# Parallel scalable solvers for Helmholtz problems

Jinqiang Chen, Vandana Dwarka, Cornelis Vuik*

Technische Universiteit Delft

August 24, 2023

# Aim and Impact

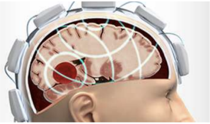
- Joint work with Ph.D. candidate Jinqiang Chen, Dr. Dwarka
- Contribute to broad research on parallel scalable Helmholtz solvers
- This presentation: matrix-free parallelization
  - Complex shift Laplace Preconditioner (CSLP)
  - Deflation methods
  - Parallel performance

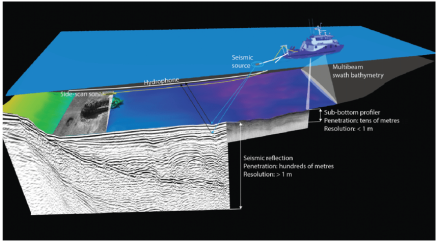
# Introduction - the Helmholtz Problem

The Helmholtz equation (describing time-harmonic waves) + BCs

$$-\Delta u(\mathbf{x}) - k(\mathbf{x})^2 u(\mathbf{x}) = g(\mathbf{x}), \text{ on } \Omega \subseteq \mathbb{R}^n$$

- $k(\mathbf{x})$ is the wavenumber, $k(\mathbf{x}) = (2\pi f)/c(\mathbf{x})$, where $f$ is the frequency and $c$ is the acoustic velocity of the media
- Applications in **seismic exploration**, medical imaging, antenna synthesis, etc.



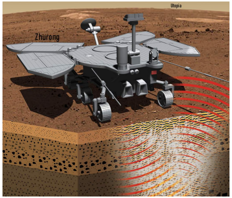




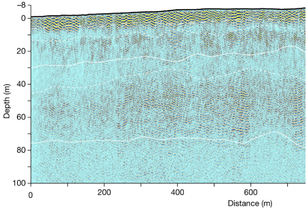
Larisa, High-performance implementation of Helmholtz equation with absorbing boundary conditions. http://www.math.chalmers.se/~larisa/www/MasterProjects/HelmholtzABSbc.pdf

M. Jakobsson, et al (2016). Mapping submarine glacial landforms using acoustic methods. Geological Society.

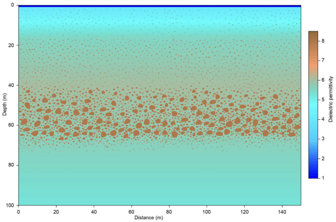
# Introduction




(a) Zhurong rover




(b) The radar imaging profile



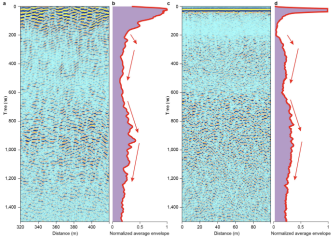

(c) Numerical model

Solve the

**Helmholtz equation**

⟵⟶

Adjust model




(d) Observed data vs. simulation

Li, C., Zheng, Y., Wang, X. et al. (2022) Layered subsurface in Utopia Basin of Mars revealed by Zhurong rover radar. Nature.

# Introduction - Challenges

- Linear system from finite-difference discretization

$$Au = b$$

  - $A$ is real, sparse, symmetric, normal, and **indefinite**; **non-Hermitian** with Sommerfeld BCs
- Direct solver or iterative solver
- **Accuracy and pollution error** ($k^3h^2 < 1$): finer grid (3D) $\Rightarrow$ larger linear system
  - Memory-efficient methods; Parallel computing
- **Negative & positive eigenvalues**: larger wavenumber $\Rightarrow$ more iterations
  - Preconditioner: Complex Shifted Laplace Preconditioner (CSLP)
  - (Higher-order) Deflation
- **Parallelism**

**Aim**

A wavenumber-independent-convergence and scalable parallel solver

# Introduction - Parallel computing

- Convergence metric:
  - Krylov-based solvers, GMRES-type: the number of iterations (#iter); IDR(s): the number of matrix-vector multiplications (#Matvec)
- Scalability:
  - Strong scaling: the number of processors is increased while the problem size remains constant
  - Weak scaling: the problem size increases along with the number of tasks, so the computation per task remains constant
  - Wall-clock time: $t_w$; number of processors: $np$
  - Speedup: $S_p = \frac{t_{w,r}}{t_{w,p}}$, $E_P = \frac{S_p}{np/np_r} = \frac{t_{w,r} \cdot np_r}{t_{w,p} \cdot np}$

