A parallel two-level polynomial Jacobi-Davidson algorithm for large sparse PDE eigenvalue problems

Yu-Fen Cheng[a,b], Feng-Nan Hwang[b,*]

[a] *National Center for High-performance Computing, Hsinchu 300, Taiwan*
[b] *Department of Mathematics, National Central University, Jhongli District, Taoyuan City 320, Taiwan*

Abstract

Many scientific and engineering applications require accurate, fast, robust, and scalable numerical solution of large sparse algebraic polynomial eigenvalue problems (PEVP's) that arise from some appropriate discretization of partial differential equations. The polynomial Jacobi-Davidson (PJD) algorithm has been numerically shown as a promising approach for the PEVP's to finding the interior spectrum. The PJD algorithm is a subspace method, which extracts the candidate eigenpair from a search space and the space updated by embedding the solution of the correction equation at the JD iteration. In this research, we develop and study the two-level PJD algorithm for PEVP's with emphasis on the application of the dissipative acoustic cubic eigenvalue problem. The proposed two-level PJD algorithm consists of two important ingredients: A good initial basis for the search space is constructed on the fine-level by using the interpolation of the coarse solution of the same eigenvalue problem in order to enhance the robustness of the algorithm. Also, an efficient and scalable two-level preconditioner based on the Schwarz framework is used for the correction equation. Some numerical examples obtained on a parallel cluster of computers are given in order to demonstrate the robustness and scalability of our PJD algorithm.

Key words: Acoustic wave equation, cubic eigenvalue problems, Jacobi-Davidson methods, domain decomposition, two-level Schwarz preconditioner, parallel computing.

1. Introduction

Polynomial eigenvalue problems (PEVP's) represent a broad range of applications in computational science and engineering modeled by partial differential equations (PDE's) [4, 8, 11, 12, 13, 14, 15, 27]. One of the fundamental numerical difficulties for solving such higher-degree polynomial eigenvalue problems is that no simple canonical form is available as the Schur form for the standard eigenvalue problem, which is useful for designing a new algorithm. Additional numerical challenges include the problem dimension in practice that is too large to prevent us from using a direct method because of an excess number of floating-point operations and limited memory source. Also, the power-type method or its variants can find efficiently the largest eigenvalues in the magnitude, but for real application, we are often interested in the search for some selected eigenvalues located within the interior of the spectrum, which corresponds to lower frequency modes. Hence,

*Corresponding author. Tel: +886-3-422-7151 Ext. 65110; Fax +886-3-425-7379
Email addresses: hwangf@math.ncu.edu.tw (Feng-Nan Hwang)

a computationally expensive shift-and-invert technique is required. Furthermore, the nonlinearity of the problems makes the theoretical analysis of an iterative method harder to establish.

On the other hand, the Jacobi-Davidson (JD) algorithm first proposed by Sleijpen and Van der Vorst [22, 23] for solving algebraic linear eigenvalue problems, then was extended to higher-degree polynomial cases [11, 12, 13, 14]. Now, JD has been shown a promising and powerful approach for solving such large, sparse polynomial PDE eigenvalue problems, especially for the cases that few selected interior spectrum is of interest. The JD algorithm, which belongs to a class of subspace iterative methods, consists of three key steps: extraction, correction, and expansion. More specifically, one first expands a subspace (or referred to as the search space) by adding a new basis vector, and then extracts the approximate eigenpair from the search space through the Rayleigh-Ritz procedure. To obtain a new basis vector for the search space, at each JD iteration, one needs to solve a large sparse linear system of equations, also known as the correction equation, by some preconditioned Krylov subspace type method, such as GMRES or CG methods [20].

