Scalable multi-level deflation preconditioning for highly indefinite time-harmonic waves

Dwarka, Vandana ; Vuik, Cornelis

DOI
10.1016/j.jcp.2022.111327

Publication date
2022

Document Version
Final published version

Published in
Journal of Computational Physics

Citation (APA)
Dwarka, V., & Vuik, C. (2022). Scalable multi-level deflation preconditioning for highly indefinite time-harmonic waves. *Journal of Computational Physics*, *469*, 1-28. Article 111327. https://doi.org/10.1016/j.jcp.2022.111327

Journal of Computational Physics 469 (2022) 111327

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Scalable multi-level deflation preconditioning for highly indefinite time-harmonic waves

Vandana Dwarka *, Cornelis Vuik

Delft University of Technology, Numerical Analysis, van Mourik Broekmanweg 6, 2628 XE, Delft, the Netherlands

ARTICLE INFO

Article history:
Received 25 July 2021
Received in revised form 17 May 2022
Accepted 19 May 2022
Available online 1 August 2022

Keywords:
Helmholtz equation
Multilevel
Indefinite
Deflation

ABSTRACT

Recent research efforts aimed at iteratively solving time-harmonic waves have focused on a broad range of techniques to accelerate convergence. In particular, for the famous Helmholtz equation, deflation techniques have been studied to accelerate the convergence of Krylov subspace methods. In this work, we extend the two-level deflation method to a multilevel deflation method for (heterogeneous) Helmholtz and elastic wave problems. By using higher-order deflation vectors, we show that up to the level where the coarse-grid linear systems remain indefinite, the near-zero eigenvalues of the these coarse-grid operators remain aligned with the near-zero eigenvalues of the fine-grid operator, keeping the spectrum of the preconditioned system away from the origin. Combining this with the well-known CSLP-preconditioner, we obtain a scalable solver for the highly indefinite linear systems. This can be attributed to a close to wave number independent convergence and an optimal use of the CSLP-preconditioner on the indefinite levels. There, we approximate the CSLP-preconditioner, while allowing the complex shift to be small, by using inner Bi-CGSTAB iterations instead of a multigrid F-cycle. The proposed method shows very promising results for the more challenging two- and three-dimensional heterogeneous time-harmonic wave problems.

1. Introduction

The Helmholtz equation has puzzled the minds of many mathematicians and numerical analysts throughout the years. Its wide application, ranging from seismology to medical tomography, has kept its relevance even till this day. As a result, many efforts have and are still being been rendered in order to obtain accurate and computationally feasible solutions.

A large branch within this research has focused on developing preconditioners, such as the (Complex) Shifted Laplacian [1–4]. In order to apply the preconditioner, one multigrid cycle is used to approximate its inverse. The latter serves as an alternative to using multigrid as a stand-alone solver as the method is generally known to diverge for the Helmholtz equation once coarser levels are reached [5]. Some works have focused on obtaining a stand-alone multigrid solver [6–9], with success for either practical wavenumbers and/or one-dimensional model problems.

* Corresponding author.
E-mail address: v.n.s.r.dwarka@tudelft.nl (V. Dwarka).

https://doi.org/10.1016/j.jcp.2022.111327

A recent and promising branch of research has combined its efforts towards preconditioning techniques based on domain decomposition methods applied to the corresponding (shifted) problem [10]. These methods split the computational domain in subdomains and solve a local subproblem of smaller dimension using a direct method [11–15]. The performance of these preconditioners depends on the accuracy of the transmission conditions, which currently is robust for constant wave number model problems [16,17]. While the domain decomposition preconditioners have resulted in a reduced number of iterations and higher computational efficiency by exploiting parallelization strategies, the number of iterations still grows with the wavenumber k.

As a result, some have studied the use of deflation techniques (combined with the CSLP-preconditioner) in order to accelerate the convergence of the Krylov subspace method, which we will denote DEF [18–20]. Incorporating the deflation preconditioner has improved the convergence, but taxed the efficiency in terms of memory and computational cost. For a two-level deflation preconditioner, the direct solve on the second level takes up most of the computational power and memory. Consequently, multilevel variants of the two-level method have been proposed in order to counter this effect [19,21]. A multilevel extension replaces the direct solve in the two-level method by applying a similar two-level extension recursively combined with an outer Flexible GMRES (FGMRES) solver.

In both variants, however, the number of iterations still slowly grows with the wave number k. In this work, we build on our recent work from [22] where we developed and tested a two-level deflation preconditioner which rendered close to wavenumber independent convergence for large wavenumbers in all spatial dimensions. We will refer to this method as the Adapted Deflation Preconditioner (ADP), where the adaption is realized through the use of higher-order interpolation polynomials. A natural question which arises is whether we can extend the wavenumber independent convergence to a multilevel setting, thereby combining both the gain in computational efficiency with our previous scalability results.

The structure of this paper is as follows. We start by introducing our model problems in Section 2. We then discuss the deflated Krylov methods and the multilevel algorithm in Section 3. We then proceed by extensively developing theory for the multilevel deflation operator in Section 4. We perform Rigorous Fourier Analysis (RFA) by block-diagonalizing the resulting operators and inspecting the spectral properties. Finally we present numerical results for benchmark problems in Section 5.

2. Problem description

We start by focusing on a one-dimensional mathematical model using a constant wave number $k > 0$

$$\begin{aligned} -\frac{d^2u}{dx^2} - k^2 u &= \delta(x - x'), x \in \Omega = [0, L] \subset \mathbb{R}, \\ u(0) = 0, u(L) &= 0, \end{aligned} \tag{1}$$

where x' denotes the location of the point source. We will refer to this model problem as MP 1-A. This simple model problem will allow us to develop the theory for the constant wave number case, as finding robust multilevel solvers for this case is still an active and current research area. To allow for more practical examples, we introduce MP 1-B as the model problem where Sommerfeld radiation conditions have been implemented. In this case, the boundary conditions become

$$\left(\frac{\partial}{\partial \mathbf{n}} - \sqrt{-1}k\right) u(x) = 0, x \in \partial[0, L].$$

If we define $h = \frac{1}{n}$, where n is chosen according to $kh = \frac{2\pi}{c}$, where c is the number of grid points per wavelength, then discretization on the unit interval using second order finite differences leads to

$$\frac{-u_{j-1} + 2u_j - u_{j+1}}{h^2} - k^2 u_j = f_j, j = 1, 2, \ldots, n.$$

Lexicographic ordering leads to the following linear system and eigenvalues for MP 1-A with indices $j = 1, 2, \ldots n$

$$\begin{aligned} Au &= \frac{1}{h^2}\text{tridiag}[-1 \;\; 2 - k^2h^2 \;\; -1]u = f, \\ \hat{\lambda}^j &= \frac{1}{h^2}\left(2 - 2\cos(j\pi h)\right) - k^2. \end{aligned} \tag{2}$$

Similarly, we define the 2-D and 3-D versions of model problem MP 1-B as above Eq. (1). The discretization using second order finite differences goes accordingly for higher dimensions with the needed alterations at the boundary when using Sommerfeld conditions. In Section 5, we will perform numerical experiments for more heterogeneous two- and three-dimensional model problems.

Table 1
Upper bound to number of non-zero elements per column of $A \in \mathbb{R}^{n\times n}, E \in \mathbb{R}^{m\times m}, Z \in \mathbb{R}^{n\times m}$ and $Z^T \in \mathbb{R}^{m\times n}$.

Operator	Linear			Quadratic		
	1D	2D	3D	1D	2D	3D
A	3	5	7	3	5	7
E	3	3^2	3^3	7	7^2	7^3
Z	3	3^2	3^3	5	5^2	5^3
Z^T	2	2^2	2^3	3	3^2	3^3
AZ	5	5^2	5^3	9	9^2	9^3

3. Deflated Krylov methods

We start by briefly explaining the two-level deflation preconditioning technique to solve the resulting linear system. We then proceed by extending the two-level method recursively to a multilevel Krylov method.

3.1. Two-level deflation

For a linear system $Au = f$ we construct the deflation preconditioner P where the column space of Z is used as the deflation subspace. The aim of including a deflation preconditioner is to project the unwanted near-zero eigenvalues to zero such that the convergence of the underlying Krylov subspace method can be accelerated. For the two-level method, the preconditioner P in fact is a projection operator. As for the deflation matrix, Z can be interpreted as interpolating from the coarse grid to the fine grid.

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

The inexact inversion which will be used for a multi-level approach requires the addition of an extra term Q in order to prevent synthetic close-to-zero eigenvalues from obstructing the convergence of the Krylov solver [23,21,24]. Without the addition of Q, the deflation operator is sensitive to rounding errors stemming from the inexact inversion of E. In [22], we used higher-order Bezier curves to construct Z. Using these higher-order polynomials, the prolongation and restriction operator act on a grid function as follows

$$Z\,[u_{2h}]_i = \left\{ \begin{array}{cc} \frac{1}{8}\left([u_{2h}]_{(i-2)/2} + 6\,[u_{2h}]_{(i)/2} + [u_{2h}]_{(i+2)/2}\right) & i \text{ is even,} \\ \frac{1}{2}\left([u_{2h}]_{(i-1)/2} + [u_{2h}]_{(i+1)/2}\right) & i \text{ is odd,} \end{array} \right\}, \tag{3}$$

for $i = 1, \ldots, n-1$ and for $i = 1, \ldots, \frac{n}{2}$. To obtain even better convergence, the CSLP-preconditioner was included, which is given by

$$M = L - (\beta_1 + \sqrt{-1}\beta_2)k^2 I,$$

where $(\beta_1, \beta_2) \in [0, 1]$ and L is the discretized Poisson equation. The system to be solved becomes $M^{-1}PAu = M^{-1}Pf$. By allowing higher-order interpolation schemes, the near-zero eigenspace of the fine- and coarse-grid coefficient matrix remains perfectly aligned. As a result, the smallest eigenvalue in magnitude of both A and E is located at the same index. This prevents the eigenvalues of the deflated system from shifting towards the origin. While the method provides close to wavenumber independent convergence in one- and two-dimensions for fairly large wavenumbers $k = 10^6$ (1D) and $k = 10^3$ (2D), it requires the exact solve of the coarse-grid coefficient matrix E, adding to the computational complexity in 3D.

In Algorithm 1, we present the two-level deflated FGMRES algorithm, where we use the following abbreviations: MV (matrix vector product), MM (matrix matrix product), ES (exact solve), VU (vector update), AS (approximate solve), DP (dot product). We moreover let C_{it} denote the number of constant iterations. The motivation for using FGMRES lies in the recursive process which will be applied in order to avoid having to solve the coarse-grid system on the second level exactly. We have split the pseudo-code into two parts. The blue section contains the part where the deflation preconditioner is applied. All matrix vector multiplications within the blue section are sparse. The pink section is the general GMRES-process. Furthermore, Table 1 contains the number of non-zero elements per column of the operators involved. These can be used to quantify the flops for sparse matrix-matrix and sparse matrix-vector products. Note that in the context of the multi-level algorithm, the number of non-zeros after the second level remains the same. We include the largest dimension-dependent constants for the leading order time complexity term. For example, for the 2D matrix-vector product we take $7n$ as the leading term instead of $5n$. As a result, we obtain a strict upperbound for the costs involved, even for 1D and 2D problems and the costs in practice will be less.

Algorithm 1: Two-level deflation FGMRES. (For interpretation of the colors in the algorithm(s), the reader is referred to the web version of this article.)

Initialization:
Choose u_0 initial guess and dimension k of the Krylov subspaces.
Define $(k+1) \times k$ $\bar{H}_k$ and initialize to zero.
Arnoldi process:
$r_0 = f - Au_0$, $\beta = \|r_0\|_2$, $v_1 = r_0/\beta$.
for $j = 1, 2, ...k$ **do**
  $\hat{v} = Z^T v_j$ ▷ MVP - $5^d n$
  $\tilde{v} = E^{-1}\hat{v}$ ▷ ES - m^2
  $t = Z\tilde{v}$ ▷ MVP - $3^d m$
  $s = At$ ▷ MVP - $7n$
  $\tilde{r} = v_j - s$ ▷ VU - n
  $r = M^{-1}\tilde{r}$ ▷ AS - $C_{it}n$
  $x_j = r + t$ ▷ VU - n
  $w = Ax_j$ ▷ MVP - $7n$
  for $i = 1, 2, ...j$ **do**
    $h_{i,j} = (w, v_j)$ ▷ DP - jn
    $w = w - h_{i,j}v_i$ ▷ VU - jn
  end
  Compute $h_{j+1,j} = \|w\|_2$ and $v_{j+1} = w/h_{j+1,j}$. ▷ MVP - n
  Define $X_k = [x_1, x_2, \ldots, x_k]$ and $\bar{H}_k = \{h_{i,j}\}_{1 \leqslant i \leqslant j+1, 1 \leqslant j \leqslant k}$ ▷ MVP - n
end
Form approximate solution:
Compute $u_k = u_0 + X_k y_k$ where $y_k = \arg\min_y \|\beta e_1 - \bar{H}_k y\|_2$.
Restart:
If satisfied stop, else set $u_0 \leftarrow u_k$ and repeat Arnoldi process.

Considering Algorithm 1, the vector update $x_j = r + t$ is split in two parts. r contains the application of $I - AQ$ to the vector v_j and then lastly applies the preconditioner M to obtain r. The vector t contains the part where we add Q to $I - AQ$ in order to prevent synthetic near-zero eigenvalues due to rounding errors. Analyzing the costs of Algorithm 1, confirms that the dominant factor is $O(m^2)$. Furthermore, in order to mitigate the cost of the preconditioning step, one $(C_{it} = 1)$ multigrid F-cycle is generally applied in order to approximate the solution of the system $Mr = \tilde{r}$ [19]. When opting for this configuration, the shift β_2 has to be kept large enough for multigrid to converge [5,9,4]. Another option is by allowing a few GMRES-iterations to approximate the preconditioner. For example in the context of using multigrid as a preconditioner, the standard relaxation step is replaced by 10-40 GMRES-iterations, acting as a polynomial smoother. On each level the unstable Jacobi and Gauss-Seidel smoother are replaced by Krylov iterations [6,25,26].

3.2. Multilevel deflation

As mentioned previously, apart from the standard computational costs associated with the FGMRES-algorithm, the largest additional cost comes from solving the coarse-grid system exactly, which dominates with the factor $O(m^2)$. In order to circumvent this, we apply the two-level cycle recursively. Before we expand the two-level algorithm to the multi-level algorithm, a few remarks are in place. In this work we deploy five changes, apart from using Bezier interpolation polynomials as a basis for the deflation vectors.

First, application of the CSLP-preconditioner to the Helmholtz operator shifts the spectrum towards the complex plane and resolves the indefiniteness. On levels where the matrix becomes negative definite, we apply a Jacobi iteration using the diagonal matrix of the CSLP as the preconditioner M.

Second, the multilevel preconditioner is applied to A rather than AM^{-1}. This saves one matrix-vector product per level.

Third, while the use of the CSLP preconditioner together with a geometric multigrid method for approximate inversion works well for homogeneous problems, it is not suitable for heterogeneous problems with high contrasts [19,21]. As we are interested in heterogeneous problems and require only an approximate application of the preconditioner, we will perform Krylov subspace iterations to approximately invert the CSLP. As mentioned previously, this can be considered as applying a polynomial smoother in the context of multigrid, which damps both ends of the spectrum. We let C_{it} denote the constant for the maximum number of iterations. The number of Krylov subspace iterations as a smoother ranges from 5-40 for two-dimensional constant wavenumber model problems, where the stopping criterion results in the residual to be scaled with kh on each level [27,6,26]. In this work we use Bi-CGSTAB as the computational costs and memory do not grow with the number of iterations such as is the case with non-restarted GMRES. We moreover do not require convergence or set any tolerance dependent on the level. However, we set the maximum number of Bi-CGSTAB iterations at a constant times $\lceil n^{(l)} \rceil^{\frac{1}{4}}$, where $n^{(l)}$ denotes the problem size on level l where the linear system is still indefinite. Our motivation for doing so is twofold. Primarily we want to have the number of outer FGMRES iterations as small as possible while the wavenumber increases, as FGMRES becomes more computationally expensive when more iterations are needed. Second of all, we do not

require the residual to remain orthogonal to all previous components, we can use Bi-CGSTAB to achieve a smaller residual within the multilevel hierarchy without necessarily imposing that it is in fact the minimized residual.

Fourth, given that we are no longer using multigrid for the approximate inversion, the restrictions for choosing the complex shift can be lifted. Thus, we can take advantage of using a small shift which makes the preconditioner more similar to the original Helmholtz operator and keeps the property of lifting the indefiniteness at certain levels. As a result, we will be able to test our algorithm on heterogeneous models with highly varying contrast profiles.

Fifth, instead of allowing many iterations on each coarse level, we only allow one iteration on the coarser levels. Consequently, we obtain a V-cycle, which leads to a similar V-cycle structure from multigrid when taking $\gamma = 1$, see Fig. 1. The multilevel deflation algorithm is given below, where we used the number of non-zero elements from Table 1 to account for the dimension dependent constants for the sparse matrix-matrix and sparse matrix-vector products on subsequent levels.

