A scalable solver for the Helmholtz problems

Cornelis Vuik*, Jinqiang Chen, Vandana Dwarka

Technische Universiteit Delft

Aim and Impact

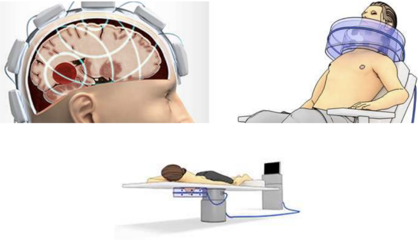
- Contribute to broad research on parallel scalable iterative solvers for Helmholtz problems
- This presentation: matrix-free parallelization
  - Complex shift Laplace Preconditioner (CSLP)
  - Deflation methods
  - Parallel performance

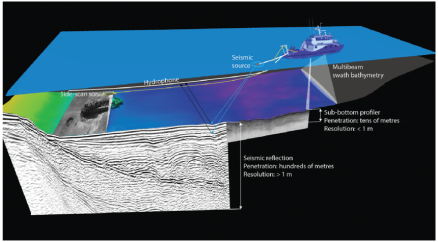
Introduction - the Helmholtz Problem

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

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

- Δ - 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) ⇒ larger linear system
  - Memory-efficient methods; High-Performance Computing (HPC)
- ⚠ **Negative & positive eigenvalues**: larger wavenumber ⇒ 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)
- 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: $S_p = \dfrac{t_{w,r}}{t_{w,p}}$, $E_P = \dfrac{S_p}{np/np_r} = \dfrac{t_{w,r} \cdot np_r}{t_{w,p} \cdot np}$

Introduction - Numerical Models

- Model problems on a rectangular domain Ω 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 - Matrix-free operations

- Perform computations with a matrix without explicitly forming or storing the matrix
  ⇒ 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

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

  - 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

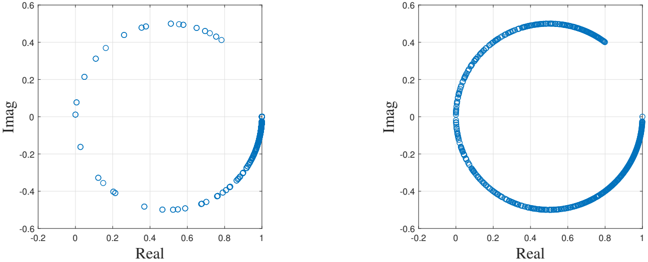


Figure: $\sigma\left(M_{(1,0.5)}^{-1}A\right)$ for $k = 20$ (left) and $k = 80$ (right)

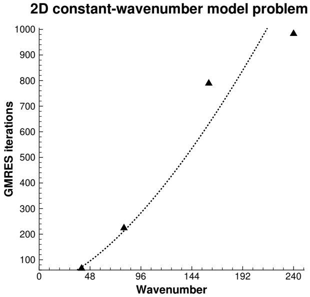


Figure: #Iter increases with k

Deflation - introduction

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

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

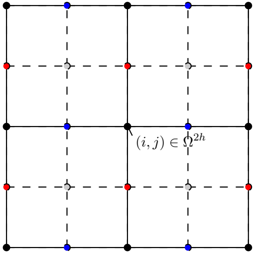
$$A \in \mathbb{R}^{n\times n}, Z \in \mathbb{R}^{n\times m}$$

- 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 → **higher-order** deflation vectors (Adapted Preconditioned DEF, **APD**)
  - wavenumber-independent convergence
- Flexible GMRES-type methods → approximate E^{-1}, tol$=10^{-1}$

Higher-order deflation vectors

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

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



●●●● : fine grid points $\in \Omega^h$

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

Figure: The allocation map of interpolation operator

Matrix-free coarse-grid operator

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

- 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 coarse-grid operator

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

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

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

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

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

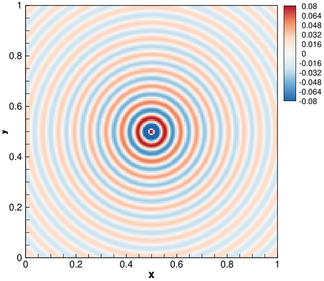
? Boundary conditions - ReD-$\mathcal{O}2$ on the boundary grid points