As the computer power gets rapidly increased, the fast and efficient numerical solutions to PDE eigenvalue problems become more demanding. The question is naturally asked: how can we take advantages of that when the PDE model and its geometric/grid information are provided to enhance the robustness and efficiency of the JD algorithm? The past numerical evidence suggested that the robustness and the efficiency of the JD algorithm strongly depend on three factors: (1) the initial search space, (2) the Ritz pair selection strategy, and (3) the solution quality of the correction equation. Similar to the Newton-type methods for solving nonlinear systems, the JD algorithm is a locally convergent iterative method, i.e., if the initial guess is not close to the exact solution, the JD algorithm often exhibits some oscillatory behavior or even worse, the convergence failure happens. Without any available information about the target eigenpair, the normalized single constant vector or a vector with random values can be a choice. However, its convergence starting with such initial search space is unpredictable and unreliable. An alternative is to construct the initial search space using Fourier series expansion, but only for simple geometrical configuration or simple boundary condition setting. On the other hand, one of the important features of the JD algorithm is that at each JD iteration, only mild solution accuracy of the correction equation is required. As a result, an efficient correction equation solver plays an important role in the success of JD eigensolver for convergence; hence, designing such a correction equation solver is one of the currently active topics for JD research. A few related research works include: Feng [7] applied a multilevel JD method for the generalized eigenvalue problem with application in the finite element analysis of structural dynamic problems. A multigrid-type preconditioner was in conjugation with the FGMRES as numerical solution of the correction equation, where the incomplete Cholesky decomposition without fill-ins was employed as the pre- and post-smoothers, and the coarse grid was solved by a direct solver. Arbenz et al. [1] proposed a hybrid preconditioner combining a hierarchical basis method and an algebraic multigrid method for the correction equation in the JD algorithm; their parallel code for solving symmetric generalized Maxwell eigenvalue problems did not attain below 65% parallel efficiency using up to 16 processors.

In this paper, we develop and study the two-level JD algorithm for the large sparse polynomial PDE eigenvalue problems. We apply the two-level approach for the JD algorithm from two aspects. The first one is to construct a set of "*good*" initial basis vectors for the search space [11]. For many applications, the smooth eigenvector corresponding to the low-frequency mode of the vibration is of interest, which can be well represented by the coarse grid data. The second one is to construct the multilevel domain decomposition preconditioner for accelerating the convergence of a Krylov

subspace method as an efficient correction equation solver. Our proposed two-level preconditioner is based on the Schwarz framework [19, 25, 28], which has a long, successful history with linear elliptic PDE's. Recently, Schwarz methods, (one-level or multi-level), have been extended to more complex, coupled, nonlinear systems of multi-physics equations in the context of the Newton-Krylov algorithm. For example, a multilevel Schwarz preconditioner is used to accelerate the convergence of a Krylov subspace method for solving the Jacobian system in Newton iteration with applications in the numerical simulation of the semiconductor device [18]; that of the cardiac bioelectrical activity [21], and that of the blood-flow in compliant arteries [3, 29]. On the other hand, comparatively fewer studies address how a multi-level approach affects the performance of an eigensolver and most of them focus on linear or generalized EVP's [10, 24].

The rest of this paper is organized as follows. Section 2 describes the two-level PJD algorithm for solving the PDE eigenvalue problems in detail. Section 3 formulates a large, sparse cubic acoustic dissipative eigenvalue problem, which is derived from the finite element discretization of a rational acoustic EVP with an absorbing wall in the pressure variable form [4, 5]. Section 4 reports the study on the numerical performance of the proposed algorithm on a cluster of PCs. Section 5 concludes the paper.

2. Parallel two-level polynomial Jacobi-Davidson algorithm

In this section, we give a detailed description of the proposed parallel two-level polynomial Jacobi-Davidson algorithm for solving large sparse polynomial eigenvalue problems as follows.

$$\mathcal{A}^h(\lambda)u \equiv \sum_{\tau=0}^{d} \lambda^\tau A_\tau^h u = \left(\lambda^d A_d^h + \cdots + \lambda^2 A_2^h + \lambda A_1^h + A_0^h\right) u = 0, \tag{1}$$