Algorithm 2: Multilevel ADP implementation.

Initialization
Set $A^{(1)} = A, M^{(1)} = M, n^{(1)} = n$
for $l = 1, 2, ...m$ *the coarsest level* **do**
  Construct $Z^{(l,l+1)}$ and $Z^{(l,l+1)^T}$
  Construct $A^{(l+1)} = Z^{(l,l+1)^T} A^{(l)} Z^{(l,l+1)}$ ▷ MMP - $45^d \cdot n^{(l+1)}$
  Construct $M^{(l+1)} = Z^{(l,l+1)^T} M^{(l)} Z^{(l,l+1)}$ ▷ MMP - $45^d \cdot n^{(l+1)}$
end
Iterative stage
$l = 1$, $u_0^{(1)} = 0$
Solve $A^{(1)} u^{(1)} = b^{(1)}$ using Two-level Deflated FGMRES
$\hat{v}^{(2)} = Z^{(1,2)^T} v^{(1)}$ ▷ MVP - $5^d \cdot n^{(1)}$
if $l + 1 = m$ **then**
  Solve $\tilde{v}^{(2)} = A^{(2)^{-1}} \hat{v}^{(2)}$ exactly
else
  $l = l + 1$, $\tilde{v}_0^{(2)} = 0$
  Solve $A^{(2)} \tilde{v}^{(2)} = \hat{v}^{(2)}$ using Two-level Deflated FGMRES
  $\hat{v}^{(3)} = Z^{(2,3)^T} v^{(2)}$ ▷ MVP - $5^d \cdot n^{(2)}$
  if $l + 1 = m$ **then**
    Solve $\tilde{v}^{(3)} = A^{(3)^{-1}} \hat{v}^{(3)}$ exactly
  else
    $l = l + 1$, $\tilde{v}_0^{(3)} = 0$
    Solve $A^{(3)} \tilde{v}^{(3)} = \hat{v}^{(3)}$ using Two-level Deflated FGMRES
    $\hat{v}^{(4)} = Z^{(3,4)^T} v^{(3)}$ ▷ MVP - $5^d \cdot n^{(3)}$
    ⋮
    if $l + 1 = m$ **then**
      Solve $\tilde{v}^{(m)} = A^{(m)^{-1}} \hat{v}^{(m)}$ exactly ▷ ES - $O(1)$
    end
    $t^{(m-1)} = Z^{(m-1,m)} \tilde{v}^{(m)}$ ▷ MVP - $3^d \cdot n^{(m)}$
    $s^{(m-1)} = A^{(m-1)} t^{(m-1)}$ ▷ MVP - $7^d \cdot n^{(m-1)}$
    $\tilde{r}^{(m-1)} = v^{m-1} - s^{(m-1)}$ ▷ VU - $n^{(m-1)}$
    $r^{(m-1)} = M^{(m-1)^{-1}} \tilde{r}^{(m-1)}$ ▷ AS - $n^{(m-1)}$
    $x^{(m-1)} = r^{(m-1)} + t^{(m-1)}$ ▷ VU - $n^{(m-1)}$
    $w^{(m-1)} = A^{(m-1)} x^{(m-1)}$ ▷ MVP - $7^d \cdot n^{(m-1)}$
    ⋮
  end
  $t^{(2)} = Z^{(2,3)} \tilde{v}^{(3)}$ ▷ MVP - $3^d \cdot n^{(3)}$
  $s^{(2)} = A^{(2)} t^{(2)}$ ▷ MVP - $7^d \cdot n^{(2)}$
  $\tilde{r}^{(2)} = v^3 - s^{(2)}$ ▷ VU - $n^{(2)}$
  $r^{(2)} = M^{(2)^{-1}} \tilde{r}^{(2)}$ ▷ AS - $21(C_{it} n^{(2)^{\frac{1}{4}}}) n^{(2)}$
  $x^{(2)} = r^{(2)} + t^{(2)}$ ▷ VU - $n^{(2)}$
  $w^{(2)} = A^{(2)} x^{(2)}$ ▷ MVP - $7^d \cdot n^{(2)}$
end
$t^{(1)} = Z^{(1,2)} \tilde{v}^{(2)}$ ▷ MVP - $3^d \cdot n^{(2)}$
$s^{(1)} = A^{(1)} t^{(1)}$ ▷ MVP - $7^d \cdot n^{(1)}$
$\tilde{r}^{(1)} = v^1 - s^{(1)}$ ▷ VU - $n^{(1)}$
$r^{(1)} = M^{(1)^{-1}} \tilde{r}^{(1)}$ ▷ AS - $21(C_{it} n^{(1)^{\frac{1}{4}}}) n^{(1)}$
$x^{(1)} = r^{(1)} + t^{(1)}$ ▷ VU - $n^{(1)}$
$w^{(1)} = A^{(1)} x^{(1)}$ ▷ MVP - $7^d \cdot n^{(1)}$

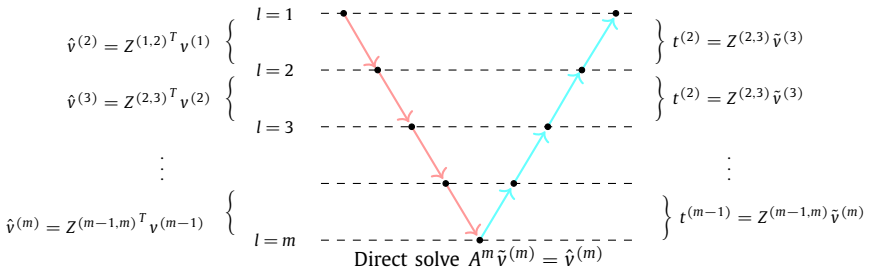


Fig. 1. V-cycle deflated FGMRES. The pink arrows represent the coarsening. The blue arrows represent the prolongation. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

A schematic representation is given in Fig. 1.
Using the above, we can formulate upper bounds in terms of FLOPs for the complete algorithm.

Theorem 3.1 *(Multilevel deflation upper bound number of operations). For $l = 1$, set $n^{(1)} = n$ and let $n^{(l)} = 2^{-ld} n$ for $l > 1$, where d denotes the dimension. Assume the following holds for $l \geqslant 1$*

- *Restriction* $\hat{v}^{(l+1)} := Z^{(l,l+1)^T} \hat{v}^{(l)}$

$$\text{flops} \leqslant c_r 5^d n^{(l)}$$

- *Prolongation* $t^{(l)} := Z^{(l,l+1)} \tilde{v}^{(l+1)}$

$$\text{flops} \leqslant c_p 3^d n^{(l+1)}$$

- *Krylov Smoothing* $r^{(l)} := {M^{(l)}}^{-1} \tilde{r}^{(l)}$ *when* $l < 3$

$$\text{flops} \leqslant c_k 7^d {n^{(l)}}^{1+\frac{1}{4}}$$

- *Jacobi Smoothing* $r^{(l)} := {M^{(l)}}^{-1} \tilde{r}^{(l)}$ *when* $l \geqslant 3$

$$\text{flops} \leqslant c_j 7^d n^{(l)}$$

- *Matrix vector product:=* $w^{(l)} = A^{(l)} x^{(l)}$

$$\text{flops} \leqslant c_v 7^d n^{(l)}$$

- *Coarse-grid solve:=* ${A^{(m)}}^{-1} \hat{v}^{(m)}$

$$\text{flops} \leqslant c_0$$

Then

$$\textit{total flops} \leqslant C_d O(n^{1+\frac{1}{4}}),$$

where C_d is a constant which only depends on the dimension d.

Proof. At each level l, after we have obtained $w^{(l)}$, we proceed with the Arnoldi process (see pink section, Algorithm 1), which is already $O(n^l)$ given that the maximum number of FGMRES iterations at each level is set at one. We thus obtain the following upper bound of the additional costs occurred with the preconditioning step

$$\begin{aligned}
\text{flops} &\leqslant c_0 + \left(c_r 5^d + C_k 7^d {n^{(l)}}^{\frac{1}{4}} + c_v 7^d\right) n^{(l)} + c_p 3^d n^{(l+1)}, \\
&+ \left(c_r 5^d + C_k 7^d {n^{(l+1)}}^{\frac{1}{4}} + c_v 7^d\right) n^{(l+1)} + c_p 3^d n^{(l+2)}, \\
&+ \ldots \left(c_r 5^d + C_k 7^d + c_v 7^d\right) n^{(m-1)} + c_p 3^d n^{(m)}, \\
&= c_0 + \left(c_r 5^d + C_k 7^d {n^{(1)}}^{\frac{1}{4}} + c_v 7^d\right) n^{(1)} + C_k 7^d {n^{(2)}}^{\frac{1}{4}} n^{(2)},
\end{aligned}$$

$$
\begin{aligned}
&+ \left(c_r 5^d + C_j 7^d + c_v 7^d + c_p 3^d\right) \sum_{l=2}^{m-1} 2^{-dl} n^{(1)} + c_p 3^d n^{(m)}, \\
&\leqslant c_0 + \left(c_r 5^d + C_k 7^d {n^{(1)}}^{\frac{1}{4}} + c_v 7^d\right) n^{(1)} + C_k 7^d {n^{(2)}}^{\frac{1}{4}} n^{(2)}, \\
&+ \left(c_r 5^d + C_j 7^d + c_v 7^d + c_p 3^d\right) \sum_{l=2}^{m-1} 2^{-l} n^{(1)} + c_p 3^d n^{(m)}, \\
&< c_0 + \left(c_r 5^d + C_k 7^d {n^{(1)}}^{\frac{1}{4}} + \frac{1}{2} C_k 7^d {n^{(1)}}^{\frac{1}{4}} + c_v 7^d\right) n^{(1)} + c_p 3^d n^{(m)}, \\
&+ \frac{1}{2}\left(c_r 5^d + C_a 7^d + c_v 7^d\right) n^{(1)}, \\
&< C 7^d n^{(1+\frac{1}{4})} = O(n^{(1+\frac{1}{4})}),
\end{aligned}
$$

where we used that $n^{(1)} = n$ and the fact that $\sum_{l=2}^{m-1} 2^{-l} < \frac{1}{2}$. The upper bound to the total computational costs is constructed with respect to the iterative stage and already accounts for the costs of the matrix vector multiplication in FGMRES. It in fact bounds the total cost of the multilevel extension of the blue section in Algorithm 1. Overall, the algorithm runs in $O({n^{(1)}}^{1+\frac{1}{4}})$ time complexity. However, a few points need further explanation and discussion.

The construction of the coarse-grid systems on each level l requires two sparse matrix-matrix multiplications. While the maximum number of non-zeros along each column remains constant with respect to $n^{(1)}$ (see Table 1), it results in a constant of 45^d for the matrix-matrix multiplication to construct $A^{(l>1)}$. While this may seem expensive, it already pays-off for very large and highly indefinite 2D and 3D problems, as the constant is independent of the fine-grid problem size $n^{(1)}$. We will illustrate this through our numerical experiments in Section 5.

Moreover, as mentioned previously, the multilevel preconditioner is applied to A rather than AM^{-1}. By using the 'First Deflate, then Precondition' method, we save one extra matrix-matrix product, one matrix-vector product and one extra application of the preconditioning step [28].

Finally, we restrict the number of FGMRES iterations on each level to one, whereas a sequence of $(8, 2, 1)$ and $(6, 2, 2)$ iterations are used [19,28,21]. For example $(8, 2, 1)$ denotes, 8 iterations on level $l = 2$, 2 iterations on $l = 3$ and 1 iteration on all levels $l > 3$. Thus, on the finest level $n^{(1)}$, the largest cost related to the matrix-vector product during the iterative stage is $O(8 \cdot 3^d n^{(1)})$ compared to $O(7^d n^{(1)})$. While the dimension-dependent constant 7^d is approximately 1.5 times larger than $8 \cdot 3^d$ for $d = 3$, the significant reduction in the number of outer iterations provides an advantageous leverage. □

4. Inscalability

In this section we will extend the theoretical results of the two-level ADP-scheme to a multilevel setting for MP 1-A. Given that the coefficient matrix remains normal, spectral analysis can be performed to assess the convergence behavior. We have provided a detailed summary of the literature as regards the role of the eigenvalues when the matrix is non-normal in [22].

4.1. Multilevel mapping

In order to develop theory for the multi-level ADP-scheme from the two-level ADP-scheme, we need expressions for the nested or composite mappings between the fine and coarse spaces. Similar to our approach for the two-level method in [22], we start with the linear case and extend it to the quadratic case. In Theorem 4.1 we start by deriving analytical expressions for the actions of the intergrid transfer operators on eigenvectors of each respective coarse space for the linear case, whereas Corollary 4.1.1 contains the expressions for the quadratic case. The detailed proof for the linear case can be found in Appendix A. Here we provide a brief outline of the proof for the sake of completeness.

Theorem 4.1 *(Multilevel prolongation and restriction (linear)). Let Z_m be the $n_{m-1} \times n_m$ prolongation matrix based on linear interpolation for $m = 1, 2, \ldots m_{\max}$, with $n_m = \frac{n}{2^m}$. If we define $v_m^j = \sin(2^m h i \pi\, j)$, and $v_m^{j'} = \sin(2^m h i \pi\,(n_m + 1 - j))$, where on the finest level we have $m = 0$, then there exist constants C_1^j and C_2^j depending on h such that restriction operator maps the eigenvectors to*

$$
\prod_{l=m}^{1} Z_l^T v_0^j = C_1^j v_m^j, \;\; j = 1, 2, \ldots, n_m \text{ and } \prod_{l=m}^{1} Z_l^T v_0^{j'} = C_2^j v_m^j, \;\; j = 1, 2, \ldots, n_m,
$$

where $C_1^j = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m} \left(1 + \cos(j\pi 2^{l-1} h)\right)$ and $C_2^j = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m} \left(\cos(j\pi 2^{l-1} h) - 1\right)$. Similarly, the prolongation operator maps the eigenvectors to

$$\prod_{l=1}^{l} Z_l[v_m]_i = C_1^j[v_0^j]_i, \text{ for } i \text{ is odd. and } \prod_{l=1}^{l} Z_l[v_m]_i = C_2^j[v_0^j]_i, \text{ for } i \text{ is even.}$$

Finally, if we let $B_m = \prod_{l=1}^{m} Z_l \prod_{l=m}^{1} Z_l^T$ *and* $\hat{B}_m = Z_m Z_m^T$ *for* $m = 1, 2, \ldots, m_{\max}$, *then* B_m *has dimension* n_0 *with* n_m *non-zero eigenvalues.*

Proof. The extended proof is given in Appendix A. We will briefly give an outline of the proof here. We start by defining the mapping operators and the respective vector spaces and their bases to which they are applied. This allows us to move between fine and coarse spaces. Then we continue by showing the action of the restriction operator on the basis for these respective vector spaces. To keep an overview of what is happening between the vector spaces on an abstract level, we use both the analytical operator and their matrix representations in the proof. We then repeat this for the prolongation operator. Once we have analytical expressions for these nested operators, we show that the kernel and range of the composite mapping consisting of the restriction and prolongation operator span a subspace containing the eigenvectors. We use this to show that the eigenvalues of B_m are related to the eigenvalues of $\hat{B}_m$. □

We similarly extend the multilevel operators for the higher-order deflation vectors. This is given in Corollary 4.1.1 and follows naturally from the linear case.

Corollary 4.1.1 *(Multilevel prolongation and restriction (quadratic)). Let* Z_m *be the* $n_{m-1} \times n_m$ *prolongation matrix based on rational Bezier curves for* $m = 1, 2, \ldots m_{\max}$, *with* $n_m = \frac{n}{2^m}$. *If we define* $v_m^j = \sin(2^m h i\pi j)$, *and* $v_m^{j'} = \sin(2^m h i\pi (n_m + 1 - j))$, *where on the finest level we have* $m = 0$. *Then there exist constants* C_1^j *and* C_2^j *depending on* h *such that the restriction operator maps the eigenvectors to*

$$\prod_{l=m}^{1} Z_l^T v_0^j = C_1^j v_m^j, \ j = 1, 2, \ldots, n_m, \text{ and } \prod_{l=m}^{1} Z_l^T v_0^{j'} = C_2^j v_m^j, \ j = 1, 2, \ldots, n_m,$$

where $C_1^j = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m} C_{1,lh}^j$ *and* $C_2^j = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m} C_{2,lh}^j$. *Similarly, the prolongation operator maps the eigenvectors to*

$$\prod_{l=1}^{l} Z_l[v_m]_i = C_1^j[v_0^j]_i, \ i \text{ is odd. and } \prod_{l=1}^{l} Z_l[v_m]_i = C_2^j[v_0^j]_i, \ i \text{ is even.}$$

Finally, if we let $B_m = \prod_{l=1}^{m} Z_l \prod_{l=m}^{1} Z_l^T$ *and* $\hat{B}_m = Z_m Z_m^T$ *for* $m = 1, 2, \ldots, m_{\max}$, *then* B_m *has dimension* n_0 *with* n_m *non-zero eigenvalues.*

Proof. The proof is exactly the same as the proof of Theorem 4.1, however we now have

$$C_{1,mh}^j = \left(\cos(j\pi 2^m h) + \cos(j\pi 2^{m+1} h)\frac{1}{4} + \frac{3}{4}\right),$$
$$C_{2,mh}^j = \left(\cos(j\pi 2^m h) - \cos(j\pi 2^{m+1} h)\frac{1}{4} - \frac{3}{4}\right).$$

For a detailed proof of deriving $C_{1,mh}^j$ and $C_{2,mh}^j$ see [22]. The statement is obtained by substituting these coefficients into the proof of Theorem 4.1. □

Using this result we can approximate the location where the near-zero eigenvalues of the coarse-grid matrices are located. This is important as we only want to apply the smoother on levels where it is needed. We start by denoting the coarse grid linear systems by E_m and we set $E_0 = A$, where A is the fine grid linear matrix. Analytical expressions for the location of the smallest eigenvalue are found in the following corollary.

