
- Varying frequency, leading to between 10M and 80M Dofs
- 68 point sources

Setup

- Adapted meshes generated by Gmsh, solutions obtained with GmshFEM+GmshDDM (OSM) and GmshFEM (ORAS)
- FEM order 3, 2nd order transmission conditions, ORAS with 1-element overlap
- Varying frequency, leading to between 10M and 80M Dofs
- 68 point sources
- Linear algebra via PETSc, linked to MUMPS and HPDDM [Jolivet, Roman & Zampini 2021]
- By default batch 32 RHS to be solved in parallel

Setup

- Adapted meshes generated by Gmsh, solutions obtained with GmshFEM+GmshDDM (OSM) and GmshFEM (ORAS)
- FEM order 3, 2nd order transmission conditions, ORAS with 1-element overlap
- Varying frequency, leading to between 10M and 80M Dofs
- 68 point sources
- Linear algebra via PETSc, linked to MUMPS and HPDDM [Jolivet, Roman & Zampini 2021]
- By default batch 32 RHS to be solved in parallel
- Tests performed on LUCIA Tier-1 cluster
  - $\rightarrow$ 2 64-core AMD Epyc Milan CPUs and 240 Gb of RAM per cluster node
  - $\rightarrow$ 1 process per subdomain, 2 threads per process

A priori comparison

A priori advantages of ORAS:

- Better convergence (thanks to overlap)
- Simpler to use (e.g. via PETSc)
- Tolerant to non-exact solutions of the subproblems

A priori comparison

A priori advantages of ORAS:

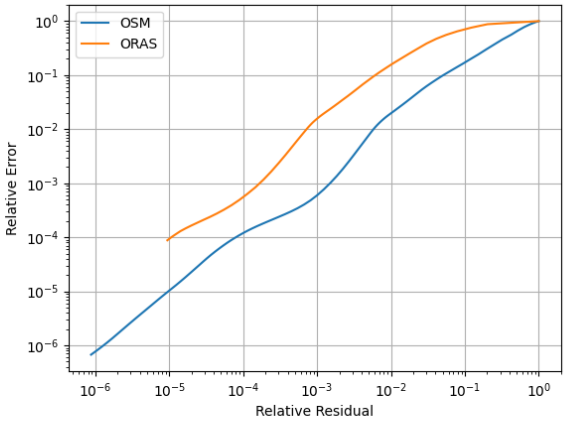
- Better convergence (thanks to overlap)
- Simpler to use (e.g. via PETSc)
- Tolerant to non-exact solutions of the subproblems

A priori advantages of OSM:

- Smaller subproblems
- Less costly Krylov iterations (thanks to interface unknowns)

Convergence criterion

OSM and ORAS minimize **different residuals**: the convergence criterion is adapted to produce fair comparisons (10^{-4} for OSM and 10^{-6} for ORAS)



Relative L^2 error vs. GMRES residual on 10M Dofs case

Partitioning

N_{dom}	ORAS Dofs/dom	OSM Dofs/dom	OSM Dofs(Ω)/Dofs(Σ)
128	123k	92k	8.2
256	69k	46k	6.0
384	50k	31k	5.0
512	39k	24k	4.4

10M Dofs case: average number of Dofs per subdomain

Partitioning

N_{dom}	ORAS Dofs/dom	OSM Dofs/dom	OSM Dofs(Ω)/Dofs(Σ)
128	123k	92k	8.2
256	69k	46k	6.0
384	50k	31k	5.0
512	39k	24k	4.4

10M Dofs case: average number of Dofs per subdomain

For larger problems: increase number of subdomains to keep similar averages per subdomain

10M Dofs: OSM with GMRES

N_{dom}	128	256	384	512
Iterations	50	65	78	87
Setup time	14s	5s	3s	2s
Local solves	65s	37s	31s	23s
Gram-Schmidt	4s	3.5s	3.5s	3.5s
Local RHS assembly	19s	16s	15s	14s
Total wall time	86s	57s	49s	42s
RAM upper bound	170 GB	219 GB	269 GB	337 GB

(Batch size: 32)

Back to comparison with direct sparse solver (MUMPS 5.7.3): 1250s / 2TB RAM

10M Dofs: ORAS with GMRES

N_{dom}	128	256	384	512
Iterations	68	67	70	74
Setup time	22s	6s	4s	2s
Local solves	150s	80s	56s	50s
Gram-Schmidt	64s	33s	27s	50s
Sparse MVP	226s	117s	82s	69s
Total wall time	417s	211s	150s	122s
RAM upper bound	329 GB	376 GB	376 GB	451 GB

(Batch size: 32)