where d is the degree of the polynomial, A_τ^h are the $n \times n$ coefficient matrices arising from some discretization of PDEs, e.g., finite element with fine grid size h and $\lambda \in \mathbb{C}$ is an eigenvalue, and $u \in \mathbb{C}^n$ is a corresponding eigenvector. Here, $\mathcal{A}^h(\lambda)$ is referred to as the global *fine* coefficient matrix. To simplify the presentation, we consider a two-level *geometric* Schwarz method for the scalar PDEs. An extension to the multi-component PDE systems is in a similar manner. We first introduce some notations needed for defining the two-level Schwarz preconditioner, including the coarse/fine grids, the subdomains, and the data transfer operators. See Fig. 1 for the graphical illustration of these components. Let Ω^l be the collection of grid points on the computational domain, Ω, on each level, where $l = H$ or $l = h$ for the coarse grid and the fine grid, with the element diameters, H and h, respectively. Each Ω^l is further partitioned into N_s nonoverlapping subdomains, Ω_i^l, $i = 1, ..., N_s$. To obtain the overlapping subdomain ones, we expand each Ω_i^l to a larger $\Omega_i^{l,\delta}$, Here, δ is an integer indicating the level of overlap. Nested grids and grid partitions are assumed, i.e., $\Omega^H \subset \Omega^h$ and $\Omega_i^H \subset \Omega_i^h$. Two classes of interpolation and restriction operators are needed. The first one is $R_i^{l,\delta}$ $((R_i^{l,\delta})^T)$ that transfers data from global spaces (each overlapping subdomain) to each overlapping subdomain (global space) on each level. The other one is I_h^H (I_H^h) that transfers data from the fine level (coarse level) to the coarse level (fine level). The global *coarse* coefficient matrix, $\mathcal{A}^H(\lambda)$, is either constructed through the Galerkin condition, $\mathcal{A}^H(\lambda) = I_h^H \mathcal{A}^h(\lambda) I_H^h$ or is obtained by using the same discretization as on the fine grid but now is on the coarse grid. Detailed discussion of these two approaches can be found Ref. [25]. Similarly, the associated subdomain coefficient matrices on each level, $\mathcal{A}_i^l$, is calculated by $\mathcal{A}_i^l = R_i^{l,\delta} \mathcal{A}^l(\lambda)(R_i^{l,\delta})^T$,

respectively.

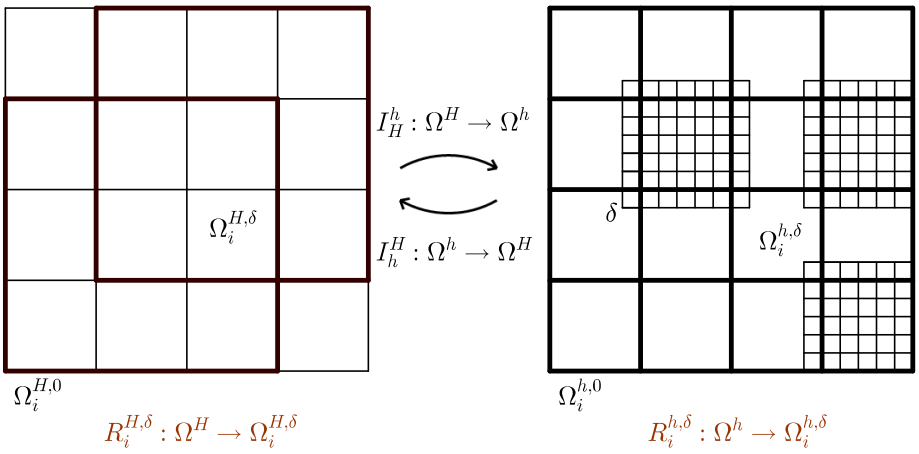

Figure 1: An illustration of the coarse/fine grids, the subdomains, and the data transfer operators.

Algorithm 1 Two-level Polynomial JD algorithm

Input: Both of the fine and coarse coefficient matrices, A_τ^h and A_τ^H for $\tau = 0, \ldots, d$, and the number of desired eigenvalues k.

