# A scalable parallel solver for Helmholtz problems

Cornelis Vuik, Jinqiang Chen, Vandana Dwarka

Technische Universiteit Delft

# Aim and Impact

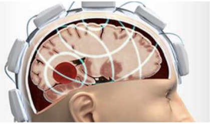
- Contribute to broad research on parallel scalable iterative solvers for time-harmonic wave problems
- This presentation: matrix-free parallelization
  - Complex shift Laplace Preconditioner (CSLP)
  - Deflation methods
  - Parallel performance

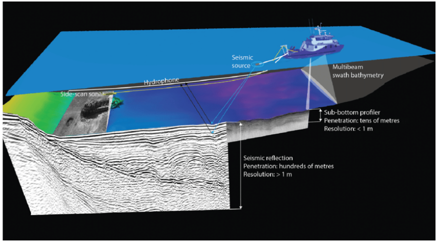
# Introduction - the Helmholtz Problem

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

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

- $\Delta$ - Laplace operator, $u(\mathbf{x})$ - Fourier-space representation of the wave function
- $k(\mathbf{x})$ - wavenumber, $k(\mathbf{x}) = (2\pi f)/c(\mathbf{x})$, where $f$ - frequency, $c$ - wave velocity
- 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 - Challenges

- Linear system from 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; High-Performance Computing (HPC)
- **Negative & positive eigenvalues**: larger wavenumber $\Rightarrow$ more iterations
  - Preconditioner: Complex Shifted Laplace Preconditioner (CSLP)
  - (Higher-order) Deflation
- **Parallelism**

## Aim

- A wavenumber-independent convergent and parallel scalable solver

# Introduction - Metrics

- 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 $\Gamma = \partial\Omega$

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

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

  - Constant wavenumber: $k(\mathbf{x}) = k$
  - Non-constant wavenumber: Wedge, Marmousi problem
- Finite-difference discretization on a uniform grid with grid size $h$. (2D example)
  - Laplace operator:

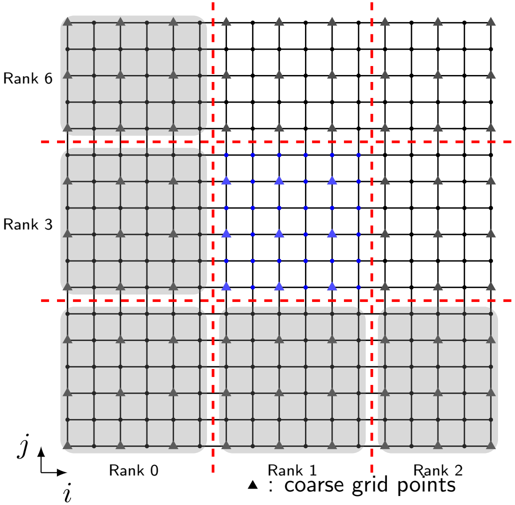
$$-\Delta_h \mathbf{u} \approx \frac{-u_{i,j-1} - u_{i-1,j} + 4u_{i,j} - u_{i+1,j} - u_{i,j+1}}{h^2}$$

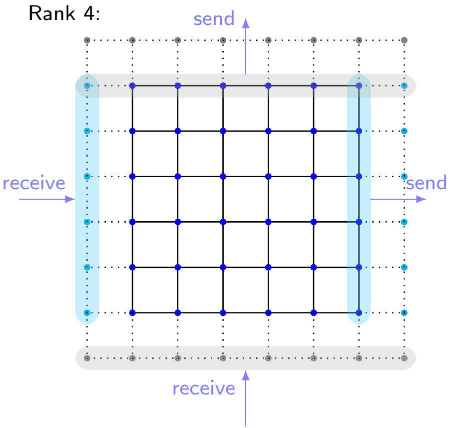
  - Sommerfeld BCs: a ghost point

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

# 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



# 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 - Matrix-free operations

**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":

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

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

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

# 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

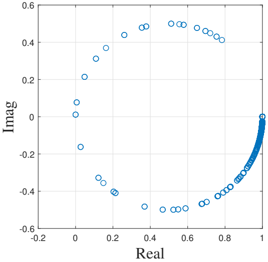
- Solve $Mx = u$ by multigrid method (V-cycle) $\Rightarrow x \approx M^{-1}u$
  - **Vertex-centered** coarsening based on the **global** grid
  - Damped Jacobi smoother (easy to parallelize)
  - Full-weight restriction $I_h^{2h}$ & linear interpolation $I_{2h}^h$