Corollary 4.1.2 *(Coarse near-zero eigenvalues). Let* Z_m *be the* $n_{m-1} \times n_m$ *prolongation matrix for* $m = 0, 1, 2, \ldots m_{\max}$, *with* $n_m = \frac{n}{2^m}$. *We define the symmetric coarse-grid coefficient matrix* $E_m = \prod_{l=m}^{1} Z_l^T A \prod_{l=1}^{m} Z_m$. *If we let* $[v_m^j]_i = \sin(2^m h i\pi j)$ *be the eigenvectors of* E_m, *where for* $m = 0$ *we have the finest level, then* $\exists \tilde{m}$: *for* $m > \tilde{m}$ E_m *is negative definite. For* $m \leqslant \tilde{m}$ E_m *is indefinite.*

Proof. Let $\Lambda(A)$ denote the $n_0 \times n_0$ diagonal matrix containing the eigenvalues of A, then using Theorem 4.1 for each i, either odd or even, we have

$$\lim_{h\to 0} \left|E_m[v_m^j]_i\right| \leqslant \lim_{h\to 0} \left|\prod_{l=m}^{1} Z_l^T \Lambda(A) \prod_{l=1}^{m} Z_l[v_m^j]_i\right| \leqslant \lim_{h\to 0} \left|\lambda_A^j (C_1^j)^2 [v_m^j]_i\right| \leqslant 4^m \left|\lambda_A^j [v_m^j]_i\right|,$$

where we used that by definition of C_1^j and C_2^j, for all j we have $\left|C_1^j C_2^j\right| \leqslant \left|(C_1^j)^2\right| \leqslant 4^m$. Note that in case of i is even, we would have $C_1^j C_2^j$ instead of $C_1^{j^2}$. Thus, in the limit as h goes to zero, we can bound the expression for $\lambda_{E_m}^j$ from above by $\left|\lambda_{E_m}^j\right| \leqslant 4^m \lambda_A^j$ for each j. Now to find a bound for the smallest eigenvalue in magnitude of E_m, we need to minimize the right-hand side of the upper-inequality over all indices j. This is achieved at $j = j_{\min}$, corresponding to the smallest eigenvalue in magnitude of A as this eigenvalue is the closest eigenvalue to zero. We thus have $\left|\lambda_{E_m}^{j_{\min}}\right| \leqslant 4^m \lambda_A^{j_{\min}}$. We now need to find the level m at which the matrix E_m becomes negative definite. Recall that

$$j_{\min} = \left\lfloor \frac{\cos^{-1}\left(\frac{1-k^2h^2}{2}\right)}{\pi h} \right\rfloor = \left\lfloor \frac{n\cos^{-1}\left(\frac{1-k^2h^2}{2}\right)}{\pi} \right\rfloor.$$

Therefore, to find the level $\tilde{m}$ which still contains index $j_{\min}$, for $j = 1, 2, \ldots n_m$, we have to find $m : n_m = \frac{n}{2^m} > j_{\min}$. Note $j_{\min}$ is unaffected by h as h goes to zero and thus we can assess how many times $j_{\min}$ fits into n. Additionally, coarsening leads to the problem size being halved for each m, and thus need to divide by 2 as well.

$$\left\lfloor \frac{n}{2j_{\min}} \right\rfloor = \left\lfloor \frac{\cos^{-1}\left(\frac{1-k^2h^2}{2}\right)}{2\pi} \right\rfloor = \tilde{m}.$$

Consequently, for $m > \tilde{m}$, $j_{\min}$ is no longer within the range of n_m. Therefore, all eigenvalues of $E_{m>\tilde{m}}$ for $j = 1, 2, \ldots n_{m>\tilde{m}} \leqslant j_{\min}$ must have the same sign, due to the fact that $\lambda_A^{j_{\min}}$ is an upper bound and the only eigenvalue of A where a sign-change can occur. □

Corollary 4.1.2 shows that for $m \leqslant \tilde{m}$, the resulting coarse-grid coefficient matrices E_m are indefinite. Thus, on these subsequent levels, it is important that the near-zero eigenvalues are reduced and aligned in coherence with the fine-grid level. In order to analytically assert this, we proceed by defining the multilevel deflation operator and block-diagonalizing it using a similar basis as we used for the two-level ADP scheme. This will allow us to perform spectral analysis of the multilevel deflation operator as the latter reduces to applying the two-level ADP scheme recursively.

4.2. Block-diagonal systems

Using the matrices Z_m and Z_m^T to denote the prolongation and restriction operator on level m, and using the theory developed so far, we can construct similar analytical expressions for the eigenvalues of the preconditioner applied to the coefficient matrix. We will perform the analysis for MP 1-A. Taking $E_0 = A$, we define the $n \times n$ projection operator $P_{h,m}$ to be

$$P_{h,m} = I - AQ_m, \text{ where } Q_m = \prod_{l=1}^{m} Z_l E_m^{-1} \prod_{l=m}^{1} Z_l^T \text{ and } E_m = Z_m^T E_{m-1} Z_m, \tag{4}$$

$$P_m = I_m - E_m Q_m, \text{ where } Q_m = Z_m E_m^{-1} Z_m^T \text{ and } E_m = Z_m^T E_{m-1} Z_m \tag{5}$$

Note that this is equivalent to constructing P by solving E_m directly on the m-th level and then prolonging the inverse back to the fine grid in order to proxy the effect of having an approximate inversion of E_1 in the two-level method. We will refer to $P_{h,m}$ as the **global** multilevel deflation preconditioner and P_m as the **local** level deflation preconditioner.

4.2.1. Global system block-diagonalization

In order to extend the spectral analysis of the two-level ADP-scheme to a multilevel setting, we will use the bases and operators defined in the first part of the proof of Theorem 4.1, see Appendix A. To assist the reading of the proofs below, we briefly mention the basis and its reordering. For $n_m = \frac{n}{2^m}$, we rearranged the space spanned by the eigenvectors at each level m such that we obtain the following subspace

$$\mathcal{V}_m^j = \text{span}\{v_m^j, v_m^{n_m+1-j}\} \text{ and } V_{m+1}^j = \text{span}\{v_{m+1}^j\}$$

for $j = 1, 2, \ldots, n_{m+1}$. Note that the subspace $\mathcal{V}_m^j$ consists of two vectors and the subspace V_{m+1}^j consists of one vector. We furthermore have that both bases span $\mathbb{C}^{n_m}$ and $\mathbb{C}^{n_{m+1}}$ respectively as we can write

$$\mathbb{C}^{n_m} = \bigoplus_{j=1}^{n_{m+1}} \mathcal{V}_m^j \text{ and } \mathbb{C}^{n_{m+1}} = \bigoplus_{j=1}^{n_{m+1}} V_{m+1}^j,$$

and at each subsequent level $m+1$ we can always define an automorphism to re-order the basis V_{m+1} to obtain $\mathcal{V}_{m+1}$.

We start with Lemma 4.2.1, which will provide the building blocks to block-diagonalize Q_m by first block-diagonalizing B_m. This is equivalent to using the operators and expressions from Theorem 4.1 and writing them in matrix form using 2×2 blocks by evaluating their action on the underlying basis of eigenvectors.

Lemma 4.2.1 *(Block-diagonalization B_m). Let Z_m be the $n_{m-1} \times n_m$ interpolation matrix with $n_m = \frac{n}{2^m}$ for $m = 0, 1, 2, \ldots, m_{\max}$. Let $B_m = \prod_{l=m}^{1} Z_l \prod_{l=1}^{m} Z_l^T$ and $\hat{B}_m = Z_m Z_m^T$ for $m = 1, 2, \ldots, m_{\max}$. Defining the rearranged basis*

$$\mathcal{V}_m = \bigoplus_{j=1}^{n_{m+1}} \operatorname{span}\{v_m^j, v_m^{n_{m+1}+1-j}\},$$

where $v_m^j = [\sin(j\pi hi2^m)]_{i=1}^{n_m}$, the eigenvalues of B_m are given by

$$\lambda_{B_m}^j = \left(\frac{1}{2}\right)^m \prod_{l=m}^{1} \left((r_l^j)^2 + (p_l^j)^2\right),$$

where $r_l^j = C_{1,mh}^j$ and $p_l^j = C_{1,lh}^j$. Here $C_{2,lh}^j$ and $C_{2,mh}^j$ are either the linear or quadratic coefficients.

Proof. We can continue by using the results from Theorem 4.1. To keep the notation compact we let $r_m^j = C_{1,mh}^j$ and $p_m^j = C_{2,mh}^j$. We start with the case where $m = 1$. Using the basis $\mathcal{V}_0, V_1$, Z_1 and Z_1^T have the block form

$$[Z_1]_{V_1}^j = \begin{bmatrix} r_1^j \\ p_1^j \end{bmatrix}, \tag{6}$$

$$[Z_1^T]_{\mathcal{V}_0}^j = \begin{bmatrix} r_1^j & p_1^j \end{bmatrix}, \tag{7}$$

for $j = 1, 2, \ldots, n_1$. In block-diagonal form on we can write Z_1 as

$$\begin{bmatrix} \boxed{\begin{matrix} r_1^1 \\ p_1^1 \end{matrix}} & & & \mathbf{0} \\ & \boxed{\begin{matrix} r_1^2 \\ p_1^2 \end{matrix}} & & \\ & & \ddots & \\ \mathbf{0} & & & \boxed{\begin{matrix} r_1^{n_1} \\ p_1^{n_1} \end{matrix}} \end{bmatrix}$$

To block-diagonalize $\hat{B}_1$, we therefore multiply the respective blocks for each j

$$[Z_1[Z_1^T]_{\mathcal{V}_0}^j]_{V_1}^j = \begin{bmatrix} r_1^j \\ p_1^j \end{bmatrix} \begin{bmatrix} r_1^j & p_1^j \end{bmatrix} = \begin{bmatrix} (r_1^j)^2 & (r_1^j p_1^j) \\ (r_1^j p_1^j) & (p_1^j)^2 \end{bmatrix}.$$

Now, $\hat{B}_1$ has n_1 non-zero eigenvalues given by the trace of each respective block and n_1 zero eigenvalues, which was also discussed in the proof of Theorem 4.1. The non-zero eigenvalues are thus given by the 1×1 block $\lambda_{\hat{B}_1}^j = (r_1^j)^2 + (p_1^j)^2$ for $j = 1, 2, \ldots, n_1$ and $\hat{B}_1 = B_1$ has the block-diagonal form

$$[B_1]_{\mathcal{V}_0} = \left[\begin{array}{ccc|c} \boxed{\lambda_{\hat{B}_1}^1} & & & \\ & \ddots & & \mathbf{0} \\ & & \boxed{\lambda_{\hat{B}_1}^{n_1}} & \\ \hline & & & 0 \\ \mathbf{0} & & & \ddots \\ & & & 0 \end{array}\right].$$

We now take $m = 2$ and block-diagonalize $\hat{B}_2$. Using the same steps as above we have

$$[Z_2 Z_2^T]^j_{\mathcal{V}_1} = \begin{bmatrix} r_2^j \\ p_2^j \end{bmatrix} \begin{bmatrix} r_2^j & p_2^j \end{bmatrix} = \begin{bmatrix} (r_2^j)^2 & (r_2^j p_2^j) \\ (r_2^j p_2^j) & (p_2^j)^2 \end{bmatrix},$$

for $j = 1, 2, \ldots, n_2$. Computing the trace of each block gives $\lambda^j_{\hat{B}_2} = (r_2^j)^2 + (p_2^j)^2$ with block-diagonal form

$$[\Lambda(\hat{B}_2)]_{\mathcal{V}_1} = \left[\begin{array}{ccc|c} \boxed{\lambda^1_{\hat{B}_2}} & & & \\ & \ddots & & \mathbf{0} \\ & & \boxed{\lambda^{n_2}_{\hat{B}_2}} & \\ \hline & \mathbf{0} & & \begin{matrix} 0 & \\ & \ddots \\ & & 0 \end{matrix} \end{array}\right]. \tag{8}$$

Note that we have $n_2 = \frac{n}{4}$ zero and non-zero eigenvalues and the dimension of $\hat{B}_2$ is $n_1 \times n_1$. This is equivalent to having n_2 blocks of dimension 1×1 containing the non-zero eigenvalues and n_2 blocks, also with dimension 1×1 containing the zero eigenvalues. We now apply Z_1 to the left and Z_1^T to the right of Eq. (8), where we use the block-diagonal form of Z_1 and Z_1^T given by Eq. (6) and Eq. (7) respectively. Z_1 has n_1 blocks of dimension 2×1 and Z_1^T has n_1 blocks of dimension 1×2. Thus, Z_1 works on each non-zero 1×1 block of $\hat{B}_2$, and then Z_1^T is applied to the resulting 2×1 block. However, only the first n_2 blocks of $\Lambda(\hat{B}_2)$ contain non-zero terms as we can see from Eq. (8) and thus only the indices $j = 1, 2, \ldots n_2$ in Z_1 and Z_1^T lead to non-zero terms. Thus, for $j = 1, 2, \ldots, n_2$ we obtain $[\Lambda(B_2)]_{\mathcal{V}_0} = [\Lambda(Z_1 \hat{B}_2 Z_1^T)]_{\mathcal{V}_0}$, which is given by the following matrix representation

$$\left[\begin{array}{ccc} \boxed{\begin{matrix} r_1^1 \\ p_1^1 \end{matrix}} & & \mathbf{0} \\ & \boxed{\begin{matrix} r_1^2 \\ p_1^2 \end{matrix}} & \\ & \ddots & \\ \mathbf{0} & & \boxed{\begin{matrix} r_1^{n_1} \\ p_1^{n_1} \end{matrix}} \end{array}\right] \left[\begin{array}{ccc|c} \boxed{\lambda^1_{\hat{B}_2}} & & & \\ & \ddots & & \mathbf{0} \\ & & \boxed{\lambda^{n_2}_{\hat{B}_2}} & \\ \hline & \mathbf{0} & & \begin{matrix} 0 & \\ & \ddots \\ & & 0 \end{matrix} \end{array}\right] \left[\begin{array}{ccc} \boxed{\begin{matrix} r_1^1 & p_1^1 \end{matrix}} & & \mathbf{0} \\ & \boxed{\begin{matrix} r_1^2 & p_1^2 \end{matrix}} & \\ & \ddots & \\ \mathbf{0} & & \boxed{\begin{matrix} r_1^{n_1} & p_1^{n_1} \end{matrix}} \end{array}\right]$$

Thus, at the level of each respective j-th block we have

$$[\Lambda(B_2)]^j_{\mathcal{V}_0} = \begin{bmatrix} r_1^j \\ p_1^j \end{bmatrix} \lambda^j_{\hat{B}_2} \begin{bmatrix} r_1^j & p_1^j \end{bmatrix} = \lambda^j_{\hat{B}_2} \begin{bmatrix} (r_1^j)^2 & (r_1^j p_1^j) \\ (r_1^j p_1^j) & (p_2^j)^2 \end{bmatrix},$$

for $j = 1, 2, \ldots, n_2$. Computing the trace of each respective block gives

$$\lambda^j_{B_2} = \left((r_1^j)^2 + (p_1^j)^2\right)(\lambda^j_{\hat{B}_2}) = \left((r_1^j)^2 + (p_1^j)^2\right)\left((r_2^j)^2 + (p_2^j)^2\right). \tag{9}$$

Thus, we obtain the following block-diagonal form

$$[B_2]_{\mathcal{V}_0} = \left[\begin{array}{ccc|c} \boxed{\lambda^1_{B_2}} & & & \\ & \ddots & & \mathbf{0} \\ & & \boxed{\lambda^{n_2}_{B_2}} & \\ \hline & \mathbf{0} & & \begin{matrix} 0 & \\ & \ddots \\ & & 0 \end{matrix} \end{array}\right],$$

where $\lambda^j_{B_2}$ is given by Eq. (9). From here it is easy to see that successive application of Z_m and Z_m^T for $m > 2$ gives

$$[\Lambda(B_m)]^j_{\mathcal{V}_0} = \begin{bmatrix} \prod_{l=m-1}^1 r_l^j \\ \prod_{l=m-1}^1 p_l^j \end{bmatrix} \lambda^j_{\hat{B}_m} \begin{bmatrix} \prod_{l=m-1}^1 r_l^j & \prod_{l=m-1}^1 p_l^j \end{bmatrix},$$

for $j = 1, 2, \ldots, n_m$ with $\lambda^j_{B_m} = \prod_{l=m}^1 \left((r_l^j)^2 + (p_l^j)^2\right)$. □

Using the results from Lemma 4.2.1, we can block-diagonalize the operator Q_m, where m again denotes the level.