Output: the desired eigenpairs (λ_j, u_j) for $j = 1, \ldots, k$

1: Set the initial search space for the coarse level, $V_{ini}^H = [\hat{u}^H]$. Possible choice is a constant vector or a random vector as described in Sec. 4.3.
2: Solve $\mathcal{A}^H(\lambda^H)u^H = 0$ on the coarse grid, by Algorithm 2, the PJD algorithm
3: Select the m eigenpairs close to the target
4: Set $u_q^h = I_H^h u_q^H$ for $q = 1, ..., m$.
5: Normalize $\hat{u}_q^h$ by $\hat{u}_q^h = u_p^h / \|u_q^h\|_2$
6: Set the initial search space for the fine level, $V_{ini}^h = [\hat{u}_1^h, ..., \hat{u}_m^h]$.
7: Solve $\mathcal{A}^h(\lambda)u^h = 0$ on the fine grid, by Algorithm 2, the PJD algorithm.

Algorithm 1 shows the two-level polynomial JD (PJD) algorithm for solving (1). Between lines 2 and 6 in Algorithm 1, we construct m initial basis in the search space on the fine grid by solving the same PEVP defined on the coarse grid using the PJD algorithm (See Algorithm 2). In the outer **for**-loop between lines 2 and 16 in Algorithm 2, we compute the k desired eigenpairs one-by-one using a locking technique. The inner loop **while**-loop between lines 4 and 12 in Algorithm 2 contains three key components:

1. an extraction of Ritz pairs from the search space by solving the projected companion eigenvalue problems,
2. a correction vector found by solving the correction equation, and
3. an expansion of the search space by embedding the correction vector into the space.

Below, we only highlight the component related to the numerical solution of the correction equation. The detailed discussion about other elements can be found in [12, 13].

Algorithm 2 Parallel Preconditioned Jacobi-Davidson Algorithm for Polynomial Eigenvalue Problems

Input: Coefficient matrices A_τ for $\tau = 0, \ldots, d$, the number of desired eigenvalues k, an initial orthonormal vector V_{ini} residual $atol$

Output: the desired eigenpairs (λ_j, u_j) for $j = 1, \ldots, k$

1: Set $V = [V_{ini}]$, $V_F = [\,]$, and $\Lambda = \emptyset$
2: **for** $j = 1$ to k **do**
3: Compute $W_\tau = A_\tau V$ and $M_\tau = V^* W_\tau$ for $\tau = 0, \ldots, d$
4: **while** (user-defined stopping criteria are not satisfied) **do**
5: Compute the eigenpairs (λ, s) of $(\sum_{\tau=0}^{d} \lambda^i M_\tau)s = 0$
6: Select the desired eigenpair (λ, s) with $\|s\|_2 = 1$ and $\lambda \notin \Lambda$
7: Compute $u = Vs$, $p = \mathcal{A}'(\lambda)u$, $r = \mathcal{A}(\lambda)u$
8: Solve the correction equation

$$B_d^{-1}\left(I - \frac{pu^*}{u^*p}\right)\mathcal{A}(\lambda)(I - uu^*)t = -B_d^{-1}r$$

approximately for $t \perp u$ by a Krylov subspace method with the one-level or two-level preconditioners as defined by Eq. (3) and Eq. (4).
9: Orthogonalize t against V, $v = t/\|t\|_2$.
10: Compute $w_\tau = A_\tau v$,

$$M_\tau = \begin{bmatrix} M_\tau & V^* w_\tau \\ v^* W_\tau & v^* w_\tau \end{bmatrix}$$

for $\tau = 0, \ldots, d$.
11: Expand $V = [V, v]$ and $W_\tau = [W_\tau, w_\tau]$, $\tau = 0, ..., d$.
12: **end while**
13: Set $\lambda_j = \lambda$, $u_j = u$, $\Lambda = \Lambda \cup \{\lambda_j\}$
14: Perform locking by orthogonalizing u_j against V_F; Compute $u_j = u_j/\|u_j\|_2$; Update $V_F = [V_F, u_j]$
15: Choose an orthonormal matrix $V_{ini} \perp V_F$; Set $V = [V_F, V_{ini}]$
16: **end for**