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

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

# CSLP - Cons

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



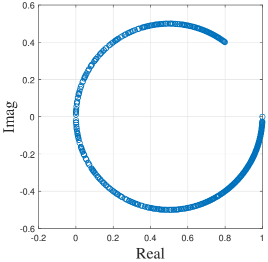




Figure: $\sigma\left(M_{(1,0.5)}^{-1}A\right)$ for $k = 20$ (left) and $k = 80$ (right)

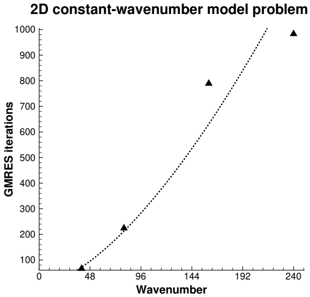



Figure: #Iter increases with $k$

# Deflation - introduction

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

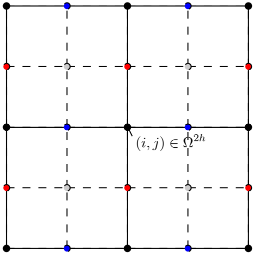
$$P = I - AQ, \quad \text{where } Q = ZE^{-1}Z^T, \quad E = Z^TAZ$$
$$A \in \mathbb{R}^{n \times n}, Z \in \mathbb{R}^{m \times n}$$

- 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
- Linear approximation basis deflation vectors $\rightarrow$ **higher-order** deflation vectors (Adapted Preconditioned DEF, **APD**)
  - 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

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

- With matrix constructed, $E = Z^T AZ$, 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

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

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

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

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

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

# Two-level deflation - overall algorithm

- Flexible GMRES-type methods $\rightarrow$ allow for variable preconditioner

**Algorithm** Two-level deflation FGMRES

Choose $u_0$ and dimension $k$ of the Krylov subspace.
Define $(k+1) \times k \bar{H}_k$ and initialize to zero
Compute $r_0 = b - Au_0$, $\beta = ||r_0||$, $v_1 = r_0/\beta$;
**for** $j = 1, 2, ..., k$ or until convergence **do**
    $\hat{v}_j = Z^T v_j$ ▷ Precondition starts
    $\tilde{v} \approx E^{-1} \hat{v}$ ▷ **Solved by GMRES approximately, preconditioned by CSLP**, **tol=**$10^{-1}$
    $t = Z\tilde{v}$
    $s = At$
    $\tilde{r} = v_j - s$
    $r \approx M^{-1} \tilde{r}$ ▷ Approximated by one multigrid V-cycle
    $x_j = r + t$ ▷ Precondition ends
    $w = Ax_j$
    **for** $i := 1, 2, ..., j$ **do**
        $h_{i,j} = (w, v_i)$
        $w := w - h_{i,j} v_i$
    **end for**
    $h_{j+1,j} := ||w||_2$, $v_{j+1} = w/h_{j+1,j}$; $X_k = [x_1, ..., x_k]$; $\bar{H}_k = \{h_{i,j}\}_{1 \le i \le j+1, 1 \le j \le m}$
**end for**
$u_k = u_0 + X_k y_k$ where $y_k = \text{arg min}_y \left|\left|\beta e_1 - \bar{H}_k y\right|\right|$

# Tolerance for the coarse grid problem

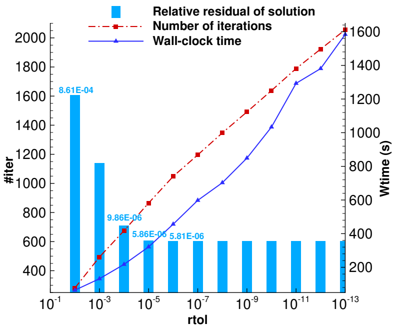



Figure: APD preconditioned GMRES

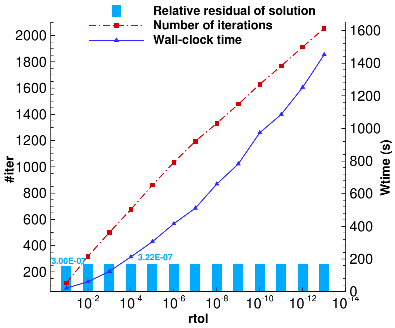



Figure: APD preconditioned FGMRES

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

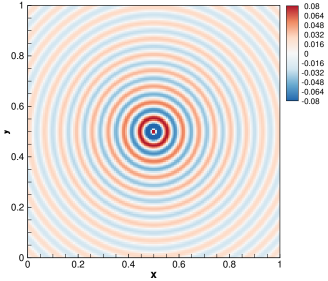
- Outer #iter keeps $10$, but $11$ for $10^{-1}$
- One more outer iteration, much less Wtime