# Introduction - Numerical Models

- Model problems on a rectangular domain $\Omega$ with boundary $\Gamma = \partial\Omega$

$$-\Delta u(\mathbf{x}) - k(\mathbf{x})^2 u(\mathbf{x}) = \delta\left(\mathbf{x} - \mathbf{x}_0\right), \text{ on } \Omega$$

$$u(\mathbf{x}) = 0 \text{ OR } \frac{\partial u(\mathbf{x})}{\partial \vec{n}} - \mathrm{i}k(\mathbf{x})u(\mathbf{x}) = 0, \text{ on } \Gamma$$

  - Constant wavenumber: $k(\mathbf{x}) = k$
  - Non-constant wavenumber: Wedge, Marmousi problem, 3D SEG/EAGE Salt Model, etc.
- Finite-difference discretization on a uniform grid with grid size $h$. (2D example)
  - Laplace operator: $-\Delta_h \mathbf{u} \approx \dfrac{-u_{i,j-1} - u_{i-1,j} + 4u_{i,j} - u_{i+1,j} - u_{i,j+1}}{h^2}$
  - Sommerfeld BCs: a ghost point

$$\frac{\partial u}{\partial \vec{n}}(0, y_j) - \mathrm{i}k(0, y_j)u(0, y_j) \approx \frac{u_{0,j} - u_{2,j}}{2h} - \mathrm{i}k_{1,j}u_{1,j} = 0 \Rightarrow u_{0,j} = u_{2,j} + 2h\mathrm{i}k_{1,j}u_{1,j}$$

- Preconditioned Krylov subspace solver: GMRES for complex system
- Preconditioner: Geometric multigrid method

# Introduction - Numerical Models

- **Stencil notation**
- Laplace operator:

$$[-\Delta_h] = \frac{1}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

- "Wavenumber operator":

$$[\mathcal{I}_h \mathbf{k}^2] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & k_{i,j}^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \overset{const}{=} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} k^2$$

- $A\mathbf{u} = \mathbf{b}$:

$$[A_h] = [-\Delta_h] - [\mathcal{I}_h \mathbf{k}^2]$$

# Framework - Matrix-free operations

- Perform computations with a matrix without explicitly forming or storing the matrix
  $\Rightarrow$ Reduce memory requirements

## Matrix-vector multiplication

If a matrix can be represented by a so-called stencil notation

$$[A] = \begin{bmatrix} a_{-1,1} & a_{0,1} & a_{1,1} \\ a_{-1,0} & a_{0,0} & a_{1,0} \\ a_{-1,-1} & a_{0,-1} & a_{1,-1} \end{bmatrix},$$

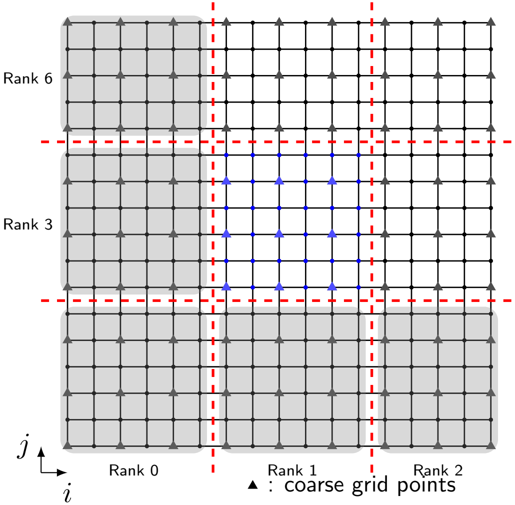
Then $\mathbf{v} = A\mathbf{u}$ can be computed by

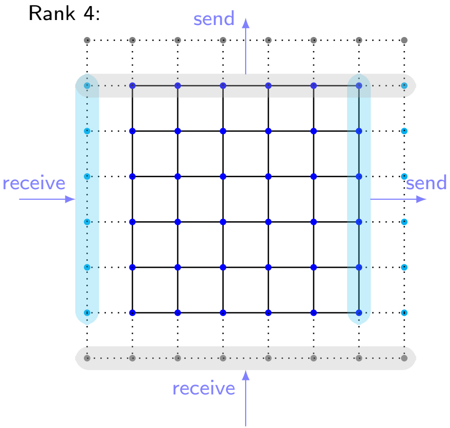
$$v_{i,j} = \sum_{p=-1}^{1} \sum_{q=-1}^{1} a_{p,q} u_{i+p,j+q}$$

with the help of a ghost point on the physical boundary and one overlapping grid point.