1. **Correction vector.** As shown in line 7 and 8 of Algorithm 2, we compute

$$p = \mathcal{A}'(\lambda)u,$$

where $\mathcal{A}'(\lambda) = \sum_{\tau=1}^{d} \tau\lambda^{\tau-1}A_\tau$ and then solve the correction equation

$$\left(I - \frac{pu^*}{u^*p}\right)\mathcal{A}(\lambda)(I - uu^*)t = -r \tag{2}$$

for a correction vector t in each inner **while**-loop and then append the normalized correction vector to V. Here, the correction equation (2) is solved approximately by a Krylov subspace method (e.g., GMRES) with the preconditioning operator, B_d^{-1}, where

$$B_d = \left(I - \frac{pu^*}{u^*p}\right)B(I - uu^*) \approx \left(I - \frac{pu^*}{u^*p}\right)\mathcal{A}(\lambda)(I - uu^*)$$

and B is an approximation of $\mathcal{A}(\theta)$. In practice, there is no need to explicitly form B_d when solving $B_d z = y$ with $z \perp u$ for a given y, as it can be done equivalently by computing

$$z = B^{-1}y - \eta B^{-1}p, \text{ with } \eta = \frac{u^*B^{-1}y}{u^*B^{-1}p}.$$

2. **Two-level hybrid restricted Schwarz preconditioner, $\tilde{y} = B^{-1}y$ (or $\tilde{p} = B^{-1}p$).** Huang et al. [12] employed the one-level restricted additive Schwarz (RAS_f) preconditioner [6] defined on the fine grid, which is written as

$$B_{RAS\_f}^{-1} = \left(\sum_{i=1}^{N_s} \left(R_i^{h,0}\right)^T \left(\mathcal{A}_i^h\right)^{-1} R_i^{h,\delta}\right). \tag{3}$$

Such preconditioner numerically exhibits good strong scalability, but it is less efficient for large-scale problems. In this work, we consider a two-level approach based on Schwarz framework, which approximates B^{-1} by,

$$B_{HRAS}^{-1} = I_H^h(\mathcal{A}^H)^{-1}I_h^H + (I - I_H^h(\mathcal{A}^H)^{-1}I_h^H\mathcal{A}^h)B_{RAS\_f}^{-1} \tag{4}$$

By using the multigrid terminology, this preconditioner can be viewed as a two-grid scheme, where the one-level fine-grid additive restricted additive Schwarz preconditioner is used as a pre-smoother (and post-smoother) to order to eliminate the high-frequency mode error. Alternatively, a two-level cascade-type Schwarz preconditioner can be used, which has been considered in [30]. The parallelization of a coarse grid correction, $(\mathcal{A}^H)^{-1}$ is necessary and two strategies are employed: (a) **A direct inexact approach**, which approximates $(\mathcal{A}^H)^{-1}$ by the one-level restricted additive Schwarz (RAS_c) preconditioner defined on the coarse grid,

$$B_{RAS\_c}^{-1} = \sum_{i=1}^{N_s} \left(R_i^{H,0}\right)^T \left(B_i^H\right)^{-1} R_i^{H,\delta}, \tag{5}$$

where B_i^H could be an LU or an ILU(k) decomposition of $\mathcal{A}_i^H (= R_i^{H,\delta} \mathcal{A}^H(\lambda) (R_i^{H,\delta})^T)$. (b) **An iterative approach**, which solves the coarse grid problem, $\mathcal{A}^H z^c = w^c$, by some parallel iterative methods, such as parallel GMRES, with a coarse additive Schwarz preconditioner (5). For any given coarse grid vector w^c,

$$z^c = (\mathcal{A}^H)^{-1} w^c$$

is understood as an approximate solution of the following preconditioned linear system of equations

$$\mathcal{A}^H \left[\sum_{i=1}^{N} (R_i^{H,\delta})^T (\mathcal{A}_i^H)^{-1} R_i^{H,\delta} \right] y^c = w^c, \text{ with } z^c = \left[\sum_{i=1}^{N} (R_i^H)^T (A_i^H)^{-1} R_i^c \right] y^c,$$

which satisfies the condition

$$\left\| w^c - \mathcal{A}^H z^c \right\|_2 \leq \epsilon_C \left\| w^c \right\|_2 .$$

or the maximum iteration number of the parallel iterative method reaches ($m_c^{\max}$).
Note that for any given w^c, the computation of z^c can be carried out in parallel using all N_s processors. $\mathcal{A}^H$ is not exactly a matrix, but a preconditioned iterative solver.

