Iterative methods for 2D/3D Helmholtz operator

Yogi Erlangga†‡⋆
SLIM EOS - UBC
yerlangga@eos.ubc.ca

jointly with

Kees Vuik†, Kees Oosterlee† (Math – TU Delft), Dwi Riyanti† (formerly at Math – TU Delft)
Reinhard Nabben‡ (Math – TU Berlin)

† Financially supported by The Dutch Ministry of Economic Affairs, under Project BTS-01044.
‡ Financially supported by DFG, Germany, under Project NA248/2-2.
⋆ Currently as a part of SINBAD, financially supported by BG, BP, Chevron, ExxonMobil, and Shell, secured through ITF. SINBAD is part of the collaborative research & development (CRD) grant number 334810-05 funded by the Natural Science and Engineering Research Council (NSERC).
Outline of the talk:

• Introduction: motivation
• Time-harmonic waves and solvers:
  – direct vs iterative methods
  – problems with standard iterative methods
• Towards grid-independent convergence: preconditioner
• Towards frequency-independent convergence: projection (multilevel Krylov)
• Future work
The 2D Marmousi model


Velocity Model

Time-domain solution, Traces on the reciever plane
Computed by inverse FFT of frequency-domain solution
Frequency domain-Finite difference inversion

[Laillly, Tarantola, Pratt, ...]

Given \( d \) field receivers, \( r \) computational receivers;
Given initial velocity model (background) \( c^0(x) \). Define \( m^0(x) = 1/(c^0(x))^2 \)

for \( f = 1, \ldots, n_f \)
  for \( k = 1, \ldots \) until convergence
    for \( s = 1, \ldots, n_s \)
      Solve for \( u_{f,s}^k: A(m^k(x), \omega_f)u_{f,s}^k(x) = b_s; \) \[ \text{[Forward propagation]} \]
    end
    \( \delta d = u_{f,s}^k(x_r) - R_d^r d_{f,s}(x_d) \)
    \( m^{k+1}(x) = m^k(x) - \alpha^k \nabla_m E^k, \quad E^k = \frac{1}{2} \sum_f \sum_r \delta d^T \delta d^* \)
  end
end
Frequency domain-Finite difference inversion

[Lailly, Tarantola, Pratt, ...]

Given \( d \) field receivers, \( r \) computational receivers;
Given initial velocity model (background) \( c^0(x) \). Define \( m^0(x) = 1/(c^0(x))^2 \)

for \( f = 1, \ldots, n_f \)
  
  for \( k = 1, \ldots \) until convergence
    
    for \( s = 1, \ldots, n_s \)
      
      Solve for \( u_{f,s}^{k} \): \( A(m^k(x), \omega_f)u_{f,s}^k(x) = b_{s} \); [Forward propagation]
    
    \( \delta d = u_{f,s}^{k}(x_r) - R_{d}f_{s}(x_d) \)
    
    \( m^{k+1}(x) = m^k(x) - \alpha^k \nabla_m E^k, \quad E^k = \frac{1}{2} \sum_f \sum_r \delta d^T \delta d^* \)
  
end

\( \alpha^k, \ \nabla_m E^k \) can be equivalently computed by solving \textit{back propagations} with multiple right-hand sides (RHS):

\[
A^T(m^k, \omega_f)[v_1 \ldots v_r] = [w_1 \ldots w_r]
\]
Why frequency domain?

• Less work per frequency: number of spatial grid, $N$, is far less than in time domain
  → low frequency, coarse grid
    For 2D Marmousi:
    Lowest frequency: 1 Hz. Grid: $751 \times 201$
    Medium frequency: 30 Hz. Grid: $2,001 \times 534$
    Highest frequency: 60 Hz. Grid: $2,501 \times 751$
• Frequency step can be made large (frequency can be sampled)
  → no CFL-like criteria
• Convolution in time domain = multiplication in frequency domain (faster and stable)
• Reversed time = backward propagation
  Forward modelling with “data” as source (no time history needed)
Helmholtz equation and solvers