# Framework - Distributed data structure

- Vector $\mathbf{u} \Leftarrow$ 2D array: u(1:Nx,1:Ny) $\Leftarrow$ each sub-domain: u(1−LAP:nx+LAP,1−LAP:ny+LAP)
- Operations (*e.g.* matvec, dot-product, vector update) perform on each u(1:nx,1:ny) simultaneously



# CSLP

- Speed up convergence of Krylov subspace methods by Preconditioning
- Solve $M^{-1}Au = M^{-1}b$
- Complex Shifted Laplace Preconditioner (CSLP)

$$M_h = -\Delta_h - (\beta_1 - \beta_2 \mathrm{i})\,\mathcal{I}_h \mathbf{k}^2,\ (\beta_1, \beta_2) \in [0,1]\,, \quad e.g.\ \beta_1 = 1, \beta_2 = 0.5$$

  - ☑ Stencil notation
- Solve $Mx = u$ by multigrid method (V-cycle) $\Rightarrow x \approx M^{-1}u$
  - **Vertex-centered** coarsening based on the **global** grid
  - Damped Jacobi smoother (easy to parallelize)
  - Full-weight restriction $I_h^{2h}$ & linear interpolation $I_{2h}^h$

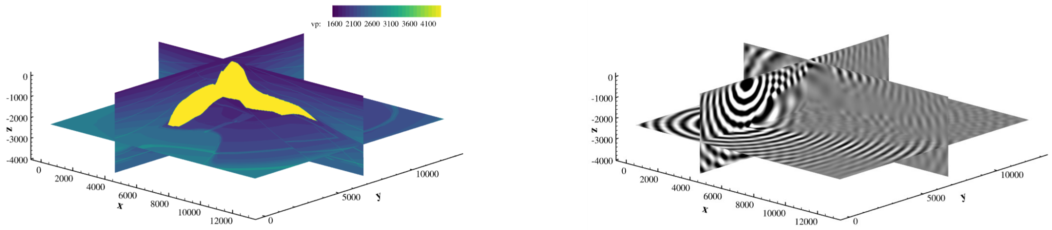
$$[I_h^{2h}] = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_h^{2h}, \quad [I_{2h}^h] = \frac{1}{4}\left]\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}\right[_{2h}^{h}$$

  - Coarse-grid operator obtained by **re-discretization**
    - ☑ Stencil notation: $[M_{2h}]$ similar to $[M_h]$

# Parallel CLSP-preconditioned Krylov solver

## 3D SEG/EAGE Salt Model

- Real large-size domain $12\,800\,\mathrm{m} \times 12\,800\,\mathrm{m} \times 3840\,\mathrm{m}$
- High heterogeneity: the velocity varies from $1500\,\mathrm{m\,s^{-1}}$ to $4482\,\mathrm{m\,s^{-1}}$
- Grid size $641 \times 641 \times 193$




(a) Velocity distribution

(b) Pattern of wave field at $f = 5\,\mathrm{Hz}$

Figure: 3D SEG/EAGE Salt Model

# Parallel CLSP-preconditioned Krylov solver

- Parallel CSLP-preconditioned IDR(4) for 3D SEG/EAGE Salt Model with grid size $641 \times 641 \times 193$ at $f = 5\,\text{Hz}$

Table: Performance on DelftBlue [1]

| npx × npy × npz | Nodes | #Matvec | t(s) | Sp | Ep |
|---|---|---|---|---|---|
| 6×4×2 | 1 | 413 | 897.25 | | |
| 6×8×2 | 2 | 423 | 510.56 | 1.76 | 0.88 |
| 6×8×4 | 4 | 423 | 298.86 | 3.00 | 0.75 |
| 9×8×4 | 6 | 404 | 203.31 | 4.41 | 0.74 |

Table: Performance on Magic Cube [2]