Theorem 4.2 *(Block-diagonalization Q_m). Let Z_m be the $n_{m-1} \times n_m$ interpolation matrix with $n_m = \frac{n}{2^m}$ for $m = 0, 1, 2, \ldots, m_{\max}$. We define the coarse linear system $E_m = Z_m^T E_{m-1} Z_m$ with $E_0 = A$. Let $B_m = \prod_{l=m}^{1} Z_l \prod_{l=1}^{m} Z_l^T$ and $\hat{B}_m = Z_m Z_m^T$ for $m = 1, 2, \ldots, m_{\max}$. Then using the basis $\mathcal{V}_0$ from Lemma 4.2.1, the eigenvalues of Q_m are given by*

$$[\Lambda(Q_m)]^j_{\mathcal{V}_0} = [\Lambda(\prod_{l=1}^{m} Z_l E_m^{-1} \prod_{l=m}^{1} Z_l^T)]^j_{\mathcal{V}_0} = \lambda^{-1}_{E_m{}^j} [\Lambda(B_m)]^j_{\mathcal{V}_0} = \lambda^{-1}_{E_m{}^j} \prod_{l=m}^{1} \left((r_l^j)^2 + (p_l^j)^2 \right),$$

with $\lambda^j_{E_m} = (r_m^j)^2 \lambda^j_{E_{m-1}} + (p_m^j)^2 \lambda^{j'}_{E_{m-1}}$ for $j = 1, 2, \ldots, n_m$ and $j' = n_{m-1} + 1 - j$.

Proof. The proof is very similar to the one for Lemma 4.2.1, details can be found in Appendix B. We give a brief outline of the proof here. We start by block-diagonalizing the fine grid linear system A. Consequently, we recursively multiply the block-diagonal version of A with the matrix containing the 2×2 blocks representing $Z_1, Z_2 \ldots Z_m$ and ${Z_1}^T, {Z_2}^T \ldots {Z_m}^T$ respectively to obtain E_m. Finally, we rewrite Q_m in terms of B_m, and use Lemma 4.2.1 to obtain the final analytical expressions. □

We can now easily block-diagonalize P_m as follows. We start by writing P_m in block-diagonal form using Theorem 4.2 and our rearranged basis ${\mathcal{V}_0}^j$.

$$[\Lambda(P_m)]^j_{\mathcal{V}_0} = [I - AQ_m]^j_{\mathcal{V}_0} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}^j_{\mathcal{V}_0} - \frac{\lambda^j_{B_m}}{\lambda^j_{E_m}} \begin{bmatrix} \lambda^j_A & 0 \\ 0 & \lambda^{j'}_A \end{bmatrix}^j_{\mathcal{V}_0} = \begin{bmatrix} 1 - \frac{\lambda^j_A \lambda^j_{B_m}}{\lambda^j_{E_m}} & \frac{\lambda^j_A \lambda^j_{B_m}}{\lambda^j_{E_m}} \\ \frac{\lambda^{j'}_A \lambda^j_{B_m}}{\lambda^j_{E_m}} & 1 - \frac{\lambda^{j'}_A \lambda^j_{B_m}}{\lambda^j_{E_m}} \end{bmatrix}^j_{\mathcal{V}_0}$$

Including the CSLP-preconditioner M^{-1} and applying the multilevel-deflation preconditioner P_m to the coefficient matrix A finally gives the block-diagonal expressions of the preconditioned system

$$[\Lambda(P_m M^{-1} A)]^j_{\mathcal{V}_0} = \frac{\lambda^j_A}{\lambda^j_M} \begin{bmatrix} 1 - \frac{\lambda^j_A \lambda^j_{B_m}}{\lambda^j_{E_m}} & \frac{\lambda^j_A \lambda^j_{B_m}}{\lambda^j_{E_m}} \\ \frac{\lambda^{j'}_A \lambda^j_{B_m}}{\lambda^j_{E_m}} & 1 - \frac{\lambda^{j'}_A \lambda^j_{B_m}}{\lambda^j_{E_m}} \end{bmatrix}^j_{\mathcal{V}_0}.$$

At last, we obtain the eigenvalues of $P_m M^{-1} A$ for $j = 1, 2, \ldots, n_1$ and $j' = n_0 + 1 - j$, by computing the trace of each respective block

$$\lambda^j(P_m M^{-1} A) = \frac{\lambda^j_A}{\lambda^j_M} \left(1 - \frac{\lambda^j_A \lambda^j_{B_m}}{\lambda^j_{E_m}} \right) + \frac{\lambda^{j'}_A}{\lambda^j_M} \left(1 - \frac{\lambda^{j'}_A \lambda^j_{B_m}}{\lambda^j_{E_m}} \right), \tag{10}$$

with $\lambda^j_{B_m} = \prod_{l=m}^{1} \left((r_l^j)^2 + (p_l^j)^2 \right)$.

4.3. Spectral analysis

Using these expressions, we proceed by analyzing the various operators involved in the multi-level deflation operator. For the purpose of this section, we choose the shift in the CSLP-preconditioner to be large ($\beta_2 = 1$) in order to emphasize the effect of the deflation method. In this section, we will plot the expressions from Eq. (4) and Eq. (5), which are the global and local multi-level deflation preconditioner respectively. The global operator is obtained by inverting E_m at level m exactly and prolongating back to the fine grid until we obtain $P_{h,m}$. The local operator is obtained by applying two-level deflation locally at level m, which gives us P_m.

4.3.1. Global near-zero eigenvalues

We start with by denoting the global deflation operator by $P_{h,m}$, where m indicates the level. We will analyze the spectrum up to the level where the coefficient matrix becomes negative definite, which according to Corollary 4.1.2 is at $\tilde{m} = 3$. Before we start with the spectral analysis, several remarks are in place. The eigenvalues of the preconditioned systems can be retrieved analytically in case we have Dirichlet boundary conditions. In case of Sommerfeld boundary conditions, the analytical eigenvalues can not be determined and we are forced to compute them numerically. For the sake of completeness, we will include them in the spectral analysis for the one-dimensional model problems.

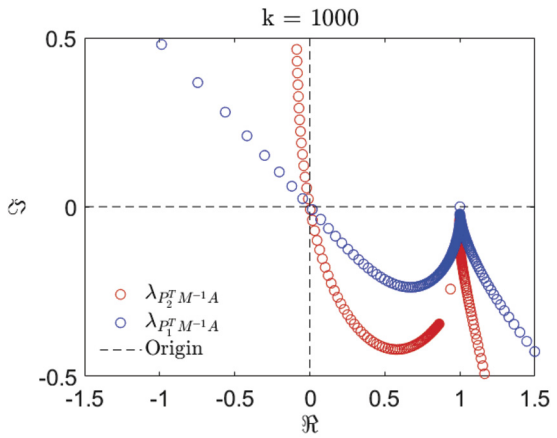


Fig. 2. Linear interpolation (Dirichlet). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Blue uses a two-level scheme and red uses a three-level scheme.

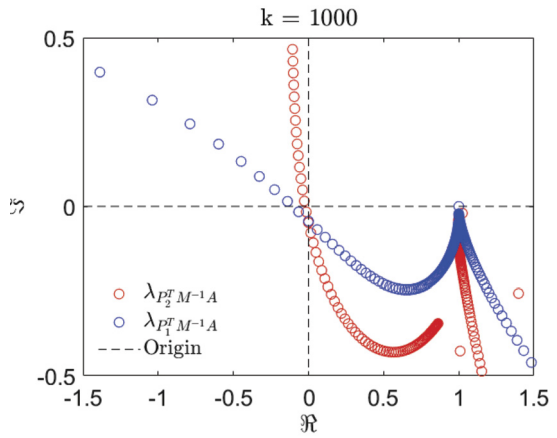


Fig. 3. Linear interpolation (Sommerfeld). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Blue uses a two-level scheme and red uses a three-level scheme.

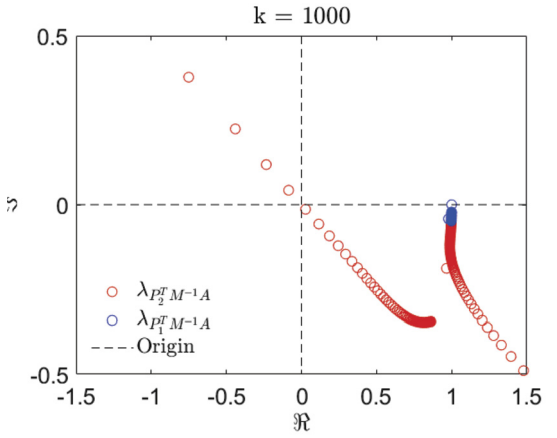


Fig. 4. Quadratic rational Bezier (Dirichlet). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Blue uses a two-level scheme and red uses a three-level scheme.

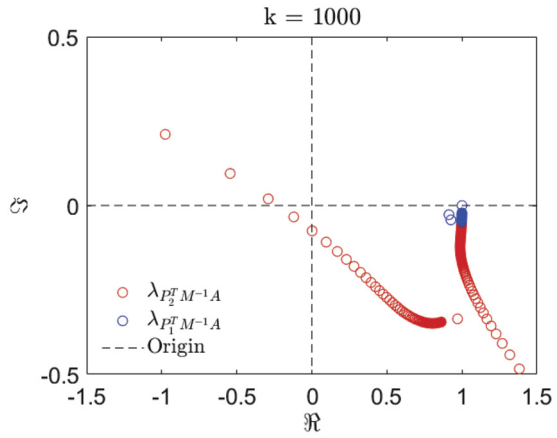


Fig. 5. Quadratic rational Bezier (Sommerfeld). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Blue uses a two-level scheme and red uses a three-level scheme.

For $k = 1\,000$ we define $P_{h,1}, P_{h,2}$ according to Eq. (4). We use 10 grid points per wavelength (gpw) for this part of the analysis. Fig. 2 and Fig. 3 contain the results using linear interpolation for both Dirichlet and Sommerfeld boundary conditions respectively. Similarly, Fig. 4 and Fig. 5 contain the results using high-order deflation vectors.

When we use Dirichlet boundary conditions and compare Fig. 2 to Fig. 4, we immediately observe that there are less near-zero eigenvalues on the first and second level when using higher-order deflation vectors. Especially for the first level (blue, moving from n to $\frac{n}{2}$), the difference seems to be significant.

Using Sommerfeld conditions, the conditions for Fig. 3 and Fig. 5 the conclusion is similar. The use of these boundary conditions for the linear interpolation case seems to be more prevalent at the first level (blue). Here, Fig. 3 shows a slightly different angle away from the zero, compared to Fig. 2. At the second level, there appears to be no difference. If we move to higher-order deflation vectors in Fig. 4 and Fig. 5 for both the Dirichlet and Sommerfeld case, the eigenvalues at the first level remain clustered near the point $(1, 0)$ in the complex plane. The eigenvalues start dispersing once we move to the second level (red) (from $\frac{n}{2}$ to $\frac{n}{4}$). An important distinction is visible for the higher-order case. Using Sommerfeld conditions in Fig. 5 keeps the eigenvalues of $P_{h,2}$ away from zero relative to Fig. 4.

Note that for $m \geqslant 3$, we have proved that the resulting coarse-grid coefficient matrix E_3 is completely negative definite. Consequently, the problem of the near-zero eigenvalues of $E_{m \geqslant 3}$ resolves itself at these levels given that the location of the smallest eigenvalue in terms of magnitude is now fixed away from zero due to the matrix being negative-definite. Moreover, the further down the levels we move, the smaller the number of eigenvalues become which get projected away.

Next, we repeat the analysis for $k = 1\,000$, but this time we use 20 gpw. We define $P_{h,1}, P_{h,2}$ and $P_{h,3}$ according to Eq. (4). When we use Dirichlet boundary conditions, comparing Fig. 6 and Fig. 8 immediately shows that there are more near-zero eigenvalues when using linear interpolation. Overall, for the first and second level, the spectrum remains tightly

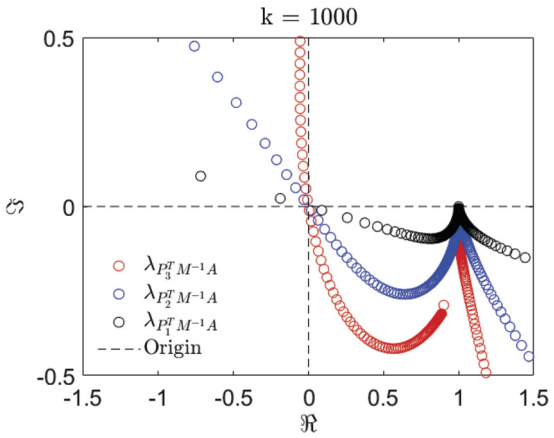


Fig. 6. Linear interpolation (Dirichlet). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Black uses a two-level scheme, blue uses a three-level scheme and red uses a four-level scheme.

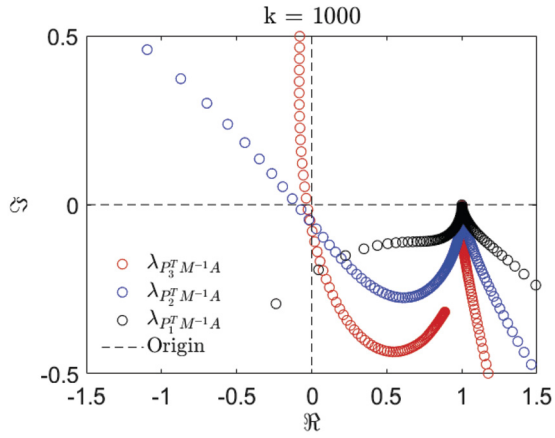


Fig. 7. Linear interpolation (Sommerfeld). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Black uses a two-level scheme, blue uses a three-level scheme and red uses a four-level scheme.

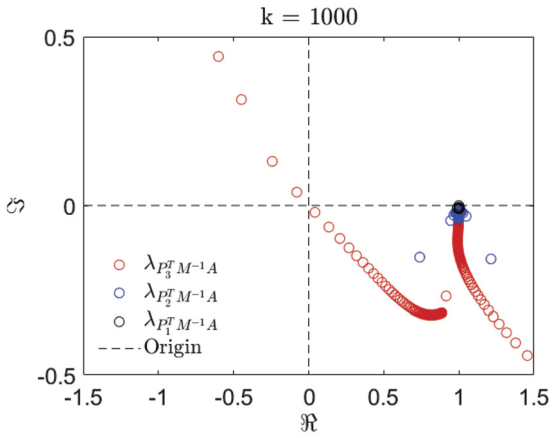


Fig. 8. Quadratic rational Bezier (Dirichlet). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Black uses a two-level scheme, blue uses a three-level scheme and red uses a four-level scheme.

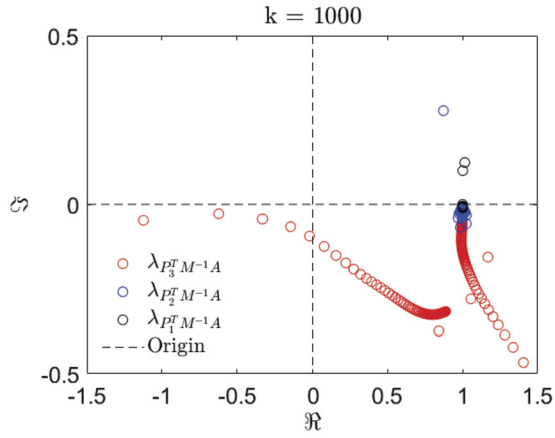


Fig. 9. Quadratic rational Bezier (Sommerfeld). Spectrum of the global deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Black uses a two-level scheme, blue uses a three-level scheme and red uses a four-level scheme.

clustered when using higher-order deflation. Thus, for the linear interpolation case in Fig. 6, the first level (black) appears to benefit the most from using a finer grid.

Moving on to the Sommerfeld boundary conditions, comparing Fig. 7 and Fig. 9 shows a large difference in the clustering of the eigenvalues at the first and second level (black and blue). Using a finer grid seems to affect the first and second level, i.e. the spectrum of $P_{h,1}$ and $P_{h,2}$ of the higher-order case more. If we compare Fig. 7 and Fig. 3 we only observe a significant difference at the first level. In all cases it shows that the largest clustering gain can be achieved at the levels where the matrix remains highly indefinite.