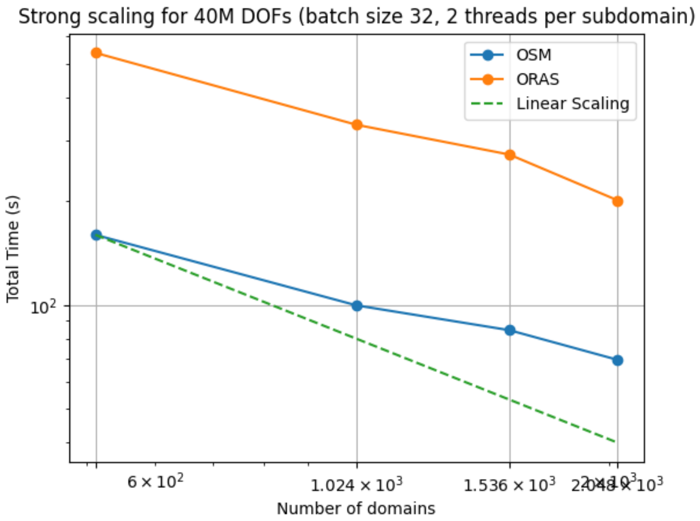
Back to comparison with direct sparse solver (MUMPS 5.7.3): 1250s / 2TB RAM

10M Dofs: early observations

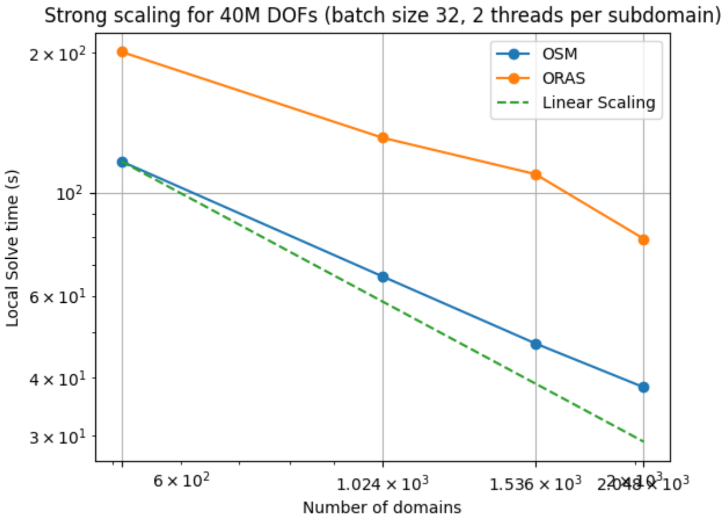
OSM outperforms ORAS here:

- smaller cost of GMRES
- smaller subdomains (no overlap)
- comparable iteration count
- replacing residual update in ORAS (global SPMV) by assembly of local interface terms (interface local SPMV) in OSM
- ORAS converges faster than OSM if there are many subdomains
- Both OSM and ORAS clearly outperform a sparse direct solver, even with 68 RHS

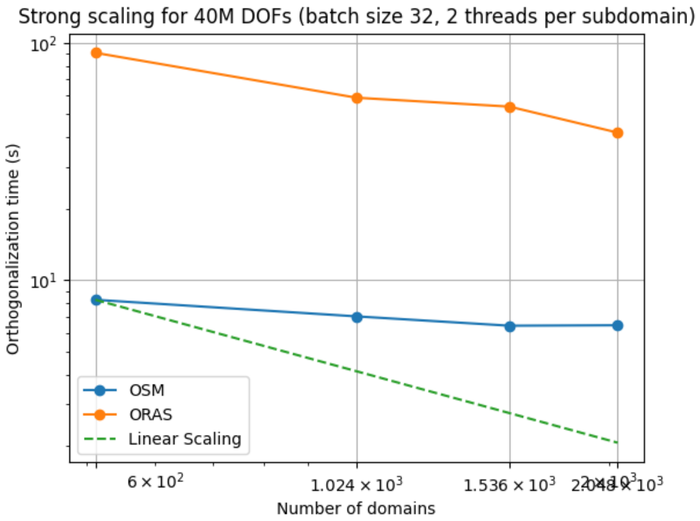
40M Dofs: strong scaling (total)



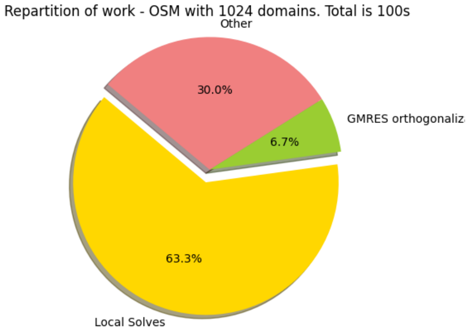
40M Dofs: strong scaling (local solves)