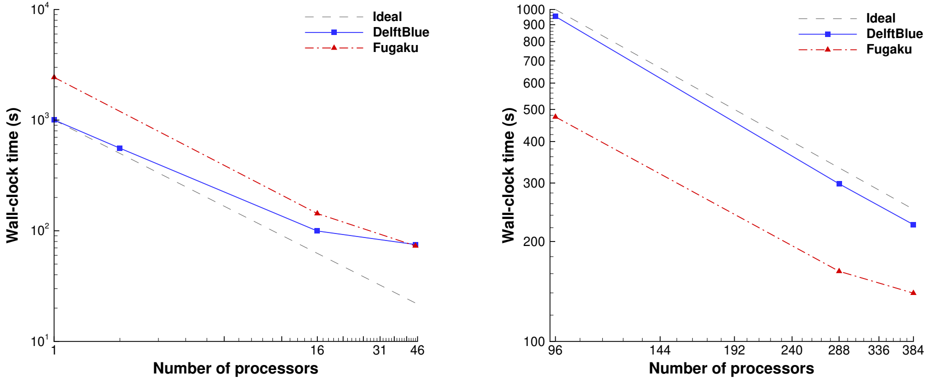
| npx × npy × npz | Nodes | #Matvec | t(s) | Sp | Ep |
|---|---|---|---|---|---|
| 4 × 4 × 2 | 1 | 405 | 505.14 | | |
| 4 × 4 × 4 | 2 | 418 | 287.60 | 1.76 | 0.88 |
| 8 × 8 × 2 | 4 | 390 | 155.64 | 3.25 | 0.81 |

- Good parallel performance
- Effective on different platforms

[1] DHPC, DelftBlue Supercomputer (Phase 1) `https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1`

[2] Supercomputer Magic Cube III: `https://www.ssc.net.cn/en/resource-hardware.html`

# Parallel CLSP-preconditioned Krylov solver



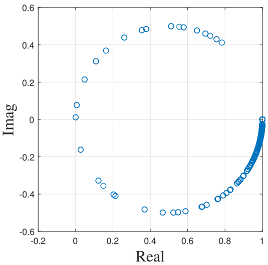

(a) Single compute node (b) Multiple compute nodes

Figure: Strong scaling[1]. 3D model problem with $\sim 100$ million unknowns, #Matvec $\simeq 850$

[1] Supercomputer Fugaku: `https://www.r-ccs.riken.jp/en/fugaku/`. Riken International HPC Summer School 2022 is acknowledged

# CSLP - Cons

- Increasing $k \Rightarrow$ eigenvalues move fast towards origin
- Too many iterations for high frequency
- Project unwanted eigenvalues to zero $\Rightarrow$ Deflation



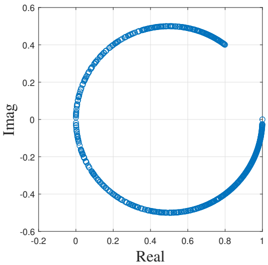




Figure: $\sigma\left(M_{(1,0.5)}^{-1}A\right)$ for $k=20$ (left) and $k=80$ (right)

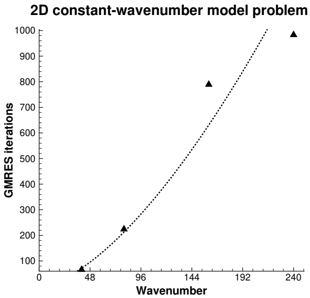



Figure: #Iter increases with $k$

# Deflation - introduction

- Project unwanted eigenvalues to zero $\Rightarrow$ Deflation
- Deflation preconditioning: solve $PA\hat{u} = Pb$

$$P = I - AQ, \quad \text{where } Q = ZE^{-1}Z^T, \quad E = Z^T AZ$$

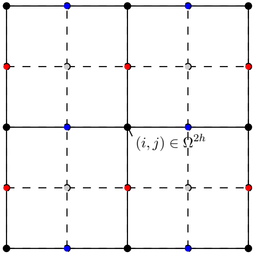
$$A \in \mathbb{R}^{n\times n}, Z \in \mathbb{R}^{m\times n}$$

- Columns of $Z$ span deflation subspace
- Ideally $Z$ contains eigenvectors
- In practice approximations: inter-grid vectors from multigrid
- Adapted Deflation Variant 1 (A-DEF1): $P_{A-DEF1} = M^{-1}_{(\beta_1,\beta_2)}P + Q$
  - Combined with the standard preconditioner CSLP
- Use CSLP-preconditioned GMRES to solve the coarse grid problem (obtain $E^{-1}$) approximately
- Linear approximation basis deflation vectors $\rightarrow$ **higher-order** deflation vectors
  - wavenumber-independent convergence

# Higher-order deflation vectors

- 2D: the higher-order interpolation & restriction has $5 \times 5$ stencil
  - **Two overlapping** grid points are needed

$$[Z] = \frac{1}{64} \left] \begin{array}{ccccc} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{array} \right[_{2h}^{h}, \quad [Z^T] = \frac{1}{64} \left[ \begin{array}{ccccc} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{array} \right]_{h}^{2h}$$