4.3.2. Local deflated near-zero eigenvalues

Here we start by plotting the local near-zero eigenvalues for $k = 1\,000$ of P_2 and P_3 and compare them to $P_{h,2}$ and $P_{h,3}$ respectively. For this part of the analysis, we will only use the case with Dirichlet boundary conditions. So far we have observed that the inclusion of Dirichlet boundary conditions leads to a spectrum which appears to be less favorably clustered compared to when we include Sommerfeld boundary conditions.

Starting with 10 gpw, for all cases irrespective of linear interpolation or higher-order deflation vectors, the eigenvalues of the local and global operator are similar. If we use a higher-order scheme the largest gain in terms of removing the near-zero eigenvalues is realized at level $m \leqslant 2$. At these levels, comparing Fig. 10 and Fig. 12, we observe that we have less near-zero eigenvalues both globally and locally. As soon as the matrix becomes negative definite, the spectrum is fully determined by the spectrum of CSLP applied to the global and/or local coefficient matrix. Comparing Fig. 11 and Fig. 13 shows no difference irrespective of the underlying basis functions used to construct the deflation vectors.

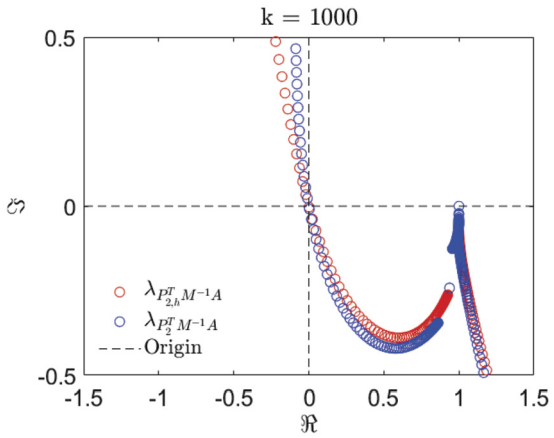


Fig. 10. Linear interpolation ($m = 2$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Red represents the global spectrum and blue represents the local spectrum.

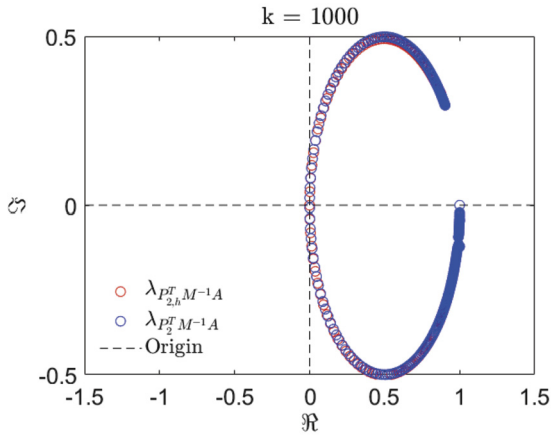


Fig. 11. Linear interpolation ($m = 3$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Red represents the global spectrum and blue represents the local spectrum.

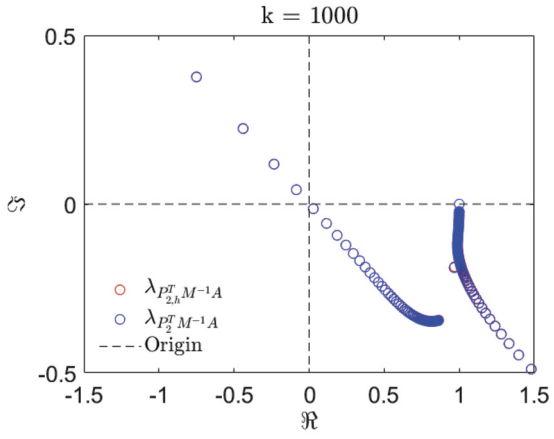


Fig. 12. Quadratic rational Bezier ($m = 2$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Red represents the global spectrum and blue represents the local spectrum.

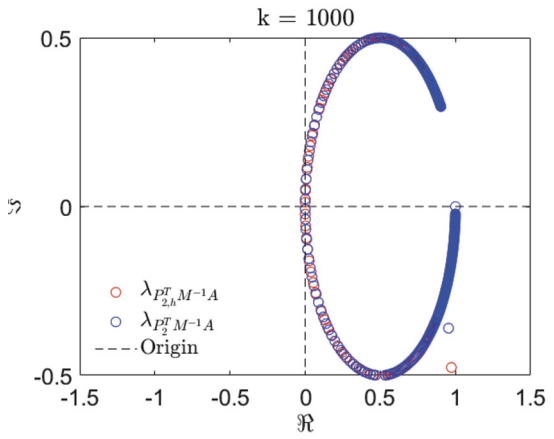


Fig. 13. Quadratic rational Bezier ($m = 3$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.625$ or equivalently 10 gpw. Red represents the global spectrum and blue represents the local spectrum.

We repeat the analysis for $k = 1\ 000$, again using 20 gpw. For both linear interpolation and quadratic rational Bezier, the global and local preconditioned spectra appear similar. We again have plotted $m = 2$ in Fig. 14 and Fig. 16 and $m = 3$ in Fig. 15 and Fig. 17. Note that when using 20 gpw the resulting underlying coarse linear system does not become negative definite at $m = 3$. Instead, the linear systems become negative definite at $m \geqslant 4$. As for the levels discussed here, we clearly observe a significant difference in clustering at both levels, when we use higher-order deflation vectors.

At the coarsest level where the matrix is still indefinite, in this case $m = 3$, we observe in Fig. 17 that the spectrum is slowly starting to disperse for the higher-order scheme. In terms of magnitude, it is easy to see that the near-zero eigenvalues in Fig. 15 are smaller. This observation supports the notion that the largest effect of using a deflation strategy with higher-order basis function can be realized when the matrices at the finer level are highly indefinite. These are also the linear systems which are the largest in terms of the problem size.

4.3.3. Local near-zero eigenvalues

Here we proceed by plotting the eigenvalues of the coarse-grid systems for levels $m \leqslant 3$. We take $k = 100$ as for smaller k, the plot containing the complete spectrum and the near-zero eigenvalues is better visible. The results are comparable to the ones obtained for the two-level ADP preconditioner. The near-zero eigenvalues for all levels where the coefficient matrices are indefinite remain aligned, see Fig. 19. Comparing this to Fig. 18 for the linear interpolation case, the near-zero eigenvalues start shifting as we move from $m = 0$ to $m = 2$. Note that at $m = 3$ all eigenvalues are negative, which follows from Corollary 4.1.2.

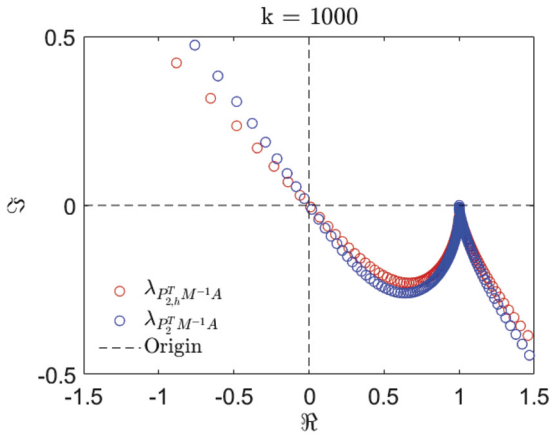


Fig. 14. Linear interpolation ($m = 2$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Red represents the global spectrum and blue represents the local spectrum.

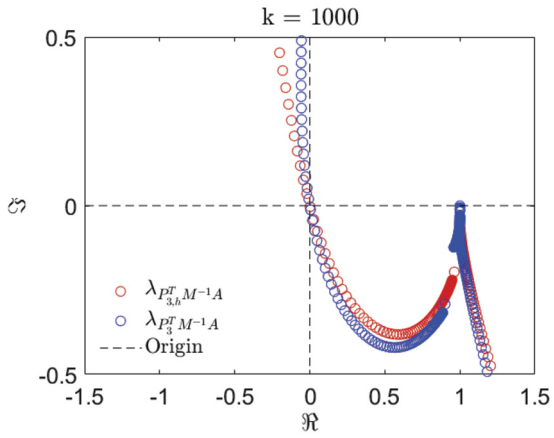


Fig. 15. Linear interpolation ($m = 3$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Red represents the global spectrum and blue represents the local spectrum.

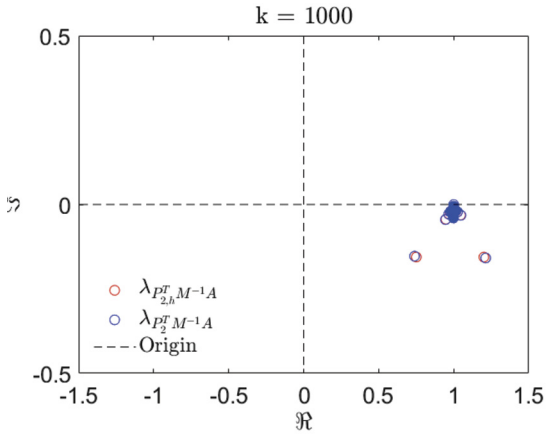


Fig. 16. Quadratic rational Bezier ($m = 2$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Red represents the global spectrum and blue represents the local spectrum.

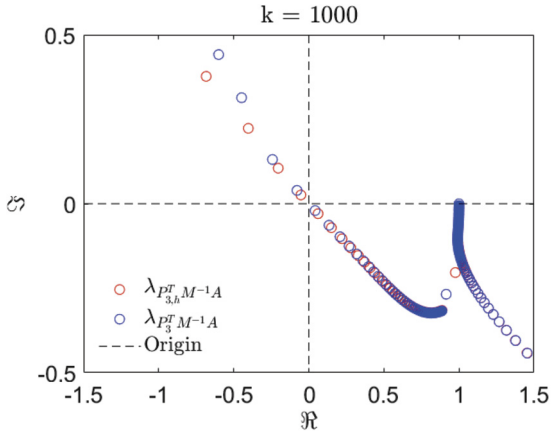


Fig. 17. Quadratic rational Bezier ($m = 3$). Spectrum of global and local deflation + CSLP preconditioned system using $kh = 0.3125$ or equivalently 20 gpw. Red represents the global spectrum and blue represents the local spectrum.

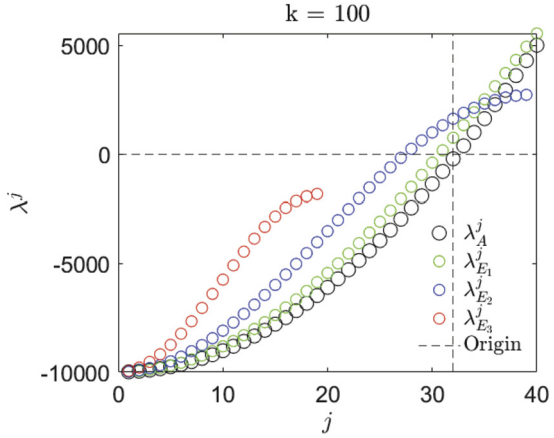


Fig. 18. Linear Interpolation. Spectrum of the coarse linear systems for $k = 100$ and $m \leqslant 3$.

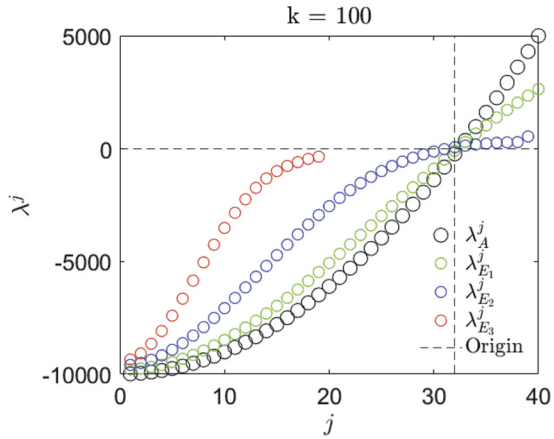


Fig. 19. Quadratic Rational Bezier. Spectrum of the coarse linear systems for $k = 100$ and $m \leqslant 3$.

Table 2
Number of outer FGMRES-iterations for MP2-A. ⊘ indicates that the problem size $n > 4 \times 10^6$ and has exceeded the working memory capacity. CPU time is given in seconds. Max. iterations for inner Bi-CGSTAB have been set at $6n^{(l)\,\frac{1}{4}}$, for $l = 1, 2$.

MP2-A	$kh = 0.625$ - 10 gpw			$kh = 0.17$ - 35 gpw		
f (Hz)	Iterations	CPU(s)	n	Iterations	CPU(s)	n
50	8	0.22	6 561	8	0.90	40 401
100	8	0.87	25 921	8	3.75	160 801
200	9	5.66	103 041	8	22.84	641 601
400	11	29.68	410 881	9	176.45	2 563 201
800	15	341.11	1 640 961	⊘	⊘	⊘

5. Numerical experiments

In this section we will provide numerical experiments to study the convergence behavior of our multilevel preconditioner. All experiments are implemented sequentially on a Dell laptop using 8 GB RAM and a i7-8665U processor. An exact solve is performed at the coarsest level with problem size $n < 10$. Moreover, we allow one FGMRES-iteration on each level to retain the V-cycle structure ($\gamma = 1$) and the rule of thumb 10 grid points per wavelength, unless stated otherwise.

At levels where the matrix is indefinite, Bi-CGSTAB is used as smoother. For all other levels, we use one smoothing step with damped Jacobi. For constant wavenumber model problems we need more Bi-CGSTAB iterations, as for heterogeneous problems the grid has been resolved with respect to the largest wavenumber. Thus for smaller values, there is in fact more accuracy than the required 10 grid points per wavelength. In all cases the number of inner iterations is determined by a constant times $n^{(l)\,\frac{1}{4}}$, where $n^{(l)}$ is the problem size of the linear system on level l, given that we do not want more iterations than necessary on the coarser levels. We will use the following test models and report the number of iterations.

- 2D constant wavenumber (Sommerfeld) - MP2-A
- 2D constant wavenumber (Dirichlet + Sommerfeld) - MP2-B
- 2D wedge (Sommerfeld) - MP2-C
- 2D full Marmousi (Sommerfeld) - MP2-D
- 3D sine model (Dirichlet) - MP3-A
- 3D time-harmonic Elastic Wave equation (Dirichlet + Sommerfeld) - MP3-B

If timings are reported they will include the CPU-time in seconds using Matlab 2019Rb. The timings are for indicative purposes, please see our detailed complexity analysis above. The timings include all costs associated with the algorithm, including setting up the coarse-grid linear systems through matrix-matrix multiplications.

5.1. Two-dimensional constant wavenumber model problems

We start by presenting the numerical results in Table 2 for the constant wavenumber model problem MP2-A and MP2-B. For this subsection we will use two grid resolutions: $kh = 0.625$ and $kh = 0.17$, which boils down to 10 and 35 gpw respectively. The reason for this is that we want to investigate how the solver performs once we need to solve larger systems in an effort to counteract the pollution error. It is known that for the Helmholtz problem, the numerical solution suffers from pollution error due to numerical dispersion [29–31]. The pollution effect can not be eliminated completely. One way to mitigate its effect is to use finer grids, which is why we will also investigate the case where we have 35 gpw.

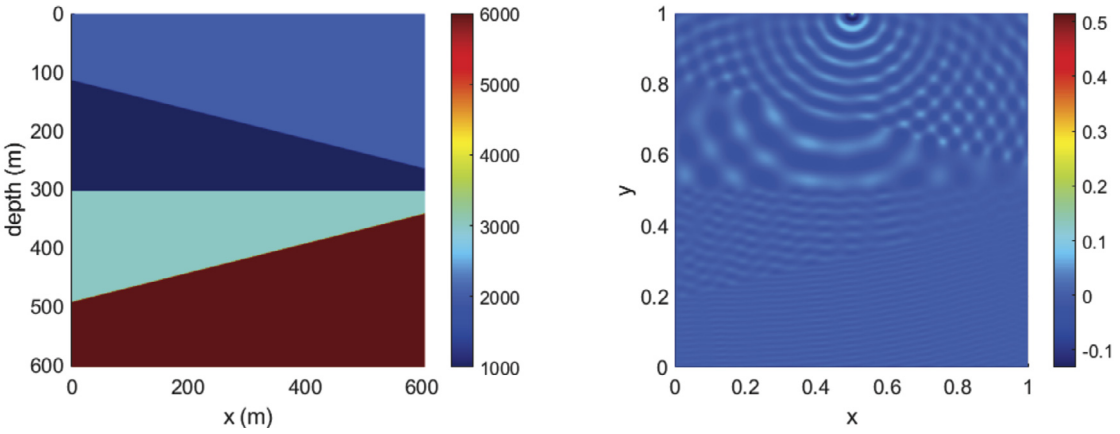
We start with MP2-A, where the results are presented in Table 2. We observe that the number of iterations slowly grows with the wavenumber k. For the largest wavenumber $k = 800$, we need 15 iterations using 10 gpw. Once we use 35 gpw, we obtain a wavenumber independent solver. In terms of CPU timings, the time scaling is clearly visible when we have wavenumber independent convergence. Here, a fourfold increase in the problem size leads to an approximate fourfold increase in the CPU time. As this can vary across hardware configurations, we have included the CPU time as an indication. For the theoretical complexity and upper bound, please see Theorem 3.1.