Convergence - Constant wavenumber

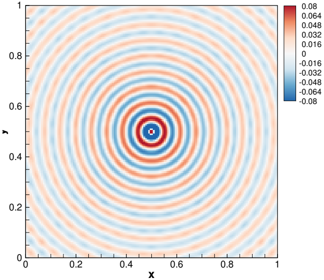
Table: The number of iterations required by using APD-GMRES.

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**

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

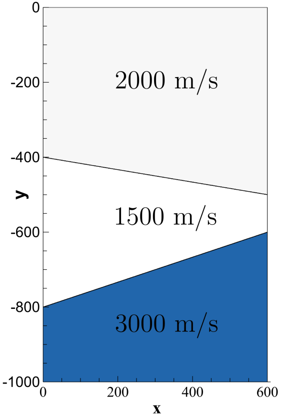


Figure: Wedge problem

Convergence - 2D Wedge

Table: The number of iterations required by using 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

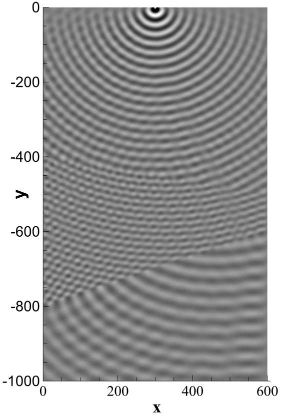
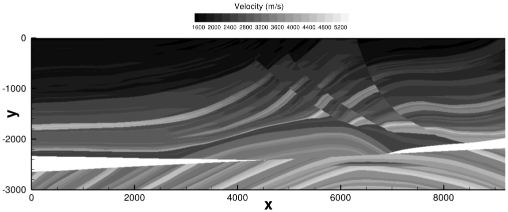
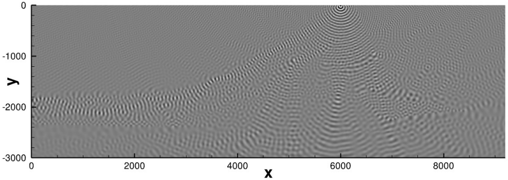


Figure: Waves pattern at 80 Hz

Convergence - Marmousi



(a) Marmousi problem



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

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

Grid size	f	kh	ReD-$\mathcal{O}$2	ReD-$\mathcal{O}$4	ReD-Glk
737 × 241	10	0.5236	**40**	**33**	**11 (64)**
1473× 481	20	0.5236	**71**	**35**	**11 (141)**
2945× 961	40	0.5236	**233**	**41**	**12 (381)**

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

Multilevel Deflation

- Apply two-level method recursively
- **Re-discretization scheme** derived from Galerkin coarsening for **both** E and M
  - The size of the stencil remains 7×7 for level > 3
  - Need three overlapping grid points
  - Zero-padding on the near-boundary grid points, not need extra boundary schemes
- **V-cycle**: Only one FGMRES iteration per **coarse** level except for the **coarsest** level, *i.e.* $m = 1$ in line 4
  - CSLP: Krylov iterations instead of multigrid
    - Max $\mathcal{O}(N^{0.25})$ iterations or tol=10^{-1}
    - Small complex shift: $1/k_{max}$
  - Coarsest level: solved by CSLP-GMRES, tol=10^{-1}