Time harmonic wave operator:

\[ \mathcal{A} := -\frac{\omega^2}{c^2(x)} - \nabla \cdot \nabla, \quad \omega : \text{temporal frequency} \]

\[ \mathcal{A}u = b \xrightarrow{\text{Finite difference}} Au = b, \quad A \in \mathbb{C}^{N \times N} \]

The matrix \( A \) is

- sparse: nonzero \( nz(A) = cN, \ c \ll N \)
- symmetric (\( A^T = A \)) but non-Hermitian (\( A^H \neq A \))
- large: higher \( \omega \), larger \( N \) (controlled by minimum grid points per smallest wavelength, \( \lambda \))
- indefinite, especially for practical \( \omega \): \( \text{Re}(\mu(A)) \in \mathbb{R}/\{0\}, \ \mu \) the eigenvalue of \( A \)
- ill-conditioned: the condition number of \( A \), \( \kappa(A) \), is large, \( |\mu|_{\text{min}} = \mathcal{O}(h^{-2}) \)
Helmholtz equation and solvers

For one $\omega$: solve $Au = b$ and $AV = W$ ($A^T = A$)
where $V = [v_1 \ldots v_r]$ and $W = [w_1 \ldots w_r]$
For one $\omega$: solve $Au = b$ and $AV = W$ ($A^T = A$)
where $V = [v_1 \ldots v_r]$ and $W = [w_1 \ldots w_r]$

Direct solver:
- LU factorization: $A = LU$
- solve (backward and forward substitution) $LUu = b$ and $LU[v_1 \ldots v_r] = [w_1 \ldots w_r]$

LU factorization is very costly, but computed only once (for an $\omega$).
Helmholtz equation and solvers

For one $\omega$: solve $Au = b$ and $AV = W$ ($A^T = A$)
where $V = [v_1 \ldots v_r]$ and $W = [w_1 \ldots w_r]$

Direct solver:
- LU factorization: $A = LU$
- solve (backward and forward substitution) $LUu = b$ and $LU[v_1 \ldots, v_r] = [w_1 \ldots, w_r]$

LU factorization is very costly, but computed only once (for an $\omega$).

Iterative solver:
- solve one-by-one: $Au = b$, $Av_1 = w_1 \ldots Av_r = w_r$
- solve $Au = b$, followed by $AV = W$

The solver must be uniformly convergent.
Comparison of direct and iterative solvers

\[ A \in \mathbb{R}^{N \times N} \text{ sparse, } B\text{-banded matrix} \]

<table>
<thead>
<tr>
<th></th>
<th>Direct Solver(^\dagger)</th>
<th>Iterative solver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity</td>
<td></td>
<td>( C_3 \cdot N )</td>
</tr>
<tr>
<td>• Reordering</td>
<td>( C_1 \cdot N \log N )</td>
<td></td>
</tr>
<tr>
<td>• LU-factorization</td>
<td>( C_2 \cdot N^{1+\alpha} )</td>
<td>( C_3 \cdot N )</td>
</tr>
<tr>
<td>• back-forward solve</td>
<td>( B \cdot N )</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>( B \cdot N )</td>
<td>( C_4 \cdot N )</td>
</tr>
<tr>
<td>Parallel-ability</td>
<td>–</td>
<td>+</td>
</tr>
<tr>
<td>Robustness</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>Multiple RHS (shots)</td>
<td>+</td>
<td>–</td>
</tr>
</tbody>
</table>

\(^\dagger\text{PARDISO, Schenk & Gärtner}\)
Helmholtz equation and solvers

Illustration: \[-(\omega/c)^2 u - \nabla \cdot \nabla u(x) = b, \quad x \in [0, 1]^2\]

[Direct method vs iterative method (GMRES and Bi-CGSTAB)]

From memory:
GMRES is not efficient (increased Arnoldi vectors)

In large 3D, computing \(LU\) factors is out of the question.
Helmholtz equation and solvers

Illustration: \(- (\omega/c)^2 u - \nabla \cdot \nabla u(x) = b, \ x \in [0, 1]^2\)