We proceed with the results for MP2-B, which now also includes Dirichlet boundary conditions as reported in Table 3. The inclusion of this condition immediately results in an increase in the number of iterations. In fact using 10 gpw, we now have that for $k = 800$, convergence is reached in 30 iterations. This is double the number of iterations when using Sommerfeld conditions entirely. These results are in line with our theory as we already noticed in the spectral analysis section that the use of Dirichlet conditions leads to a less favorable spectrum Section 4.3. At coarser levels near-zero eigenvalues start to appear slowly, which is why we indeed expect the number of iterations to increase slightly.

To put these results into perspective for $kh = 0.625$, i.e. 10 gpw, if we were to use the industry standard configuration with the CSLP inverted approximately using one multigrid V-cycle, then for MP2-B and $k = 200$ we would need 296 Bi-CGSTAB iterations which take 99.96 seconds to reach convergence. For MP2-A we would need 160 iterations which takes approximately 40 seconds.

Table 3
Number of outer FGMRES-iterations for MP2-B. ⊘ indicates that the problem size $n > 4 \times 10^6$ and has exceeded the working memory capacity. CPU time is given in seconds. Max. iterations for inner Bi-CGSTAB have been set at $6n^{(l)\,\frac{1}{4}}$, for $l = 1, 2$.

MP2-B	$kh = 0.625$ - 10 gpw			$kh = 0.17$ - 35 gpw		
f (Hz)	Iterations	CPU(s)	n	Iterations	CPU(s)	n
50	8	0.32	6 561	8	1.30	40 401
100	11	1.93	25 921	9	7.16	160 801
200	16	18.52	103 041	11	62.95	641 601
400	23	193.60	410 881	17	1075.77	2 563 201
800	30	891.50	1 640 961	⊘	⊘	⊘



(a) Velocity $c(x, y)$ (b) Pressure field

Fig. 20. Velocity profile and numerical solution for MP2-C for $f = 60$.

Without a preconditioner and if no outer FGMRES with multilevel deflation were to be used, we would need 6 188 Bi-CGSTAB iterations with a total time of 70.98 seconds for MP2-B and 2 797 with 32.54 seconds for MP2-A. For $k = 200$, the maximum number of inner Bi-CGSTAB iterations at the indefinite levels is set at approximately 102. While the inner iterations may appear to be a lot, note that, if we were to apply 9 times 102 iterations on a stand-alone basis, it is still less than the number of iterations of Bi-CGSTAB without any preconditioner (6 188 and 2 797 for MP2-B and MP2-A respectively). Moreover, both GMRES and Bi-CGSTAB used with the approximated inverse of the CSLP, require much more computation time. What we thus observe is that the synergy of using outer FGMRES to create the hierarchy of coarse-grid levels with inner Bi-CGSTAB leads to both lower computation times and lower iteration counts for highly indefinite systems.

Finally, we would like to make a brief comparison with the old deflation scheme (DEF-ML) based on linear interpolation. Again for $k = 200$ and $kh = 0.625$, the DEF-ML preconditioner requires 24 iterations and 22.25 seconds to reach convergence for MP2-A and 61 iterations and 93.32 seconds for MP2-B respectively. In both cases the difference with respect to the higher-order deflation scheme is significant (ADP scheme) as for the new method it takes 9 iterations and 5.66 seconds for MP2-A and 8 iterations and 22.84 seconds respectively.

5.2. Two-dimensional heterogeneous model problems

In this subsection we will provide the results to the numerical experiments for the Wedge model (MP2-C) and the Marmousi model (MP2-D).

5.2.1. Wedge

Starting with MP2-C, Fig. 20 illustrates the underlying geometry of the wedge and the numerical solution. We divide the numerical domain into four sections containing a wedge.

From Table 4 we again observe that for both velocity profiles reported, the number of iterations weakly depends on the frequency. Note that when $c(x, y) \in [500, 3\,000]$, the dimensionless wavenumber $k(x, y)$ lies between and 125 and 750 for $f = 60$. Similarly for $c(x, y) \in [1\,000, 6\,000]$, we have that $k(x, y)$ lies between 63 and 375. As a result, we need more grid points per wavelength when $c(x, y) \in [500, 3\,000]$ to resolve all waves with a resolution of 10 gpw. Consequently, the problem size is larger whenever $c(x, y) \in [500, 3\,000]$.

Table 4
Number of outer FGMRES-iterations for MP2B (Wedge). The largest wavenumber k is resolved using 10 gpw. Max. iterations for inner Bi-CGSTAB have been set at $6n^{(l)\frac{1}{4}}$, for $l = 1, 2$.

$k = \frac{2\pi f 1\,000}{c(x,y)}$	$c(x,y) \in [500, 3\,000]$ m/s			$c(x,y) \in [1\,000, 6\,000]$ m/s		
f (Hz)	Iterations	CPU(s)	n	Iterations	CPU(s)	n
10	12	4.10	41 209	9	0.58	10 201
20	18	37.14	162 409	12	3.97	41 209
30	22	118.22	366 025	16	18.99	91 809
40	29	370.91	648 025	19	34.29	162 409
60	35	1 097.31	1 456 849	22	174.03	366 025

Table 5
Number of outer FGMRES-iterations for MP2B (Wedge). The largest wavenumber k is resolved using 35 gpw to account for the pollution effect. Max. iterations for inner Bi-CGSTAB have been set at $6n^{(l)\frac{1}{4}}$, for $l = 1, 2, 3, 4$.

$k = \frac{2\pi f 1\,000}{c(x,y)}$	$c(x,y) \in [500, 3\,000]$ m/s			$c(x,y) \in [1\,000, 6\,000]$ m/s		
f (Hz)	Iterations	CPU(s)	n	Iterations	CPU(s)	n
10	12	17.05	253 009	12	4.49	64 009
20	15	159.25	1 014 049	12	24.86	253 009
30	17	524.42	2 277 081	13	82.78	570 025

The fact that the underlying wavenumber is much larger in the first case, where $c(x, y) \in [500, 3\,000]$, explains why we need more iterations to reach convergence. Similarly, for both cases in Table 4, we observe that if the problem size is doubled, the computing time increases by a factor of 3.

Next, we investigate how the preconditioner performs if we refine the grid using 35 gpw instead of 10 gpw. One of the reasons why we investigate this is to assess the convergence behavior once we allow for pollution correction. The pollution error is present in numerical solutions for large wavenumbers or high-frequencies due to underlying numerical dispersion. The natural way to resolve this issue is by using a very fine and restrictive grid resolution. In Table 5 we report the number of iterations when using 35 gpw. Note that now for $f = 30$ and $c(x, y) \in [500, 3\,000]$, the largest problem size is now 2 277 081 compared to 366 025 previously. Observe that using a finer grid, also reduces the number of iterations to reach convergence. This can be explained from a theoretical perspective as well as we have observed from the spectral analysis that using finer grids leads to more levels which already have less near-zero eigenvalues, see Section 4.3. In terms of CPU timings, we find that while the problem size is now approximately 6 times larger, the CPU time has increased by a factor of 4. The same holds for when we take $c(x, y) \in [500, 3\,000]$.

5.2.2. Marmousi

Next we consider an adapted version of the original Marmousi problem developed in [18]. The original domain has been truncated to $\Omega = [0, 8\,192] \times [0, 2\,048]$ in order to allow for efficient geometric coarsening of the discrete velocity profiles. This way, the problem sizes can remain powers of 2. Similar to some experiments in the literature, the coarsening keeps the proportions of the original velocity the same but lets $c(x, y)$ vary between $2\,587.5 \leqslant c \leqslant 3\,325$. On Ω, we define

$$\begin{aligned} -\Delta u(x,y) - k(x,y)^2 u(x,y) &= \delta(x - 4\,000, y),\ (x,y) \in \Omega \backslash \partial\Omega \subset \mathbb{R}^2, \\ \left(\frac{\partial}{\partial \mathbf{n}} - \sqrt{-1}k\right) u(x,y) &= 0,\ (x,y) \in \partial\Omega, \end{aligned} \tag{11}$$

where $\mathbf{n}$ denotes the outward normal unit vector. The wave number is given by $k(x, y) = \frac{2\pi f}{c(x,y)}$, where the frequency f is given in Hertz.

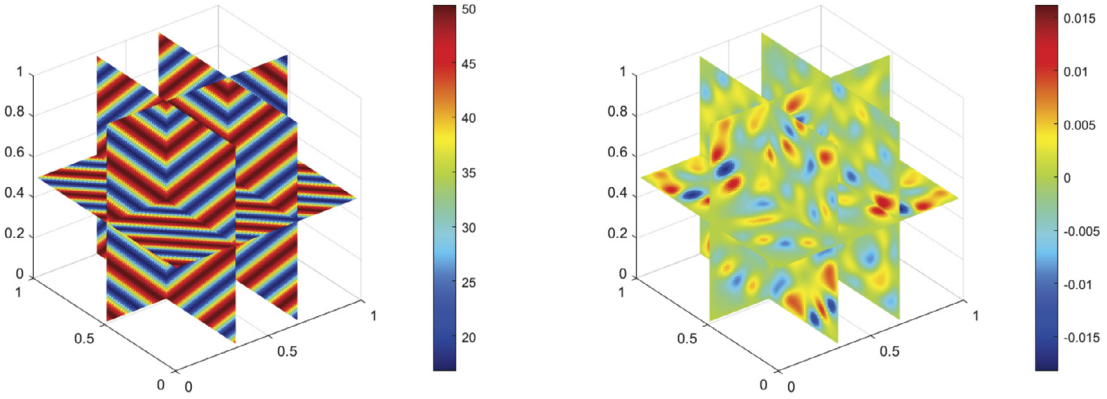
The results from Table 6 show that the number of iterations again weakly depends on the wave number. Here we experiment with using a W-cycle instead of a V-cycle to construct the multilevel hierarchy. For the Marmousi problem we observe that it leads to a lower number of iterations for all the reported frequencies. However, for the largest test case with $n = 1\,878\,251$, while the number of iterations are lower (24 instead of 33), the computation time increases. This can be explained by noting that for the W-cycle, more work is performed within each level.

5.3. Three-dimensional heterogeneous model problems

In this subsection we will provide the results to the numerical experiments for the Sine model (MP3-A) and the Elastic wave model (MP3-B). Note that in the elastic wave equation, both force and displacement are vector quantities.

Table 6
Number of outer FGMRES-iterations for the Marmousi problem MP2C, where f denotes the frequency in Hertz. The largest wavenumber has been resolved using 10 gpw. Max. iterations for inner Bi-CGSTAB have been set at $6n^{(l)\frac{1}{4}}$, for $l = 1, 2$.

	n	$\gamma = 1$		$\gamma = 2$	
f (Hz)	n	Iterations	CPU(s)	Iterations	CPU(s)
10	66 177	18	18.11	13	18.55
20	263 425	21	117.68	14	75.17
40	1 051 137	30	810.90	20	914.30
60	1 878 251	33	1 559.30	24	1 633.57



(a) Wave number $\tilde{k}(x, y, z)$ (b) Numerical solution

Fig. 21. Wave number and numerical solution for MP3B (Sine) for $f = 8$.

Table 7
Number of outer FGMRES-iterations for sine-problem (MP3C), where f denotes the frequency in Hertz. Max. iterations for inner Bi-CGSTAB have been set at $6n^{(l)\frac{1}{4}}$, for $l = 1, 2$.

		8π			
$k = 2\pi f$		$\gamma = 1$		$\gamma = 2$	
f (Hz)	n	Iterations	CPU(s)	Iterations	CPU(s)
4	68 921	8	3.04	6	4.02
8	531 441	26	133.68	15	123.21
12	1 771 561	49	1 259.18	28	1 359.92

5.3.1. Sine model

In this model we artificially construct a variant of the Helmholtz equation with highly varying wavenumbers across the entire numerical domain. We therefore define the following

$$-\Delta u(x, y, z) - \tilde{k}(x, y, z)^2 u(x, y, z) = \delta(x - \frac{1}{2}, y, z), (x, y, z) \in \Omega = [0, 1]^3 \subset \mathbb{R}^3,$$
$$\tilde{k}(x, y, z) = \frac{k_1^2 + k_2^2}{2} + \left| \frac{k_2^2 - k_1^2}{2} \right| \sin(8\pi(x + y + z)), k_1 \in \mathbb{N}, k_2 = \frac{k_1}{3}$$
$$u(x, y, z) = 0, (x, y, z) \in \partial\Omega.$$

An illustration of the wavenumber profile is given in Fig. 21 (a).

The results are reported in Table 7. We observe that for this highly varying wave number model problem, where the wave number switches rapidly from low- to high-contrast, the dependency of the iteration count on $k(x, y, z)$ is more pronounced. Experimenting with the W-cycle instead of the V-cycle does lead to a lower iteration count. However, for the largest test case ($f = 12$), we again observe an increase in the computation time.

Table 8
Number of outer FGMRES-iterations for the time-harmonic elastic wave equation (MP3-B), where f denotes the frequency in Hertz using 20 gpw. Max. iterations for inner Bi-CGSTAB have been set at $7n^{(l)\,\frac{1}{4}}$, for $l = 1, 2$.

$k = 2\pi f$	n	$\gamma = 1$		$\gamma = 2$	
f (Hz)		Iterations	CPU(s)	Iterations	CPU(s)
1	19 968	8	2.87	8	3.59
2	147 033	11	87.21	9	77.97
4	1 127 463	15	1 665.68	13	1 735.29

5.3.2. Elastic wave

For the time-harmonic elastic wave equation in a three-dimensional wedge we use the model from [32]. No splitting has been performed and the global system is solved. The results are given in Table 8. We again experiment with the V-cycle and the W-cycle. For the frequencies reported, the number of iterations slowly increases with the wave number. When comparing the computation time, once the frequency increases and the problem becomes large, the V-cycle is preferred. While the W-cycle leads to less iterations, it requires more computational work.

6. Conclusion

In this work we extend the two-level deflation preconditioner using higher-order deflation vectors to a multilevel deflation preconditioner [22]. We provide theoretical and numerical evidence to show that up to a certain level, the coefficient matrices are indefinite. These levels are of paramount importance as the near-zero eigenvalues at these levels can effectively be removed by the multilevel deflation preconditioner. If the near-zero eigenvalues are aligned, then the eigenvalues cluster near the point $(1, 0)$ in the complex plane, accelerating the convergence of the underlying Krylov solver.

After this level, the subsequent coarse coefficient matrices become negative definite. We implement $O(n^{\frac{1}{4}})$ inner Bi-CGSTAB-iterations on the indefinite levels to approximate the CSLP using the inverse of the wave number k as the shift $(\beta_2 = k^{-1})$ and use damped Jacobi on the levels where the matrices are negative definite. This circumvents the difficulty of multigrid approximations, where the shift β_2 has to be kept large. The proposed configuration leads to scalable results as we obtain close to wave number independent convergence in terms of a fixed number of iterations. It furthermore, extends the results for both a constant and non-constant wave number model problem, such as the two-dimensional industrial Marmousi model problem and the three-dimensional elastic wave equation. Additionally, sequential implementation of the method leads to scalable timing results for the model problems, which has been demonstrated using numerical experiments and a complexity analysis.

CRediT authorship contribution statement

Vandana Dwarka: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Cornelis Vuik:** Funding acquisition, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank two anonymous reviewers for their time, constructive feedback and valuable comments which allowed us to greatly improve an earlier version of this paper.