Algorithm Recursive two-level deflated FGMRES: `TLADP-FGMRES(A, b)`

```
1:  Determine the current level $l$ and dimension $m$ of the Krylov subspace
2:  Initialize $u_0$, compute $r_0 = b - Au_0$, $\beta = ||r_0||$, $v_1 = r_0/\beta$;
3:  Define $\bar{H}_m \in \mathbb{C}^{(m+1)\times m}$ and initialize to zero
4:  for $j = 1, 2, ..., m$ or until convergence do
5:      $\hat{v}_j = Z^T v_j$                                  ▷ Restriction
6:      if $l + 1 == l_{max}$ then                   ▷ Predefined coarsest level $l_{max}$
7:          $\tilde{v} \approx E^{-1}\hat{v}$                  ▷ Approximated by CSLP-FGMRES
8:      else
9:          $l \leftarrow l + 1$
10:         $\tilde{v} \leftarrow$TLADP-FGMRES(E, $\hat{v}$)   ▷ Apply two-level deflation recursively
11:     end if
12:     $t = Z\tilde{v}$                                      ▷ Interpolation
13:     $s = At$
14:     $\tilde{r} = v_j - s$
15:     $r \approx M^{-1}\tilde{r}$               ▷ CSLP, by multigrid method or Krylov iterations
16:     $x_j = r + t$
17:     $w = Ax_j$
18:     for $i := 1, 2, ..., j$ do
19:         $h_{i,j} = (w, v_i)$
20:         $w \leftarrow w - h_{i,j}v_i$
21:     end for
22:     $h_{j+1,j} := ||w||_2$, $v_{j+1} = w/h_{j+1,j}$
23:     $X_m = [x_1, ..., x_m]$, $\bar{H}_m = \{h_{i,j}\}_{1\leq i\leq j+1, 1\leq j\leq m}$
24: end for
25: $u_m = u_0 + X_m y_m$ where $y_m = \arg\min_y ||\beta e_1 - \bar{H}_m y||$
26: Return $u_m$
```

Multilevel deflation - V-cycle

Remark

$\exists \tilde{m}$: for $m > \tilde{m}$, E_m is **negative definite**. For $m \le \tilde{m}$, E_m is indefinite.

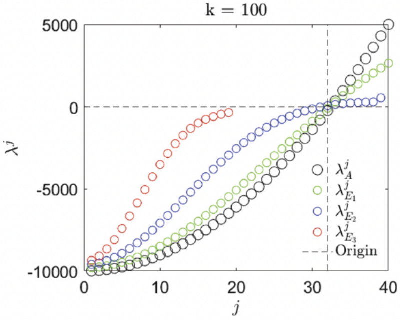


Figure: Spectrum of the coarse linear systems for $k = 100$ and $kh = 0.3125$.

Table: Number of outer FGMRES-iterations and CPU time required for the Wedge problem with $kh = 0.35$. The coarse-grid systems become **negative definite** from the **4th level**.

f (Hz)	Grid size	Three-level		Four-level	
		Outer #iter	CPU time (s)	Outer #iter	CPU time (s)
20	145×241	7	3.78	8	7.00
40	289×481	7	20.14	9	103.31
80	577×961	8	195.14	11	907.00
160	1153×1921	8	1060.50	13	5101.73

- V-cycle: coarsening needs to **remain on indefinite levels**
- What about coarsening to **negative definite** levels?

Multilevel deflation - a robust and efficient variant

For the scenario of coarsening to **negative definite** levels:

- A **tolerance** for **the second level (L2)** (instead of one FGMRES iteration)
  - L2 tol=1×10^{-1} → close to constant outer iterations
  - L2 tol=3×10^{-1} → extra outer iterations but **reduced computation time** ✓
- **One** FGMRES iteration for **the other coarse levels** including the coarsest level
- CSLP: **the first and second** levels: multigrid method (one V-cycle); **the other** coarse levels: Krylov iterations (GMRES), tol=1×10^{-1}

Table: Number of outer FGMRES-iterations and **sequential** CPU time required to solve the Marmousi problem. For $kh = 0.54$, the coarse-grid systems become **negative definite** starting from the **3rd level**. In parentheses are the number of iterations to solve the second-level grid system.