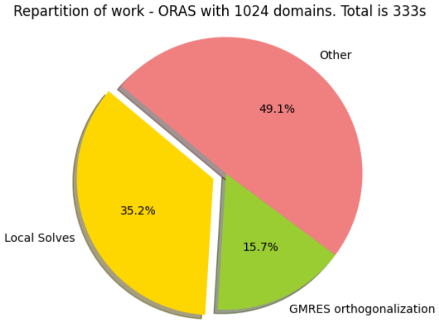


40M Dofs: strong scaling (orthogonalizations)



40M Dofs: details on the solving phase for $N_{\text{dom}} = 1024$

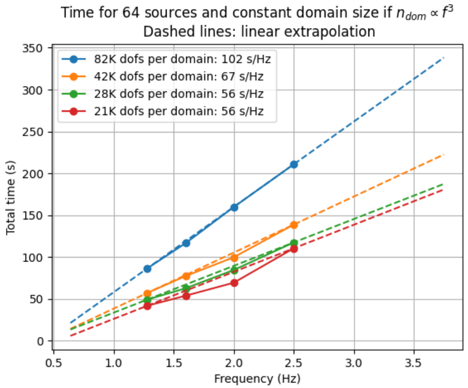


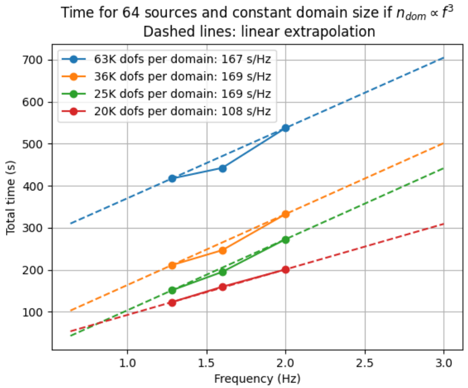


OSM spends **a larger fraction of time** on **useful work** (local solves), especially when subdomains are large (substructuring effect)

Combined with cheaper local solves (no overlap), this makes OSM very economical

Weak scaling





Weak scaling of OSM (left) vs. ORAS (right)

40M Dofs: influence of the batch size

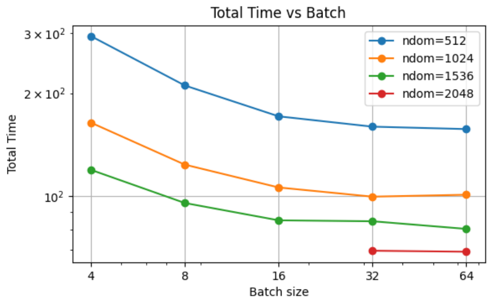
- Solving many RHS in parallel yields better arithmetic intensity...

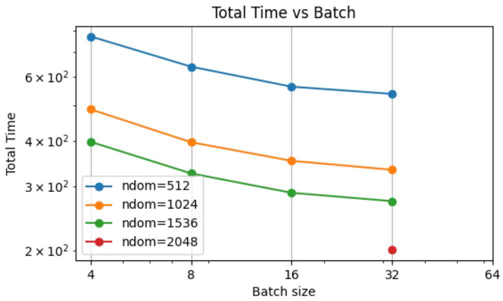
40M Dofs: influence of the batch size

- Solving many RHS in parallel yields better arithmetic intensity...
- At the cost of more memory!
  - $\rightarrow$ For ORAS the cost is about 16 GB per source for the 40M Dofs case: batching all the sources would consume more than half the allocated amount!
  - $\rightarrow$ OSM mitigates this (4 to 8 times less memory) thanks to smaller size of interface problem

40M Dofs: influence of the batch size

- Solving many RHS in parallel yields better arithmetic intensity...
- At the cost of more memory!
  - → For ORAS the cost is about 16 GB per source for the 40M Dofs case: batching all the sources would consume more than half the allocated amount!
  - → OSM mitigates this (4 to 8 times less memory) thanks to smaller size of interface problem





Impact of the batch size for OSM (left) and ORAS (right)

On the use of Block GMRES (BGMRES)

For a square matrix A and a vector b the m-th Krylov subspace is defined as

$$K^m(A, b) = \text{span}\{b, Ab, A^2b, \dots, A^{m-1}b\}$$

GMRES and BGMRES

GMRES provides at iteration m the element $x_m \in K^m(A, b)$ that minimizes $||b - Ax_m||_2$

On the use of Block GMRES (BGMRES)

For a square matrix A and a vector b the m-th Krylov subspace is defined as

$$K^m(A,b) = \text{span}\{b, Ab, A^2b, \dots, A^{m-1}b\}$$

GMRES and BGMRES

GMRES provides at iteration m the element $x_m \in K^m(A,b)$ that minimizes $||b - Ax_m||_2$

For several vectors $b_1, b_2, \dots, b_p$, BGMRES provides the best approximation of $A^{-1}b_l,\ l = 1, 2, \dots, p$ in the sum of the p subspaces $K^m(A,b_1), K^m(A,b_2), \dots, K^m(A,b_p)$