Appendix A. Proof of Theorem 4.1

Basis and ordering

We start by defining $n_m = \frac{n}{2^m}$ and rearranging the space spanned by the eigenvectors at each level such that we obtain the following subspace

$$\mathcal{V}_m^j = \text{span}\{v_m^j, v_m^{n_m+1-j}\},$$

for $j = 1, 2, \ldots, n_{m+1}$. Moreover let

$$V_{m+1} = \bigoplus_{j=1}^{n_{m+1}} \text{span}\{v_{m+1}^j\},$$

denote the space spanned by the eigenvectors at a coarser level $m+1$. Note that the basis spans $\mathbb{C}^{n_m}$ and $\mathbb{C}^{n_{m+1}}$ as we can write

$$\mathbb{C}^{n_m} = \bigoplus_{j=1}^{n_{m+1}} \mathcal{V}_m^j \text{ and } \mathbb{C}^{n_{m+1}} = V_{m+1}^j,$$

and at each subsequent level $m+1$ we re-order the basis to obtain $\mathcal{V}_{m+1}$. Thus, on each level we define the automorphism such that we can bring the basis of V_m in to the order of $\mathcal{V}_m$

$$\alpha_{\pi(j)}^m : V_m \to V_m : j \mapsto n_m + 1 - (j-1) \text{ for } j \text{ is even}.$$

For $m = 0, 1, 2 \ldots m_{\max}$, the linear interpolation and restriction operator maps between subsequent vector spaces

$$\mathcal{I}_m^{m+1} : \mathcal{V}_m \to V_{m+1}, \text{ such that } \mathcal{V}_m^j \mapsto \mathcal{I}_m^{m+1} V_m^j$$
$$\mathcal{I}_{m+1}^m : V_{m+1} \to \mathcal{V}_m, \text{ such that } v_{m+1}^j \mapsto \mathcal{I}_{m+1}^m v_{m+1}^j.$$

Restriction operator

We will now apply the corresponding matrices to the respective eigenvectors on each level, where we let $\mathcal{I}_m^{m+1} = Z_{m+1}$. We start by taking $m = 0$. Using the basis of eigenvectors for $\mathcal{V}_0$ we have for index j

$$\begin{aligned}\left[Z_1^T v_0^j\right]_i &= \frac{1}{4}\left(\sin((2i-1)h\pi j) + 2\sin(2ih\pi j) + \sin((2i+1)h\pi j)\right), \\ &= \frac{1}{2}\left(1 + \cos(j\pi h)\right)\sin(2hi\pi j), \\ &= C_{1,h}^j \left[v_1^j\right]_i.\end{aligned}$$

Now, for the complementary mode on level $m = 0$ corresponding to index j we define $j' = n_0 + 1 - j$. Note that we can write

$$[v_0^{j'}]_i = -(-1)^j \sin(ihj\pi), \tag{A.1}$$
$$i = 1, 2, \ldots n_m, \text{ and } j = 1, 2, \ldots n_{m+1}.$$

Applying the restriction operator to the complementary eigenvector gives

$$\begin{aligned}\left[Z_1^T v_0^{j'}\right]_i &= \frac{1}{4}\left(\cos(j\pi h)\sin(2hi\pi j) - (-1)^{2i}\sin(2hi\pi j)\right), \\ &= \frac{1}{4}\left(\cos(j\pi h) - 1\right)\sin(2hi\pi j), \\ &= C_{2,h}^j \left[v_1^j\right]_i.\end{aligned}$$

We thus have that at level $m = 1$, the fine-grid eigenvectors from level $m = 0$ are mapped by the restriction operator Z_1^T according to

$$Z_1^T v_0^j = C_{1,h}^j v_1^j, \ j = 1, 2, \ldots, n_1, \tag{A.2}$$
$$Z_1^T v_0^{n_0+1-j} = C_{2,h}^j v_1^j, \ j = 1, 2, \ldots, n_1. \tag{A.3}$$

Note that $v_1^j \in V_1 \ \forall j$. Additionally, note that n_1 vectors from $\mathcal{V}_0$ are mapped to zero which implies that the nullspace of Z_1^T has $\dim \mathcal{N}(Z_1^T) = n_1$. In order to move from $m = 1$ to $m = 2$, which maps $\mathcal{V}_1 \to V_2$, we apply Z_2^T. The mapping trajectory is given by the following diagram

$$\mathcal{I}_1^2 \circ \mathcal{I}_0^1 : \mathcal{V}_0 \xrightarrow{\mathcal{I}_0^1} \mathcal{V}_1 \xrightarrow{\mathcal{I}_1^2} V_2, \text{ where } \begin{array}{ccccc} \mathcal{V}_0 & \xrightarrow{\mathcal{I}_0^1} & V_1 & \circlearrowleft_{\mathcal{V}_1}^{\alpha_{\pi(j)}^1} & \\ & \searrow^{\mathcal{I}_1^2 \circ \mathcal{I}_0^1} & \downarrow{\scriptstyle \mathcal{I}_1^2} & & \\ & & V_2 & & \end{array}$$

We obtain $\mathcal{V}_0$ by first applying $\alpha^0_{\pi(j)}$ such that we get the ordering of the basis in pairs j, j'. The restriction operator $\mathcal{I}^1_0$ maps these basis vectors to V_1. Then in order to move to the second coarse space V_2, we again have to reorder the basis on V_1 by applying the automorphism $\alpha^1_{\pi(j)}$. After permuting the elements of the basis, we can apply $\mathcal{I}^2_1$. Consequently, the range of $\mathcal{I}^2_1$ is V_2. This is equivalent to having a composition of the linear transformations $\mathcal{I}^2_1 \circ \mathcal{I}^1_0$. Thus, in terms of the matrix representations, applying Z^T_2 gives

$$\begin{aligned}
\left[Z_2^T\left[Z_1^T v_0^j\right]\right]_i &= C^j_{1,h}\left(Z_2^T\left[v_1^j\right]_i\right),\\
&= \frac{1}{2}\left(1+\cos(j\pi h)\right)\left(Z_2^T \sin(2hi\pi j)\right),\\
&= \frac{1}{2}\left(1+\cos(j\pi h)\right)\left(\frac{1}{4}\sin((2i-1)2h\pi j) + 2\sin((2i)2h\pi j) + \sin((2i+1)2h\pi j)\right),\\
&= \left(\frac{1}{2}\left(1+\cos(j\pi h)\right)\right)\left(\frac{1}{2}\left(1+\cos(j\pi 2h)\right)\right)\sin(4hi\pi j),\\
&= C^j_{1,h} C^j_{1,2h}\left[v_2^j\right]_i.
\end{aligned}$$

As regards the complementary modes on level $m=1$ note that $\alpha^1_{\pi(j)} : V_1 \mapsto \mathcal{V}_1$ enables us to redefine $j' = n_1 + 1 - j$, where

$$\begin{aligned}
&[v_1^{j'}]_i = -(-1)^j \sin(i2hj\pi),\\
&i = 1, 2, \ldots n_1, \text{and } j = 1, 2, \ldots n_2.
\end{aligned} \tag{A.4}$$

Thus, applying the restriction operator to the complementary modes on $m=1$ gives

$$\begin{aligned}
\left[Z_2^T\left[Z_1^T v_0^{j'}\right]\right]_i &= C^j_{2,h}\left(Z_2^T\left[v_1^j\right]_i\right),\\
&= \frac{1}{2}\left(\cos(j\pi h)-1\right)\left(Z_2^T\left[v_1^j\right]_i\right),\\
&= \frac{1}{2}\left(\cos(j\pi h)-1\right)\left(\frac{1}{4}\left(\cos(j\pi h)\sin(2hi\pi j) - (-1)^{2i}\sin(2hi\pi j)\right)\right),\\
&= \left(\frac{1}{2}\left(\cos(j\pi h)-1\right)\right)\left(\frac{1}{2}\left(\cos(j\pi 2h)-1\right)\right)\sin(4hi\pi j),\\
&= C^j_{2,h} C^j_{2,2h}\left[v_2^j\right]_i.
\end{aligned}$$

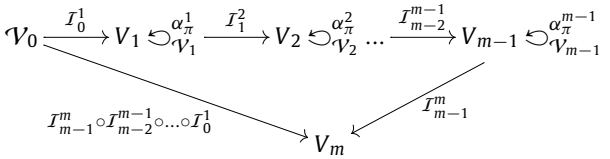
Note that $v_2^j \in V_2\ \forall j$. Consequently, using Z^T_1 to map from level $m=0$ to $m=1$ and Z^T_2 to map from level $m=1$ to $m=2$, results in the fine-grid eigenvectors being mapped in a nested application according to

$$\begin{aligned}
Z_2^T\left(Z_1^T v_0^j\right) &= C_1^j v_2^j,\ j = 1, 2, \ldots, n_2,\\
Z_2^T\left(Z_1^T v_0^{n+1-j}\right) &= C_2^j v_2^j,\ j = 1, 2, \ldots, n_2, \text{ where,}\\
C_1^j &= \left(\frac{1}{2}\right)^m \prod_{l=1}^{m}\left(1+\cos(j\pi 2^{l-1}h)\right) \text{ and,}\\
C_2^j &= \left(\frac{1}{2}\right)^m \prod_{l=1}^{m}\left(\cos(j\pi 2^{l-1}h)-1\right).
\end{aligned}$$

In this case, n_2 vectors from $\mathcal{V}_1$ are mapped to zero which implies that the nullspace of Z^T_2 has $\dim \mathcal{N}(Z^T_2) = n_2$. Consequently, in order to move to $m=3$ which maps $\mathcal{V}_2 \to V_3$, we can continue applying Z^T_3. From here, it is easy to see that for each subsequent level $m > 2$, consecutive application of the matrices Z^T_m is equivalent to the following linear mapping between the vector spaces $\mathcal{V}_m$

$$\mathcal{I}^m_{m-1} \circ \mathcal{I}^{m-1}_{m-2} \circ \ldots \circ \mathcal{I}^1_0 : \mathcal{V}_0 \overset{\mathcal{I}^1_0}{\to} \mathcal{V}_1 \overset{\mathcal{I}^2_1}{\to} \mathcal{V}_2 \ldots \mathcal{V}_{m-1} \overset{\mathcal{I}^m_{m-1}}{\to} \mathcal{V}_m,$$

which can be represented by the following diagram



We thus have $v_m^j \in V_m \ \forall j$, and in terms of the matrices, we therefore obtain

$$\begin{aligned}\left[\prod_{l=m}^{1} Z_l^T v_0^j\right]_i &= \left[Z_m^T Z_{m-1}^T \ldots \left[Z_2^T \frac{1}{2}\left(1+\cos(j\pi h)\right) v_1\right]\right]_i, \\ &= \left[Z_m^T Z_{m-1}^T \ldots \left[Z_3^T \frac{1}{4}\left(1+\cos(j\pi h)\right)\left(1+\cos(j\pi 2h)\right) v_2\right]\right]_i, \\ &= \left(\frac{1}{2}\right)^m \prod_{l=1}^{m}\left(1+\cos(j\pi 2^{l-1}h)\right)[v_m]_i = C_1^j [v_m^j]_i,\end{aligned}$$

for $j = 1, 2, \ldots, n_m$. Similarly, for the complementary part corresponding to $j' = n_{m-1} + 1 - j$ we obtain

$$\left[\prod_{l=m}^{1} Z_l^T v_0^{j'}\right]_i = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m}\left(\cos(j\pi 2^{l-1}h) - 1\right)[v_m]_i = C_2^j [v_m^j]_i.$$

To conclude, we obtain

$$\prod_{l=m}^{1} Z_l^T v_0^j = C_1^j v_m^j, \ j = 1, 2, \ldots, n_m, \tag{A.5}$$

$$\prod_{l=m}^{1} Z_l^T v_0^{j'} = C_2^j v_m^j, \ j = 1, 2, \ldots, n_m, \tag{A.6}$$

where $C_1^j = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m}\left(1+\cos(j\pi 2^{l-1}h)\right)$ and $C_2^j = \left(\frac{1}{2}\right)^m \prod_{l=1}^{m}\left(\cos(j\pi 2^{l-1}h) - 1\right)$.

Prolongation operator

The restriction operator was defined as the transpose of I_{m+1}^m, and thus we have that the matrix representation of the prolongation operator is given by Z_m. For the prolongation operator, we again start with $m = 1$ and take the basis V_1 as the prolongation operator works on a coarse-grid eigenvector on level m and maps it to a fine-grid counterpart on level $m-1$. We distinguish two cases; i is odd and i is even. We start with the first case

$$\begin{aligned}[Z_1 v_1^j]_i &= \frac{1}{4}\left(\sin(\frac{(i-1)2h\pi j}{2}) + \sin(\frac{(i+1)2h\pi j}{2})\right), \\ &= \frac{1}{4}\left(\sin((i-1)h\pi j) + \sin((i+1)h\pi j)\right), \\ &= \frac{1}{2}\cos(j\pi h)\sin(ih\pi j),\end{aligned} \tag{A.7}$$

for $j = 1, 2, \ldots, n_1$. If i is even, we obtain

$$[Z_1 v_1^j]_i = \frac{1}{2}\sin(\frac{2hi\pi j}{2}) = \frac{1}{2}\sin(hi\pi j) = \frac{1}{2}[v_0^j]_i. \tag{A.8}$$

Using Eq. (A.4), if we define $j' = n_{m-1} + 1 - j$, we can write Eq. (A.8) as

$$[Z_1 v_1^j]_i = \sin(hi\pi j) = -(-1)^i \sin(j\pi hi) = [v_0^{j'}]_i, \tag{A.9}$$

if i = odd. Thus, if i is odd, combining Eq. (A.4) and Eq. (A.9), gives

$$[Z_1 v_1^j]_i = \frac{1}{2}[v_0^{j'}]_i + \frac{1}{2}\cos(j\pi h)[v_0^j]_i = C_{1,h}^j [v_0^j, v_0^{j'}]_i,$$

for $j = 1, 2, \ldots, n_1$. Similarly, if i is even, we obtain

$$[Z_1 v_1^j]_i = -\frac{1}{2}[v_0^{j'}]_i + \frac{1}{2}\cos(j\pi h)[v_0^j]_i = C_{2,h}^j [v_0^j, v_0^{j'}]_i,$$

for $j = 1, 2, \ldots, n_1$. Note that $[v_0^j, v_0^{j'}]_i$ is an element of $\mathcal{V}_0$ and the coarse-grid eigenvectors are mapped by the interpolation operator Z_1 according to

$$\mathcal{I}_1^0 : V_1 \xrightarrow{\mathcal{I}_1^0} \mathcal{V}_0.$$

Also note that $\mathcal{R}(Z_1) \subset V_0$, and we have $V_0 = \mathcal{N}(Z_1^T) \bigoplus \mathcal{R}(Z_1)$. We now take $m = 2$, using the basis V_2. From the above, it follows that

$$[Z_2 v_2^j]_i = \frac{1}{2}[v_1^{j'}]_i + \frac{1}{2}\cos(j\pi 2h)[v_1^j]_i = C_{1,2h}^j [v_1^j, v_1^{j'}]_i, \ i \text{ is odd} \tag{A.10}$$

$$[Z_2 v_2^j]_i = -\frac{1}{2}[v_1^{j'}]_i + \frac{1}{2}\cos(j\pi 2h)[v_1^j]_i = C_{2,2h}^j [v_1^j, v_1^{j'}]_i, \ i \text{ is even}, \tag{A.11}$$

for $j = 1, 2, \ldots, n_2$ and $j' = n_1 + 1 - 1$. As the v_1^j's are the eigenvectors on level $m = 1$, we can rewrite the complementary indices j' in terms of j again by using

$$[v_1^{j'}]_i = -(-1)^i \sin(i2hj\pi), \tag{A.12}$$
$$i = 1, 2, \ldots n_1, \text{and } j = 1, 2, \ldots n_2.$$

Substituting Eq. (A.12) into Eq. (A.10) and Eq. (A.11) gives

$$[Z_2 v_2^j]_i = \frac{1}{2}[v_1^j]_i + \frac{1}{2}\cos(j\pi 2h)[v_1^j]_i = C_{1,2h}^j [v_1^j]_i, \ i \text{ is odd} \tag{A.13}$$

$$[Z_2 v_2^j]_i = -\frac{1}{2}[v_1^j]_i + \frac{1}{2}\cos(j\pi 2h)[v_1^j]_i = C_{2,2h}^j [v_1^j]_i, \ i \text{ is even}, \tag{A.14}$$

and $\mathcal{R}(Z_2) \subset V_1$, and we have $V_1 = \mathcal{N}(Z_2^T) \bigoplus \mathcal{R}(Z_2)$. Moving from $m = 1$ to $m = 0$ by left-multiplying Eq. (A.13) and Eq. (A.14) with Z_1 is now straightforward as we get the coefficient $C_{1,h}^j$ and $C_{2,h}^j$ times $[Z_1 v_1^j]_i$ from above. This corresponds to a composition of the linear transformations where at $\mathcal{V}_1$ we reorder the basis to V_1 using Eq. (A.12)

$$\mathcal{I}_1^0 \circ \mathcal{I}_2^1 : V_2 \xrightarrow{\mathcal{I}_2^1} \mathcal{V}_1 \xrightarrow{\mathcal{I}_1^0} \mathcal{V}_0, \text{ where } V_2 \xrightarrow{\mathcal{I}_2^1} \mathcal{V}_1 \circlearrowleft_{V_1}, \ \mathcal{I}_1^0 \circ \mathcal{I}_2^1 : V_2 \to \mathcal{V}_0, \ \mathcal{I}_1^0 : \mathcal{V}_1 \to \mathcal{V}_0.$$

From here it is easy to see that for $m > 2$ successive application gives

$$\begin{aligned}
\left[\prod_{l=1}^{l} Z_l v_m\right]_i &= \left[Z_1 Z_2 \ldots \frac{1}{2}\left(1 + \cos(j\pi 2^m h)\right)\left[Z_{m-1} v_{m-1}^j\right]\right]_i, \\
&= \left[Z_1 Z_2 \ldots \frac{1}{4}\left(1 + \cos(j\pi 2^m h)\right)\left(1 + \cos(j\pi 2^{m-1} h)\right)\left[Z_{m-2} v_{m-2}^j\right]\right]_i, \\
&= \left(\frac{1}{2}\right)^m \prod_{l=m}^{1}\left(1 + \cos(j\pi 2^l h)\right)[v_0^j]_i = C_1^j [v_0^j]_i, \text{ for } i \text{ is odd}.
\end{aligned} \tag{A.15}$$

Finally, if i is even we get $\left[\prod_{l=1}^{l} Z_l v_m\right]_i = \left(\frac{1}{2}\right)^m \prod_{l=m}^{1}\left(\cos(j\pi 2^l h) - 1\right)[v_0^j]_i = C_2^j [v_0^j]_i$ and $\mathcal{R}(Z_{m+1}) \subset V_m$, and we have $V_m = \mathcal{N}(Z_{m+1}^T) \bigoplus \mathcal{R}(Z_{m+1})$.