# Convergence - Constant wavenumber

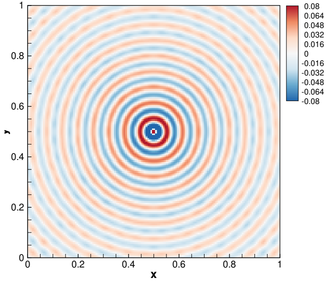
Table: The number of iterations required by using APD-GMRES.

| Grid size | $k$ | $kh$ | ReD-$\mathcal{O}2$ | ReD-$\mathcal{O}4$ | ReD-Glk |
|---|---|---|---|---|---|
| 65 × 65 | 40 | 0.625 | **20** | **17** | **9** |
| 129 × 129 | 80 | 0.625 | **30** | **18** | **9** |
| 257 × 257 | 160 | 0.625 | **87** | **19** | **9** |
| 513 × 513 | 320 | 0.625 | **319** | **23** | **10** |
| 1025 × 1025 | 640 | 0.625 | **1099** | **34** | **11** |
| 2049 × 2049 | 1280 | 0.625 | **3417** | **79** | **13** |
| 129 × 129 | 40 | 0.3125 | **18** | **18** | **7** |
| 257 × 257 | 80 | 0.3125 | **19** | **18** | **7** |
| 513 × 513 | 160 | 0.3125 | **21** | **18** | **7** |
| 1025 × 1025 | 320 | 0.3125 | **28** | **20** | **6** |
| 2049 × 2049 | 640 | 0.3125 | **53** | **23** | **6** |

- $Ex = Z^T A_h Z x$: #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

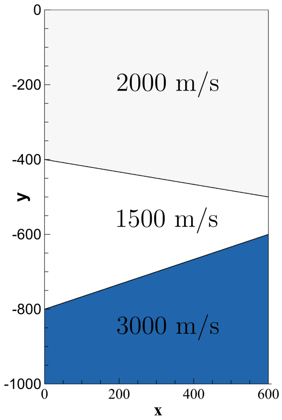



Figure: Wedge problem

# Convergence - 2D Wedge

Table: The number of iterations required by using APD-GMRES.

| Grid size | $f$ | $kh$ | ReD-$\mathcal{O}$2 | ReD-$\mathcal{O}$4 | ReD-Glk |
|---|---|---|---|---|---|
| 73× 121 | 10 | 0.35 | **22** | **22** | **9** |
| 145× 241 | 20 | 0.35 | **28** | **27** | **9** |
| 289× 481 | 40 | 0.35 | **31** | **29** | **9** |
| 577× 961 | 80 | 0.35 | **37** | **30** | **9** |
| 1153× 1921 | 160 | 0.35 | **58** | **34** | **8** |

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

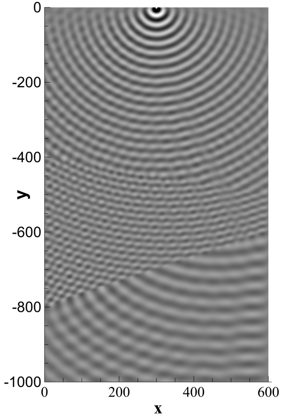
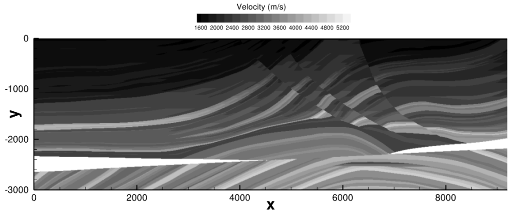
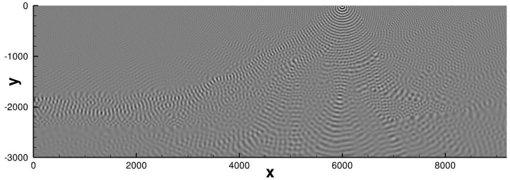



Figure: Waves pattern at 80 Hz

# Convergence - Marmousi




(a) Marmousi problem




(b) Wave pattern at $f = 40$ Hz

Table: The number of iterations required by using APD-FGMRES.

| Grid size | $f$ | $kh$ | ReD-$\mathcal{O}$2 | ReD-$\mathcal{O}$4 | ReD-Glk |
|---|---|---|---|---|---|
| 737 × 241 | 10 | 0.5236 | **40** | **33** | **11** |
| 1473× 481 | 20 | 0.5236 | **71** | **35** | **12** |
| 2945× 961 | 40 | 0.5236 | **233** | **41** | **12** |

- $Ex = Z^T A_h Z x$: #iter=8
- ReD-Glk: close to wavenumber independence for highly heterogeneous media
- Many iterations are required to solve the coarse grid problem (in parentheses)