On the use of Block GMRES (BGMRES)

For a square matrix A and a vector b the m-th Krylov subspace is defined as

$$K^m(A,b) = \text{span}\{b, Ab, A^2b, \dots, A^{m-1}b\}$$

GMRES and BGMRES

GMRES provides at iteration m the element $x_m \in K^m(A,b)$ that minimizes $||b - Ax_m||_2$

For several vectors $b_1, b_2, \dots, b_p$, BGMRES provides the best approximation of $A^{-1}b_l,\ l = 1, 2, \dots, p$ in the sum of the p subspaces $K^m(A,b_1), K^m(A,b_2), \dots, K^m(A,b_p)$

Using BGMRES instead of GMRES with large batches should lead to

- less iterations
- at the cost of more orthogonalizations

The latter is significantly cheaper without overlap

Is it worth it?

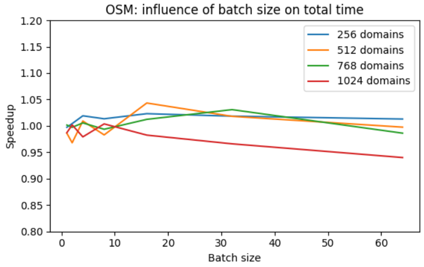
40M Dofs: GMRES vs. BGMRES

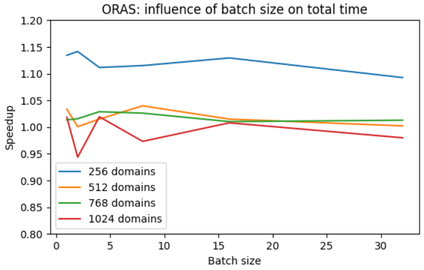
N_{dom}	GMRES - 32	BGMRES - 32	BGMRES - 64
512	95	87	84
1024	117	110	105
1536	135	128	128
2048	150	141	138

Number of iterations using OSM

N_{dom}	GMRES - 32	BGMRES - 32
512	90	82
1024	105	100
1536	133	115
2048	129	123

Number of iterations using ORAS





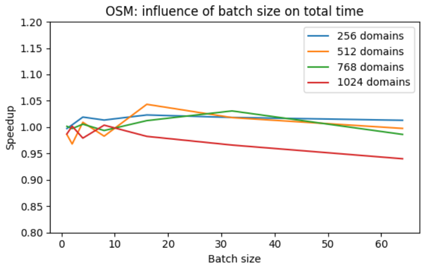
40M Dofs: GMRES vs. BGMRES

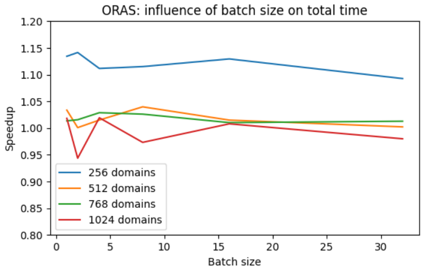
N_{dom}	GMRES - 32	BGMRES - 32	BGMRES - 64
512	95	87	84
1024	117	110	105
1536	135	128	128
2048	150	141	138

Number of iterations using OSM

N_{dom}	GMRES - 32	BGMRES - 32
512	90	82
1024	105	100
1536	133	115
2048	129	123

Number of iterations using ORAS





BGMRES provides a moderate speedup in OSM but is usually slower in ORAS: this probably depends a lot on the geometry and on the location of the sources

Conclusions

We evaluated overlapping and non-overlapping DDM for solving the Helmholtz equation in 3D with multiple sources in realistic conditions

- Both DDMs are much less expensive than a sparse direct solver for the considered number of sources
- Substructured non-overlapping DDM is significantly more efficient than ORAS: orthogonalisations, MVP and smaller local solves
- Parallelizing the sources (*batch size*) is efficient, but only affordable without overlap
- BGMRES has limited impact, but might be worth it with OSM

Current and future work

- More detailed study in an incoming paper
- Integration in our FWI code
  - $\rightarrow$ In particular interactions with the optimization algorithms

Current and future work

- More detailed study in an incoming paper
- Integration in our FWI code
  - $\rightarrow$ In particular interactions with the optimization algorithms
- Two-level methods (geometrical or spectral coarse grids)
  - $\rightarrow$ **One of my goals for this workshop**: discuss with you about how they could be efficiently applied to OSM?

Current and future work

- More detailed study in an incoming paper
- Integration in our FWI code
  - $\rightarrow$ In particular interactions with the optimization algorithms
- Two-level methods (geometrical or spectral coarse grids)
  - $\rightarrow$ **One of my goals for this workshop**: discuss with you about how they could be efficiently applied to OSM?

Thanks!

✉ cgeuzaine@uliege.be