Composite mapping subspaces

Let us now take $B_m = \prod_{l=1}^{m-1} Z_l \prod_{l=m-1}^{1} Z_l^T$, and $\hat{B}_m = Z_m Z_m^T$. We furthermore let

$${}^t f^m : \mathcal{V}_0 \to V_m : \mathcal{I}_{m-1}^m \circ \mathcal{I}_{m-2}^{m-1} \circ \ldots \circ \mathcal{I}_0^1, \text{ and}$$
$$f^m : V_m \to \mathcal{V}_0, \text{ and}$$
$$g^m : \mathcal{V}_{m-1} \to \mathcal{V}_{m-1} : \mathcal{I}_m^{m-1} \circ \mathcal{I}_{m-1}^m$$

where ${}^t f^m$ is the transpose of the linear map f^m. Note that g^m is a automorphism. We can define

$$h^m : \mathcal{V}_0 \to \mathcal{V}_0 : f^m \circ {}^t f^m, \ f^m \in V_m,$$

to denote the composite linear mapping along the m-vectors spaces. Here ${}^t f^m$ maps elements of $\mathcal{V}_0$ to V_m and we can write $h^m : f^{m-1} \circ \left(g^m \circ {}^t f^{m-1}\right)$. This gives

$$\ker g^m = \{v_0^{j'} \in \mathcal{V}_0, : {}^t f^{m-1} v_0^j = 0\} \subset V_{m-1}, \text{ and}$$
$$\operatorname{Im} g^m = \{v_0^j \in \mathcal{V}_0 : {}^t f^{m-1} v_0^j \neq 0\} = V_{m-1}/\ker g^m \subset V_{m-1},$$

where j' are the complementary indices corresponding to $n_0 + 1 - j$. But then by definition and the fact that g^m is an automorphism, ${}^t f^{m-1} v_0^j$ must be an eigenvector of g^m. Given that we can write $V_{m-1} = \ker g^m \bigoplus \operatorname{Im} g^m$, the rank-nullity theorem furthermore tells us that $\dim(V_{m-1}) = \dim(\ker g^m) + \dim(\operatorname{Im} g^m) = n_m + n_m = n_{m-1}$. Thus, g^m must have n_m zero eigenvalues and n_m non-zero eigenvalues as the kernel of g^m is non-trivial. This leads to

$$\begin{aligned}(g^m \circ {}^t f^{m-1}) v_0^j &= g^m({}^t f^{m-1} v_0^j),\\ &= \lambda(g^m)({}^t f^{m-1} v_0^j) = \lambda(g^m) v_{m-1}^j,\end{aligned}$$

where $\lambda(g^m)$ denotes the scalar eigenvalue corresponding to g^m. Applying f^{m-1}, finally gives

$$\begin{aligned}f^{m-1} \circ (g^m \circ {}^t f^{m-1}) v_0^j &= f^{m-1}\left(g^m({}^t f^{m-1}\right) v_0^j),\\ &= \lambda(g^m) f^{m-1}({}^t f^{m-1} v_0^j) = \lambda(g^m)\lambda(h^{m-1}) v_{m-1}^j.\end{aligned}$$

Eigendecomposition of B_m

If B_{m-1} and $\hat{B}_m$ are the matrix representations of h^{m-1} and g^m respectively, then $\dim(\ker g^m) = \dim(\mathcal{N}(\hat{B}_m)) = n_m$, and $\dim(\operatorname{Im} g^m) = \dim(\mathcal{R}(\hat{B}_m)) = n_m$, and thus $\hat{B}_m$ has only n_m non-zero eigenvalues. But then B_m must also have n_m non-zero eigenvalues as well.

Appendix B. Proof of Theorem 4.2

Proof. On the basis $\mathcal{V}_0$ defined with respect to the finest level $m = 0$, we can block-diagonalize the coefficient matrix A in terms of a total of n_1 blocks with size 2×2. If we define the complementary index $j' = n_m + 1 - j = n_0 + 1 - j$, then each j-th respective block has the form

$$[\Lambda(A)]^j_{\mathcal{V}_0} = \begin{bmatrix} \lambda_A^j & 0 \\ 0 & \lambda_A^{j'} \end{bmatrix},$$

for $j = 1, 2, \ldots, n_1$. Moving to $m = 1$, we now start using $\mathcal{V}_1$ as E_1 resides in the coarse-space. After applying Z_1^T and Z_1, we obtain, for $j = 1, 2, \ldots n_1$, the 1×1 block

$$[\Lambda(E_1)]^j_{\mathcal{V}_1} = \left[Z_1^T A_0 Z_1\right]^j_{\mathcal{V}_1} = \begin{bmatrix} r_1^j & p_1^j \end{bmatrix} \begin{bmatrix} \lambda_A^j & 0 \\ 0 & \lambda_A^{j'} \end{bmatrix} \begin{bmatrix} r_1^j \\ p_1^j \end{bmatrix} = (r_1^j)^2 \lambda_A^j + (p_1^j)^2 \lambda_A^{j'}.$$

Thus, if we define $\lambda_{E_1}^j = (r_1^j)^2 \lambda_A^j + (p_1^j)^2 \lambda_A^{j'}$ for $j = 1, 2, \ldots, n_1$, then E_1 has block-diagonal form.

$$[\Lambda(E_1)]_{V_1} = \begin{bmatrix} \boxed{\lambda_{E_1}^1} & & & \mathbf{0} \\ & \boxed{\lambda_{E_1}^2} & & \\ & & \ddots & \\ \mathbf{0} & & & \boxed{\lambda_{E_1}^{n_1}} \end{bmatrix}.$$

Note that E_1 has no zero eigenvalues and dimension $n_1 \times n_1$. Consequently, we have a total of n_1 blocks with size 1×1 corresponding to each index j at level $m = 1$. To apply Z_2^T and Z_2 to E_1, we now need the 2×2 blocks. We can apply the permutation matrix corresponding to α_π with respect to V_1 such that we get the ordered basis $\mathcal{V}_1$. On this basis the block-diagonal form of E_1 is form

$$[\Lambda(E_1)]_{\mathcal{V}_1} = \begin{bmatrix} \boxed{\begin{matrix} \lambda_{E_1}^1 & 0 \\ 0 & \lambda_{E_1}^{1'} \end{matrix}} & & & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & & & \boxed{\begin{matrix} \lambda_{E_1}^{n_2} & 0 \\ 0 & \lambda_{E_1}^{n_2'} \end{matrix}} \end{bmatrix},$$

for $j = 1, 2, \ldots n_2$. Now, applying the block-diagonal form of Z_2^T and Z_2 to $[\Lambda(E_1)]_{\mathcal{V}_1}$ gives

$$\begin{bmatrix} \boxed{r_2^1 \quad p_2^1} & & & \mathbf{0} \\ & \boxed{r_2^2 \quad p_2^2} & & \\ & & \ddots & \\ \mathbf{0} & & & \boxed{r_2^{n_2} \quad p_2^{n_2}} \end{bmatrix} \begin{bmatrix} \boxed{\lambda_{E_1}^1 \quad 0 \\ 0 \quad \lambda_{E_1}^{1'}} & & & \mathbf{0} \\ & & \ddots & \\ & & & \boxed{\lambda_{E_1}^{n_2} \quad 0 \\ 0 \quad \lambda_{E_1}^{n_2'}} \\ \mathbf{0} & & & \end{bmatrix} \begin{bmatrix} \boxed{r_2^1 \\ p_2^1} & & & \mathbf{0} \\ & \boxed{r_2^2 \\ p_2^2} & & \\ & & \ddots & \\ \mathbf{0} & & & \boxed{r_2^{n_2} \\ p_2^{n_2}} \end{bmatrix}.$$

Note that $[\Lambda(E_1)]_{\mathcal{V}_1}$ has size $(n_1 \times n_1)$ and Z_2^T has size $(n_2 \times n_1)$. Thus, for $j = 1, 2, \ldots, n_2$ and $j' = n_1 + 1 - j$, each respective j-th block leads to the (1×1) block containing

$$[\Lambda(E_2)]_{\mathcal{V}_1}^j = \begin{bmatrix} r_2^j & p_2^j \end{bmatrix} \begin{bmatrix} \lambda_{E_1}^j & 0 \\ 0 & \lambda_{E_1}^{j'} \end{bmatrix} \begin{bmatrix} r_2^j \\ p_2^j \end{bmatrix} = (r_2^j)^2 \lambda_{E_1}^j + (p_2^j)^2 \lambda_{E_1}^{j'}.$$

From here it is easy to see that for $m > 2$, application of Z_m^T and Z_m recursively gives a j-th (1×1) block with $\lambda_{E_m}^j = (r_m^j)^2 \lambda_{E_{m-1}}^j + (p_m^j)^2 \lambda_{E_{m-1}}^{j'}$ for $j = 1, 2, \ldots, n_m$ and $j' = n_{m-1} + 1 - j$, where each j-th block has the form

$$[\Lambda(E_m)]_{\mathcal{V}_m}^j = \begin{bmatrix} \lambda_{E_m}^j & 0 \\ 0 & \lambda_{E_m}^{j'} \end{bmatrix}.$$

We can now combine Lemma 4.2.1 and the previous expression for the eigenvalues of E_m to block-diagonalize Q_m. We can now use the result from Lemma 4.2.1. This gives

$$[\Lambda(Q_m)]_{\mathcal{V}_0}^j = [\Lambda(\prod_{l=1}^{m} Z_l E_m^{-1} \prod_{l=m}^{1} Z_l^T)]_{\mathcal{V}_0}^j = \lambda_{E_m{}^j}^{-1} [\Lambda(B_m)]_{\mathcal{V}_0}^j = \lambda_{E_m{}^j}^{-1} \prod_{l=m}^{1} \left((r_l^j)^2 + (p_l^j)^2 \right),$$

for $j = 1, 2, \ldots, n_m$. □

References

[1] Y.A. Erlangga, C. Vuik, C.W. Oosterlee, On a class of preconditioners for solving the Helmholtz equation, Appl. Numer. Math. 50 (2004) 409–425.
[2] Y.A. Erlangga, C.W. Oosterlee, C. Vuik, A novel multigrid based preconditioner for heterogeneous Helmholtz problems, SIAM J. Sci. Comput. 27 (2006) 1471–1492.
[3] M.J. Gander, I.G. Graham, E.A. Spence, Applying GMRES to the Helmholtz equation with shifted Laplacian preconditioning: what is the largest shift for which wavenumber-independent convergence is guaranteed?, Numer. Math. 131 (2015) 567–614.
[4] P.-H. Cocquet, M.J. Gander, How large a shift is needed in the shifted Helmholtz preconditioner for its effective inversion by multigrid?, SIAM J. Sci. Comput. 39 (2017) A438–A478.
[5] O.G. Ernst, M.J. Gander, Why it is difficult to solve Helmholtz problems with classical iterative methods, in: Numerical Analysis of Multiscale Problems, Springer, 2012, pp. 325–363.
[6] H.C. Elman, O.G. Ernst, D.P. O'leary, A multigrid method enhanced by Krylov subspace iteration for discrete Helmholtz equations, SIAM J. Sci. Comput. 23 (2001) 1291–1315.
[7] S. Kim, S. Kim, Multigrid simulation for high-frequency solutions of the Helmholtz problem in heterogeneous media, SIAM J. Sci. Comput. 24 (2002) 684–701.
[8] I. Livshits, A. Brandt, Accuracy properties of the wave-ray multigrid algorithm for Helmholtz equations, SIAM J. Sci. Comput. 28 (2006) 1228–1251.
[9] O.G. Ernst, M.J. Gander, Multigrid methods for Helmholtz problems: a convergent scheme in 1D using standard components, in: Direct and Inverse Problems in Wave Propagation and Applications, De Gruyter, 2013, pp. 135–186.
[10] M.J. Gander, F. Magoules, F. Nataf, Optimized Schwarz methods without overlap for the Helmholtz equation, SIAM J. Sci. Comput. 24 (2002) 38–60.
[11] L. Conen, V. Dolean, R. Krause, F. Nataf, A coarse space for heterogeneous Helmholtz problems based on the Dirichlet-to-Neumann operator, J. Comput. Appl. Math. 271 (2014) 83–99.
[12] I.G. Graham, E.A. Spence, E. Vainikko, Recent results on domain decomposition preconditioning for the high-frequency Helmholtz equation using absorption, in: Modern Solvers for Helmholtz Problems, Springer, 2017, pp. 3–26.
[13] I. Graham, E. Spence, E. Vainikko, Domain decomposition preconditioning for high-frequency Helmholtz problems with absorption, Math. Comput. 86 (2017) 2089–2127.
[14] I. Graham, E. Spence, J. Zou, Domain decomposition with local impedance conditions for the Helmholtz equation, arXiv preprint, arXiv:1806.03731, 2018.
[15] M. Bonazzoli, V. Dolean, I. Graham, E. Spence, P.-H. Tournier, Domain decomposition preconditioning for the high-frequency time-harmonic Maxwell equations with absorption, Math. Comput. 88 (2019) 2559–2604.
[16] M.J. Gander, H. Zhang, A class of iterative solvers for the Helmholtz equation: factorizations, sweeping preconditioners, source transfer, single layer potentials, polarized traces, and optimized Schwarz methods, SIAM Rev. 61 (2019) 3–76.
[17] M.J. Gander, H. Zhang, Restrictions on the use of sweeping type preconditioners for Helmholtz problems, in: International Conference on Domain Decomposition Methods, Springer, 2017, pp. 321–332.
[18] A. Sheikh, Development of the Helmholtz Solver Based on a Shifted Laplace Preconditioner and a Multigrid Deflation Technique, PhD Thesis, TU Delft, Delft University of Technology, 2014.

[19] A. Sheikh, D. Lahaye, L.G. Ramos, R. Nabben, C. Vuik, Accelerating the shifted Laplace preconditioner for the Helmholtz equation by multilevel deflation, J. Comput. Phys. 322 (2016) 473–490.
[20] A. Sheikh, D. Lahaye, C. Vuik, On the convergence of shifted Laplace preconditioner combined with multilevel deflation, Numer. Linear Algebra Appl. 20 (2013) 645–662.
[21] Y.A. Erlangga, R. Nabben, On a multilevel Krylov method for the Helmholtz equation preconditioned by shifted Laplacian, Electron. Trans. Numer. Anal. 31 (2008) 3.
[22] V. Dwarka, C. Vuik, Scalable convergence using two-level deflation preconditioning for the Helmholtz equation, SIAM J. Sci. Comput. 42 (2020) A901–A928.
[23] D. Lahaye, C. Vuik, How to choose the shift in the shifted Laplace preconditioner for the Helmholtz equation combined with deflation, in: Modern Solvers for Helmholtz Problems, Springer, 2017, pp. 85–112.
[24] R. Nabben, C. Vuik, A comparison of deflation and the balancing preconditioner, SIAM J. Sci. Comput. 27 (2006) 1742–1759.
[25] H. Chen, P. Lu, X. Xu, A robust multilevel method for hybridizable discontinuous Galerkin method for the Helmholtz equation, J. Comput. Phys. 264 (2014) 133–151.
[26] H. Chen, H. Wu, X. Xu, Multilevel preconditioner with stable coarse grid corrections for the Helmholtz equation, SIAM J. Sci. Comput. 37 (2015) A221–A244.
[27] C.W. Oosterlee, A GMRES-based plane smoother in multigrid to solve 3D anisotropic fluid flow problems, J. Comput. Phys. 130 (1997) 41–53.
[28] Y.A. Erlangga, L.G. Ramos, R. Nabben, The multilevel Krylov-multigrid method for the Helmholtz equation preconditioned by the shifted Laplacian, in: Modern Solvers for Helmholtz Problems, Springer, 2017, pp. 113–139.
[29] F. Ihlenburg, I. Babuska, Finite element solution of the Helmholtz equation with high wave number part ii: the hp version of the fem, SIAM J. Numer. Anal. 34 (1997) 315–358.
[30] A. Deraemaeker, I. Babuška, P. Bouillard, Dispersion and pollution of the fem solution for the Helmholtz equation in one, two and three dimensions, Int. J. Numer. Methods Eng. 46 (1999) 471–499.
[31] K. Gerdes, F. Ihlenburg, On the pollution effect in FE solutions of the 3D-Helmholtz equation, Comput. Methods Appl. Mech. Eng. 170 (1999) 155–172.
[32] M. Baumann, R. Astudillo, Y. Qiu, E. Ang, M. Van Gijzen, R. Plessix, An MSSS-preconditioned matrix equation approach for the time-harmonic elastic wave equation at multiple frequencies, Comput. Geosci. 22 (2018) 43–61.