• • • • : fine grid points $\in \Omega^h$
• : coarse grid points $\in \Omega^{2h}$

Figure: The allocation map of interpolation operator

# Matrix-free two-level deflation

$$P = I - AQ, \quad \text{where } Q = ZE^{-1}Z^T, \quad E = Z^TAZ$$

- With matrix constructed, $E = Z^TAZ$, so-called Galerkin Coarsening

## Matrix-free coarse grid operation $y = Ex$?

- Straightforward Galerkin Coarsening operator;

$$x_1 = Zx,\ x_2 = A_h x_1,\ y = Z^T x_2 \Rightarrow y = Ex$$

  - unacceptable computational cost for consideration of multilevel method

- Re-discretization:
  - **ReD-$\mathcal{O}$2**: The same as the fine grid
  - **ReD-$\mathcal{O}$4**: Fourth-order re-discretization of the Laplace operator

$$[E] = \frac{1}{12 \cdot (2h)^2} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -16 & 0 & 0 \\ 1 & -16 & 60 & -16 & 1 \\ 0 & 0 & -16 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} - \mathcal{I}_{2h}\mathbf{k}_{2h}^2$$

# Matrix-free two-level deflation

**ReD-Glk**: Re-discretized scheme (stencil) from the result of Galerkin coarsening

$$[-\Delta_{2h}] = \frac{1}{(2h)^2} \cdot \frac{1}{256} \begin{bmatrix} -3 & -44 & -98 & -44 & -3 \\ -44 & -112 & 56 & -112 & -44 \\ -98 & 56 & 980 & 56 & -98 \\ -44 & -112 & 56 & -112 & -44 \\ -3 & -44 & -98 & -44 & -3 \end{bmatrix}$$

$$\Rightarrow -\Delta_{2h} u_{2h} = -4\frac{\partial^2 u}{\partial x^2} - 4\frac{\partial^2 u}{\partial y^2} - \left(\frac{13}{48}\frac{\partial^4 u}{\partial x^4} + \frac{1}{2}\frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{13}{48}\frac{\partial^4 u}{\partial y^4}\right)(\mathbf{2h})^{\mathbf{2}} + \mathcal{O}(h^4)$$

$$[\mathcal{I}_{2h}\mathbf{k}_{2h}^2] = \frac{1}{64^2} \begin{bmatrix} 1 & 28 & 70 & 28 & 1 \\ 28 & 784 & 1960 & 784 & 28 \\ 70 & 1960 & 4900 & 1960 & 70 \\ 28 & 784 & 1960 & 784 & 28 \\ 1 & 28 & 70 & 28 & 1 \end{bmatrix} \mathbf{k}_{2h}^2$$

$$\Rightarrow [E] = [-\Delta_{2h}] - [\mathcal{I}_{2h}\mathbf{k}_{2h}^2]$$

? Boundary conditions - ReD-$\mathcal{O}$2 on the boundary grid points

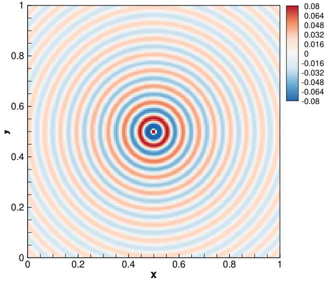
# Convergence - Constant wavenumber

Table: The number of iterations required by using higher-order A-DEF1 preconditioned GMRES. In parentheses is the number of CSLP-GMRES-iterations to solve the coarse grid problem once.

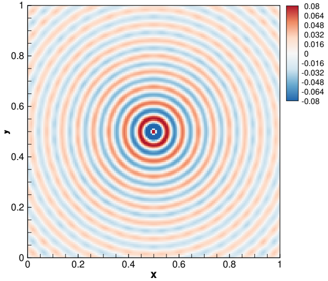
| Grid size | $k$ | $kh$ | ReD-$\mathcal{O}2$ | ReD-$\mathcal{O}4$ | ReD-Glk |
|---|---|---|---|---|---|
| 65 × 65 | 40 | 0.625 | **20** (98) | **17** (106) | **9** (128) |
| 129 × 129 | 80 | 0.625 | **30** (305) | **18** (298) | **9** (251) |
| 257 × 257 | 160 | 0.625 | **87** (731) | **19** (650) | **9** (585) |
| 513 × 513 | 320 | 0.625 | $>$**100** | **23** (1330) | **10** (1276) |
| 129 × 129 | 40 | 0.3125 | **18** (76) | **18** (81) | **7** (144) |
| 257 × 257 | 80 | 0.3125 | **19** (205) | **18** (212) | **7** (231) |
| 513 × 513 | 160 | 0.3125 | **21** (438) | **18** (458) | **7** (432) |
| 1025 × 1025 | 320 | 0.3125 | **28** (885) | **20** (909) | **6** (859) |
| 2049 × 2049 | 640 | 0.3125 | **53**(1722) | **23** (1763) | **6** (1690) |