3. Cubic acoustic eigenvalue problem

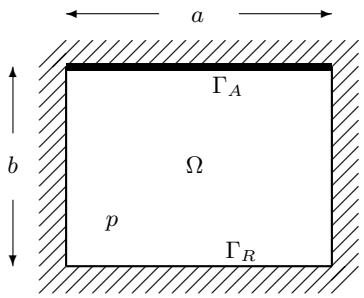

Figure 2: The benchmark problem that models fluid in a cavity with an absorbing wall.

As shown in Fig. 2, consider the acoustics in a cavity, defined on $\Omega = [0, a] \times [0, b]$ in $\mathbb{R}^2$ with $a > 0$ and $b > 0$. We apply the two-level PJD algorithm to the dissipative acoustic EVP [17] given by

$$\frac{\lambda^2}{c^2} p = \Delta p \text{ in } \Omega \subset \mathbb{R}^2, \tag{6}$$

with the absorbing (homogeneous Neumann) boundary condition on the top wall and the reflecting (Robin) boundary conditions on the rest of walls,

$$\frac{\partial p}{\partial \mathbf{n}} = \frac{-\rho \lambda^2}{\alpha + \lambda \beta} p \text{ on } \Gamma_A \text{ and} \tag{7}$$

$$\frac{\partial p}{\partial \mathbf{n}} = 0 \text{ on } \Gamma_R, \tag{8}$$

respectively. Here, (λ, p) forms the complex eigenpair of (6). The imaginary and real parts of λ are the angular frequency and the decay rate of sound disturbances, respectively, and p is the pressure perturbation. Besides, ρ is the density of fluids, c is the speed of sound, α is the elastic coefficient, β is the damping coefficient due to viscous effect, and $\mathbf{n}$ is an outward normal vector. Note that the acoustic EVP is derived from the acoustic wave equation [16] under the time-harmonic assumption, which is a simplified mathematical model of inviscid, compressible, and barotropic fluids with small perturbations. A bilinear Galerkin finite element discretization on a given quadrilateral mesh $\mathcal{T}^h = \{\mathcal{K}\}$ leads to a rational EVP due to the frequency-dependent impedance (8). By multiplying the common denominators, the model cubic polynomial EVP then derived from the rational EVP is as follows: Find some complex eigenpair(s) (λ, u) to satisfy

$$\mathcal{A}(\lambda)u = (\lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0)u = 0, \tag{9}$$

where $A_0 = \alpha K$, $A_1 = \beta K$, $A_2 = \alpha M + C$, and $A_3 = \beta M$. Here, the coefficient matrices

$$M = \frac{1}{c^2}\int_\Omega \Phi\Phi^T\, dx,\ C = \rho\int_{\Gamma_A} \Phi\Phi^T\, dx, \text{ and } K = \int_\Omega \nabla\Phi\cdot\nabla\Phi^T\, dx$$

stand for mass, damping, and stiffness, respectively. In general, M and K are both real symmetric positive definite matrices, and C is a real positive semidefinite matrix containing diagonal blocks. Here, $\Phi = [\phi_1, \phi_2, ..., \phi_n]^T$ is a vector of global bilinear shape functions, where n is the number of total interior nodes, including all boundary nodes on Γ_A and Γ_R. Due to rationalization, the cubic eigenvalue problem (9) has extraneous roots with algebraic multiplicity n all located at $\lambda = -\alpha/\beta$. In the case where no Dirichlet-type boundary is presented, a zero eigenvalue exists for Eq. (9). These two unphysical eigenvalues are considered to be potential troublemakers in numerical computations that either cause the convergence failure of an iterative method or slow down its convergence. Some oscillatory convergence behaviors are often observed. Note that $\mathrm{Re}(\lambda) < 0$ for all eigenvalues, where $\lambda \neq 0$, which is consistent with the theoretical prediction or physical observation that the acoustic vibrations are damped due to the effect of the viscoelastic materials. We show this fact below.