$\Rightarrow$ Apply two-level method recursively $\Rightarrow$ Multilevel deflation

# Parallel performance - Weak scaling

- APD using ReD-Glk
- DelftBlue, GNU Fortran 8.5.0, Open MPI 4.1.1

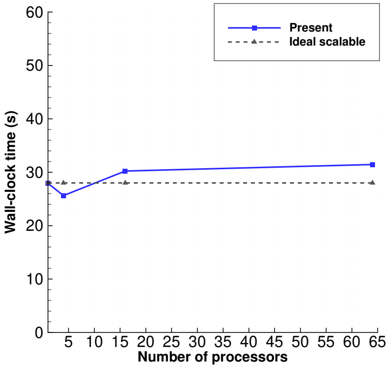


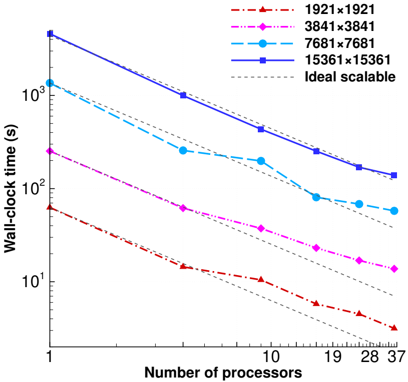

Figure: Weak scaling for constant-wavenumber problem with $k = 160$ and a grid size of $1024 \times 1024$ per processes.

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

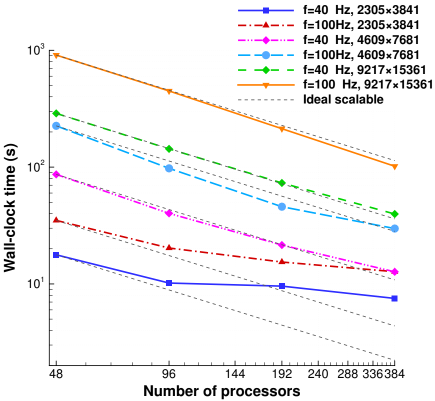
| grid size | np | #iter | CPU time (s) |
|---|---|---|---|
| Wedge, $f = 40\,\text{Hz}$ | | | |
| 577 × 961 | 6 | 10 (46) | 4.86 |
| 1153 × 1921 | 24 | 10 (43) | 5.75 |
| Marmousi, $f = 10\,\text{Hz}$ | | | |
| 737 × 241 | 3 | 11 (63) | 10.55 |
| 1473 × 481 | 12 | 10 (58) | 12.08 |
| 2945 × 961 | 48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

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

# Multilevel deflation

Table: The number of outer iterations required to solve the Wedge problem with $kh = 0.17$ by using the **multilevel (six-level)** APD-FGMRES.

| Grid size | $f$ (Hz) | Outer #iter (L2 #iter) |
|---|---|---|
| 289× 481 | 20 | 11 (3) |
| 577× 961 | 40 | 12 (4) |
| 1153× 1921 | 80 | 12 (7) |
| 2305× 3841 | 160 | 13 (13) |
| 4609× 7681 | 320 | 14 (27) |
| 9217× 15361 | 640 | 17 (47) |

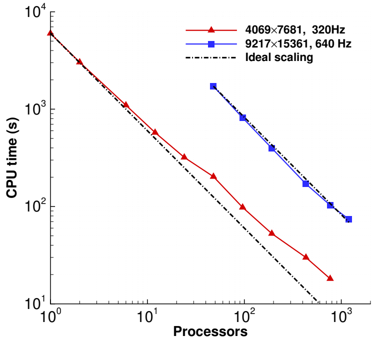



Figure: Strong scaling for Wedge problem

- The number of iterations weakly depends on the frequency
- Good performance for massively parallel computing

# Conclusions

- Parallelization of higher-order deflation preconditioned Krylov solvers
- Matrix-free implementation with wavenumber-independent convergence
- Parallel framework with good weak and strong scalability

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, Electronic Transactions on Numerical Analysis, 59 (2023), 270–294.
- Chen, J., Dwarka, V., Vuik, C.: A matrix-free parallel two-level deflation preconditioner for two-dimensional heterogeneous Helmholtz problems, Journal of Computational Physics 514 (2024): 113264.
- Chen, J., Dwarka, V., Vuik, C.: Matrix-Free Parallel Scalable Multilevel Deflation Preconditioning for Heterogeneous Time-Harmonic Wave Problems, Journal of Scientific Computing, 102.2 (2025), 1-33.

# Q&A

Thanks!