"$>$" indicates it does not converge to the specified residual tolerance (Inner iterations $10^{-12}$, outer iterations $10^{-6}$) within a certain number of iterations.

- $Ex = Z^T A_h Zx$: #iter=**7** for $kh = 0.625$, **5** for $kh = 0.3125$
- ReD-$\mathcal{O}4$ better than ReD-$\mathcal{O}2$
- ReD-Glk: close to wavenumber independence




(a) Exact solution




(b) $kh = 0.625$

# Convergence - 2D Wedge

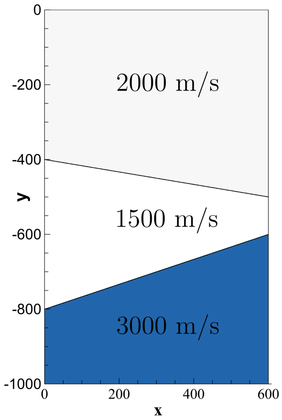



Figure: Wedge problem

# Convergence - 2D Wedge

Table: The number of iterations required by using higher-order A-DEF1 preconditioned GMRES. In parentheses is the number of CSLP-GMRES-iterations to solve the coarse grid problem once.

| Grid size | $f$ | $kh$ | ReD-$\mathcal{O}2$ | ReD-$\mathcal{O}4$ | ReD-Glk |
|---|---|---|---|---|---|
| 73× 121 | 10 | 0.35 | **22** (104) | **22** (108) | **9** (138) |
| 145× 241 | 20 | 0.35 | **28** (244) | **27** (243) | **9** (303) |
| 289× 481 | 40 | 0.35 | **31** (535) | **29** (519) | **9** (583) |
| 577× 961 | 80 | 0.35 | **37** (1200) | **30** (1175) | **9** (1255) |
| 1153× 1921 | 160 | 0.35 | >**50** | **34** (>1500) | **8** (>2500) |

">" indicates it does not converge to the specified residual tolerance (Inner iteration $10^{-12}$, outer iterations $10^{-6}$) within a certain number of iterations.

- $Ex = Z^T A_h Z x$: #iter=**6**
- ReD-$\mathcal{O}4$ better than ReD-$\mathcal{O}2$
- ReD-Glk: wavenumber independence although it is derived from constant wavenumber

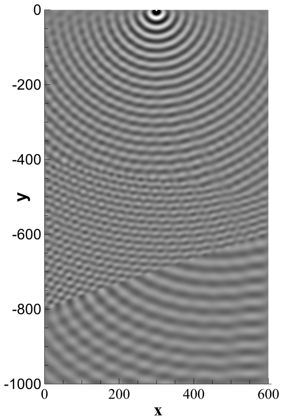
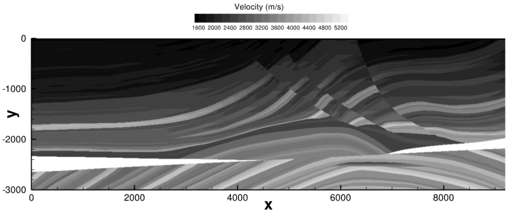
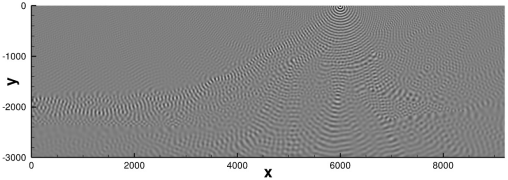



Figure: Waves pattern at 80 Hz

# Convergence - Marmousi




(a) Marmousi problem




(b) Wave pattern at $f = 40\,\text{Hz}$

Table: The number of iterations required by using higher-order A-DEF1 preconditioned GMRES. In parentheses is the number of CSLP-GMRES-iterations to solve the coarse grid problem once.