Proposition 3.1. *Let (λ, u) be an eigenpair of the cubic acoustic eigenvalue problem, $(\lambda^3 A_3 + \lambda^2 A_2 + \lambda A_1 + A_0)u = 0$, where $A_0 = \alpha K$, $A_1 = \beta K$, $A_2 = \alpha M + C$, and $A_3 = \beta M$. Assume that M is symmetric positive definite, C and K are symmetric positive semi-definite. Then real part of all nonzero eigenvalues are negative.*

Proof. Consider two cubic functions, $F_1(\lambda)$ and $F_2(\lambda)$, defined as

$$F_1(\lambda) \equiv (u^* A_3 u)\lambda^3 + (u^* A_2 u)\lambda^2 + (u^* A_1 u)\lambda + (u^* A_0 u) = 0. \tag{10}$$

and

$$\begin{aligned} F_2(\widetilde{\lambda}) &\equiv (\alpha + \beta\widetilde{\lambda})\left[(u^* M u)\widetilde{\lambda}^2 + (\frac{1}{\beta}u^* C u)\widetilde{\lambda} + (u^* K u - \frac{\alpha}{\beta^2}(u^* C u))\right] \\ &= F_1(\widetilde{\lambda}) - \frac{\alpha^2}{\beta^2}u^* C u = 0 \end{aligned} \tag{11}$$

Next we rewrite the quadratic term in Eq. (11) as $a\lambda^2 + b\lambda + c = 0$, where a, b, and c are given

$$\begin{aligned} a &= u^* M u, \\ b &= \frac{1}{\beta} u^* C u, \\ c &= u^* K u - \frac{\alpha}{\beta^2} u^* C u, \end{aligned}$$

Here, $a > 0$ and $b \geq 0$, since M is symmetric positive definite, C is positive semi-definite, and $\beta > 0$. Since $\lambda = -\alpha/\beta$ is the common root of $F_1(\lambda)$ and $F_2(\widetilde{\lambda})$ and by comparing the coefficients, the sum of all roots for these two cubic functions are identical, we have

$$\mathrm{Re}(\lambda) = \mathrm{Re}(\widetilde{\lambda}) = -\frac{b}{2a} \leq 0.$$

Next, we claim that there is no purely imaginary root. Assume that q is a root of the cubic polynomial Eq. (10), where $q \in \mathbb{R}$ and $q \neq 0$. Hence, we have

$$\begin{cases} \alpha u^* M u - q^2 u^* C u + \alpha u^* K u = 0 \text{ (real part)} \\ -q^3 \beta u^* M u + q \beta u^* K u = 0 \text{ (imaginary part)} \end{cases}$$

This implies $u^* C u = 0$, but it is only true when $\lambda = -\alpha/\beta$, which contracts the assumption that λ is purely imaginary. Hence, we conclude that the real parts of nonzero eigenvalues are negative, i.e., $\mathrm{Re}(\lambda) < 0$. □

4. Numerical results and discussions

4.1. Numerical experiment setup

Table 1 lists all physical parameters, such as the fluid density, and the elastic and damping parameters used for the numerical experiment in the following subsections. A sequence of uniformly-refined regular bilinear finite element meshes is employed. We implemented the two-level PJD eigensolver built on top of the Portable, Extensible Toolkit for Scientific Computation (PETSc) [2] and the Scalable Library for Eigenvalue Problem Computation (SLEPs) [9].

Parameter	Symbol	Value	Unit	Parameter	Symbol	Values	Unit
domain width	a	1.00	m	domain height	b	0.75	m
fluid density	ρ	1.00	kg/m^3	sound speed	c	340.00	m/s
elasticity	α	50000.00	N/m^3	damping	β	200.00	Ns/m^3

Table 1: Physical parameters used for the numerical experiments.