f (Hz)	Grid size	Two-level, L2 tol=1×10^{-1}		Five-level, L2 tol=1×10^{-1}		Five-level, L2 tol=3×10^{-1}	
		Outer #iter (L2 #iter)	CPU time (s)	Outer #iter (L2 #iter)	CPU time (s)	Outer #iter (L2 #iter)	CPU time (s)
10	737×241	11 (64)	23.15	11 (13)	18.57	13 (7)	12.67
20	1473×481	11 (141)	224.21	11 (24)	108.03	15 (15)	84.06
40	2945×961	12 (381)	4354.83	13 (50)	1084.42	18 (29)	816.38

Multilevel deflation - complexity analysis

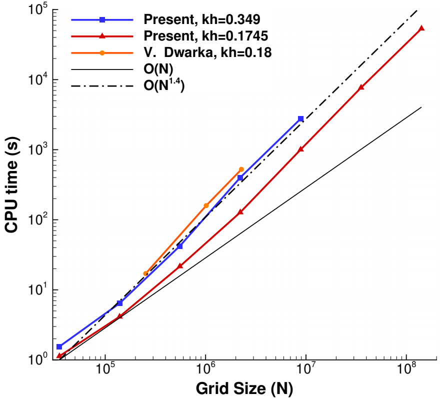


Figure: Complexity analysis of the multilevel APD preconditioned Krylov subspace method. Evolution of the **sequential** computational time versus problem size. Wedge model problem.

Table: The number of outer iterations required to solve the Wedge problems with $kh = 0.17$ by using the multilevel APD-FGMRES.

Six-level deflation, L2 tol=3×10^{-1}		
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)

- ✓ The number of iterations **weakly** depends on the frequency
- ✓ The computational time behaves asymptotically as $N^{1.4}$

Parallel performance

- Six-level deflation Preconditioned FGMRES
- **DeltfBlue**, GNU Fortran 8.5.0, Open MPI 4.1.1

Table: Weak scaling for constant-wavenumber problem, $k = 1600$.

Grid size	N	np	#iter	CPU time (s)
5121×5121	26,224,641	64	14	100.84
10241×10241	104,878,081	256	13	79.69
20481×20481	419,471,361	1024	13	93.62

Table: Weak scaling for the Wedge model problem, $f = 320$ Hz.

Grid size	N	np	#iter	CPU time (s)
2305×3841	8,853,505	48	16	69.75
4609×7681	35,401,729	192	14	53.20
9217×15361	141,582,337	768	14	67.03

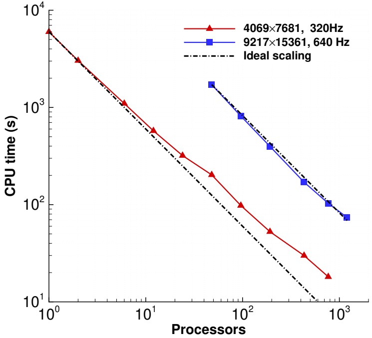


Figure: Strong scaling for Wedge problem

- ✓ Good **weak scalability** for large wavenumber - in the context of **minimizing pollution error** by grid **refinement**.
- ✓ Good strong scalability for massively parallel computing

Conclusions and Perspectives

- ✓ Parallel CSLP preconditioned Krylov solvers
- ✓ Parallel two-level deflation preconditioned Krylov solvers
- ✓ Robust parallel multilevel deflation for high-frequency heterogeneous problems
- ✓ Matrix-free implementation with wavenumber-independent convergence
- ✓ Parallel framework with fairly good weak and strong scaling
- ⟳ Generalize to real-world 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, Electronic Transactions on Numerical Analysis, 59 (2023), 270–294.
- Chen, J., Dwarka, V., Vuik, C.: A matrix-free parallel two-level deflation preconditioner for the two-dimensional Helmholtz problems, Journal of Computational Physics, 514 (2024), 113264.

Q&A

Thanks!