[Direct method vs iterative method (GMRES and Bi-CGSTAB) with ILU(0.01) preconditioner]

Convergence:
- dependent on \(\omega\)
- high \(\omega\) breakdown
- dependent of \(h\)
  (not shown here)

Efficient preconditioner must lead to \(h\)- and \(\omega\)-independent convergence.
Grid-independent convergence

Preconditioner operator:

\[ \mathcal{M} := -(1 - 0.5\hat{i}) \frac{\omega^2}{c^2(x)} - \nabla \cdot \nabla, \quad \hat{i} = \sqrt{-1}. \]

\[ \mathcal{M} z = y \xrightarrow{\text{Finite difference}} Mz = y, \quad M \in \mathbb{C}^{N \times N} \]

Then solve: \[ AM^{-1} \tilde{u} = b, \quad u = M^{-1} \tilde{u}. \]

Preconditioning:

- \( \kappa(AM^{-1}) \ll \kappa(A) \)
- \( |\mu(AM^{-1})|_{\text{max}} \rightarrow 1 \)
- \( |\mu(AM^{-1})|_{\text{min}} = (k^{-1}), \quad k \) the wavenumber

Practical aspect:

- \( M^{-1} \) is approximately done by one multigrid (MG) sweep (\( O(N) \) complexity)

[E., Oosterlee, Vuik, 2005, 2006]
Grid-independent convergence

Illustration: \[-(\omega/c)^2 u - \nabla \cdot \nabla u(x) = b, \ x \in [0, 1]^2\]

[Iterative method (Bi-CGSTAB): ILU(0.01) preconditioner and MG on M]

Convergence:
- depends linearly on \( \omega \)
- independent of \( h \)

(MG can be done implicitly (no coarse-grid and interpolation matrices stored)
 Well parallellizable)
Grid-independent convergence

Realistic example: The 2D Marmousi problem

In-core, F77 GNU Compiler + MPI

A Pentium 4, 1GB RAM

- Convergence linearly dependent of $\omega$.
- Good scalability

A parallel machine
Grid-independent convergence

The 3D salt dome velocity model

SGI Altix 3700, SARA Amsterdam, NL

Riyanti, Kononov, E., Vuik, Oosterlee, Plessix, Mulder, 2006

With direct solver:

F = 10 Hz, N = 50 Mio
For LU \sim 9.6 \text{ GMem Unit}
Towards frequency-independent convergence

Problem: small eigenvalues $O(k^{-1})$

Solution: Projection $P := I - (A - \lambda_N I)ZE^{-1}Y^T$, $Z, Y : \mathbb{R}^m \rightarrow \mathbb{R}^N$, $m \ll N$.

Then solve: $PAM^{-1}\tilde{u} = Pb$, $u = M^{-1}\tilde{u}$.

$\kappa(PAM^{-1}) \rightarrow 1$: convergence independent of everything!

[E., Nabben, 2006, 2007]
Towards frequency-independent convergence

Illustration: \(-\left(\frac{\omega}{c}\right)^2 u - \nabla \cdot \nabla u(x) = b, \ x \in [0, 1]^2, \ \omega/c = k = constant\)

\(PAM^{-1}\) implementation: Multilevel Krylov method (MK-MG)

On a Pentium 4, 1 GB RAM

Number of gridpoints

Iterations

CPU Time

In progress: the 2D Marmousi problem with \(f = 1 - 60\) Hz [with Tim Lin]
Future work

- Implementation of the algorithm in SLIMpy environment
- Compressed adjoint-state method:
  (i) compressively sampled frequency, shots and receivers
    Traditional approach: $\Delta f$ is from “relaxing” Nyquist criteria [Plessix & Mulder, Sirgue & Pratt]
    New approach: compressive sampling
  (ii) compressed image
    $\rightarrow SA[m]S^T, m \in \mathcal{M}$
    $\rightarrow A[S^Tx], S^Tx \in \mathcal{M}$
- Iterative method for multiple right-hand sides (multiple shots)
  $\rightarrow A[m]U = B, B = [b_1 \ldots b_s]$