| Grid size | $f$ | $kh$ | ReD-$\mathcal{O}2$ | ReD-$\mathcal{O}4$ | ReD-Glk |
|---|---|---|---|---|---|
| 737 $\times$ 241 | 10 | 0.5236 | **38** (748) | **30** (762) | **10** (802) |
| 1473$\times$ 481 | 20 | 0.5236 | **71** (1988) | **34** (1947) | **10** (1923) |
| 2945$\times$ 961 | 40 | 0.5236 | $>$**50** | **50** ($>$2500) | **11** ($>$2500) |

- $Ex = Z^T A_h Z x$: #iter=7
- Similar convergence properties for highly heterogeneous media
- ReD-Glk: close to wavenumber independence

# Tolerance for the coarse grid problem - Recall

Table: Marmousi: the number of iterations required by using higher-order A-DEF1 preconditioned GMRES. In parentheses is the number of CSLP-GMRES-iterations to solve the coarse grid problem once.

| Grid size | $f$ | $kh$ | ReD-$\mathcal{O}2$ | ReD-$\mathcal{O}4$ | ReD-Glk |
|---|---|---|---|---|---|
| 737 $\times$ 241 | 10 | 0.5236 | 38 (748) | 30 (762) | 10 (802) |
| 1473$\times$ 481 | 20 | 0.5236 | 71 (1988) | 34 (1947) | 10 (1923) |
| 2945$\times$ 961 | 40 | 0.5236 | $>$50 | 50 ($>$2500) | 11 ($>$2500) |

- Tolerance for outer GMRES: $10^{-6}$
- Tolerance for the coarse grid problem solver (*i.e.* $y = E^{-1}x$): $10^{-12}$ $\rightarrow$ Sufficient but not necessary
- Outer solver: GMRES vs. Flexible GMRES / GCR

# Tolerance for the coarse grid problem

- Marmousi problem, $f = 20\,\text{Hz}$, grid size $1473 \times 481$.

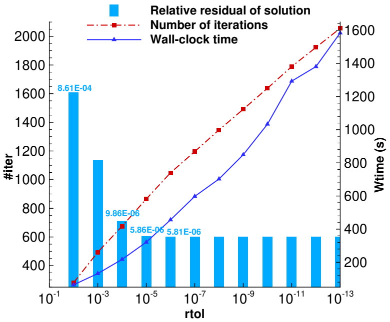



Figure: A-DEF1 preconditioned GMRES

- Outer #iter keeps constant
- $10^{-6}$ for the coarse grid problem is essential

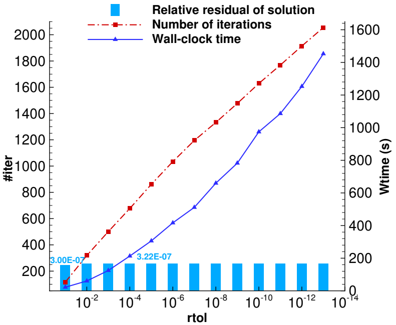



Figure: A-DEF1 preconditioned GCR

- Outer #iter keeps $10$, but $11$ for $10^{-1}$
- one more outer iteration, much less Wtime.

# Parallel performance - Weak scaling

- Preconditioned GCR
- higher-order two-level A-DEF1 using ReD-Glk
- DelftBlue, GNU Fortran 8.5.0, Open MPI 4.1.1

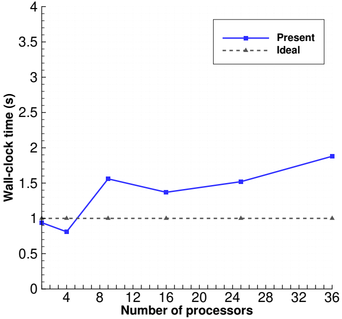


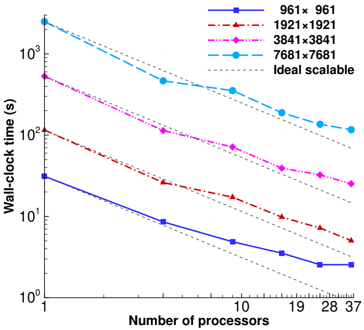

Figure: Weak scaling for constant-wavenumber problem with $k = 100$ and a grid size of $160 \times 160$ per processes.

Table: Weak scaling for model problems with non-constant wavenumber.

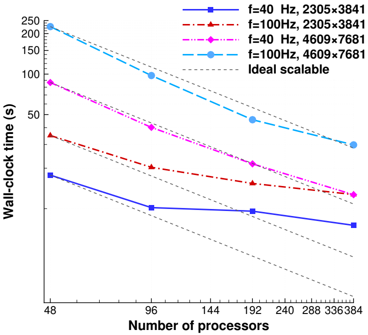
| grid size | np | #iter | CPU time (s) |
|---|---|---|---|
| Wedge, $f = 40$ Hz | | | |
| 577 $\times$ 961 | 6 | 10 (46) | 4.86 |
| 1153 $\times$ 1921 | 24 | 10 (43) | 5.75 |
| Marmousi, $f = 10$ Hz | | | |
| 737 $\times$ 241 | 3 | 11 (63) | 10.55 |
| 1473 $\times$ 481 | 12 | 10 (58) | 12.08 |
| 2945 $\times$ 961 | 48 | 10 (58) | 17.72 |

Close to weak scalability

# Parallel performance - Strong scaling




(a) Constant-wavenumber problem with $k = 200$




(b) Wedge problem with $f = 40$ Hz and $f = 100$ Hz

Figure: Strong scaling

Good strong scaling for large problems (larger computation/communication ratio)

# Multilevel Deflation

- Line 6: $\tilde{v} \approx E^{-1}\hat{v}$: apply two-level method recursively
- Only one FGMRES iteration per level except for the coarsest level
- Line 10: $r \approx M^{-1}\tilde{r}$ : max $\mathcal{O}(n^{0.25})$ Krylov iterations instead of multigrid
- Re-discretization scheme derived from Galerkin coarsening for both $E$ and $M$
  - The size of the stencil remains $7 \times 7$ (2D) for level $> 3$
  - Need three overlapping grid points
  - Truncate on the near-boundary grid points, not need extra boundary schemes
- Convergence slightly depends on wavenumber

**Algorithm 1:** Two-level deflation FGMRES

1: Choose $u_0$ and dimension $k$ of the Krylov subspace.
2: Define $(k+1) \times k\bar{H}_k$ and initialize to zero
3: Compute $r_0 = b - Au_0$, $\beta = ||r_0||$, $v_1 = r_0/\beta$;
4: **for** $j = 1, 2, ..., k$ or until convergence **do**
5: $\quad \hat{v}_j = Z^T v_j$
6: $\quad \tilde{v} \approx E^{-1}\hat{v}$, solve approximately
7: $\quad t = Z\tilde{v}$
8: $\quad s = At$
9: $\quad \tilde{r} = v_j - s$
10: $\quad r \approx M^{-1}\tilde{r}$
11: $\quad x_j = r + t$
12: $\quad w = Ax_j$
13: $\quad$ **for** $i := 1, 2, ..., j$ **do**
14: $\quad\quad h_{i,j} = (w, v_i)$
15: $\quad\quad w := w - h_{i,j} v_i$
16: $\quad$ **end for**
17: $\quad h_{j+1,j} := ||w||_2$, $v_{j+1} = w/h_{j+1,j}$
18: $\quad X_k = [x_1, ..., x_k]$; $\bar{H}_k = \{h_{i,j}\}_{1 \le i \le j+1, 1 \le j \le m}$
19: **end for**
20: $u_k = u_0 + X_k y_k$ where $y_k = \arg\min_y ||\beta e_1 - \bar{H}_k y||$

# Conclusions and Perspectives

- ✓ Parallel CSLP preconditioned Krylov solvers (2D/3D)
- ✓ Parallel two-level deflation preconditioned Krylov solvers (2D)
- ✓ Matrix-free implementation with wavenumber-independent convergence
- ✓ Parallel framework with fairly good weak and strong scaling
- ✓ Limited by coarse grid solver
- ⟳ Parallel multilevel deflation method
- ⟳ Generalize to large-scale 3D applications

Further reading:


- Dwarka, V., Vuik, C.: Scalable convergence using two-level deflation preconditioning for the Helmholtz equation, SIAM Journal on Scientific Computing 42 (2020) A901-A928.
- Dwarka, V., Vuik, C.: Scalable multi-level deflation preconditioning for highly indefinite time-harmonic waves, Journal of Computational Physics 469 (2022) 111327
- Chen, J., Dwarka, V., Vuik, C.: A matrix-free parallel solution method for the three-dimensional heterogeneous Helmholtz equation, `https://doi.org/10.48550/arXiv.2308.06085`.
- Chen, J., Dwarka, V., Vuik, C.: A Matrix-free parallel two-level deflation preconditioner for the two-dimensional Helmholtz problems, `https://doi.org/10.48550/arXiv.2308.06152`.

# Q&A

Thanks!