We claim the convergence of the PJD eigensolver if $\|r_f\|_2 < atol_f$ or $\|r_f\|_2 < rtol_f \|r_0\|_2$ where $rtol_f$ and $atol_f$ are set to be 10^{-10} and 10^{-8}, respectively. The PJD eigensolver restarts every 30 steps and retains five original best vectors with the smallest magnitudes in the search space during the restarting procedure. For the acoustic problem in industrial and scientific applications, the eigenvalues of interest are often interior ones, which correspond to low-frequency modes within the range of $0 < \frac{\mathrm{Im}(\lambda)}{2\pi} < 600$Hz. Hence, the target eigenvalue for the PJD eigensolver was set to be the range. All computations were performed in complex arithmetic double precision and were

run on a cluster of computers. We report various numerical results and discuss our findings in the following subsections.

4.2. Convergence behavior of computed eigenpairs

The closed solutions for this particular problem (6) along with the boundary conditions (8) can be derived from the technique of separation variables [5] in which (η_m, λ) satisfy

$$\eta_m^2 = \frac{\lambda^2}{c^2} + \frac{m^2\pi^2}{a^2} \text{ and } \eta_m \tanh \eta_m b = \frac{-\rho\lambda^2}{\alpha + \lambda\beta}$$

for a particular $m \in \mathbb{N}$. We obtain (η_m, λ) numerically by Newton's method and refer to them as the semi-analytical solutions and the corresponding eigenvectors are

$$p(x, y) = \cos(\frac{m\pi x}{a}) \cosh(\eta_m y)$$

The implementation of Galerkin finite element methods for the test problem has been carefully validated, and the detailed grid-independent test results can be found in [12]. Also the numerical evidence in [12] shows that the convergence behavior for the eigenvalues of the cubic acoustic problem closely follows the theoretical prediction for the elliptic eigenvalue problems [26], i.e.,

$$\lambda_i \leq \lambda_i^h \leq \lambda_i + c_1 h^2 \lambda_i^2, \tag{12}$$

where λ_i is the i-th exact eigenvalue and λ_i^h is the i-th computed eigenvalue. Some consequences of the theory and the numerical results are summarized below.

1. The optimal 2nd order rate of convergence is achieved.
2. For fixed i, the sequence of the absolute value of the eigenvalues $|\lambda_i^h|$ is bounded from below, and $|\lambda_i|$ is a lower bound. The absolute value of the eigenvalue $|\lambda_i^h|$ decreases and approaches to the exact eigenvalue $|\lambda_i|$ when the grid is refined. i.e. $|\lambda_i| \leq \ldots \leq |\lambda_i^{\frac{h}{4}}| \leq |\lambda_i^{\frac{h}{2}}| \leq |\lambda_i^h|$.
3. The computed eigenvalues corresponding to the low-frequency modes are more accurate than those corresponding to the high-frequency modes.

Target	32×24	64×48	128×96	256×192	512×384	1024×768	Rate
$1281i$	2.0270E-02	1.0059E-02	5.0094E-03	2.4996E-03	1.2485E-03	6.2391E-04	1.006
$2250i$	2.4396E-02	1.2063E-02	5.9965E-03	2.9893E-03	1.4924E-03	7.4562E-04	1.009
$3023i$	2.5901E-02	1.2776E-02	6.1356E-03	3.1616E-03	1.6165E-03	7.8839E-04	1.009

Table 2: The convergence analysis for eigenfunctions in magnitude with respect to different grid sizes.

On the other hand, Table 2 presents the L_2-norm errors for the complex eigenfunctions in magnitude for different grid sizes and its convergence rate. Three target eigenvalues are $1281i$, $2250i$, and $3023i$ corresponding to low, medium, and high frequencies, respectively. The eigenfunction in magnitude converges linearly, and its convergence rate does not depend on the frequency, which is slightly better than the predicted error estimate for the numerical eigenfunction of the acoustic eigenvalue problem in the displacement-pressure formulation [5]:

$$\| |p| - |p^h| \|_0 \leq Ch^r,$$

where r is a real number of between 0 and 1.

4.3. A comparison of initial search space constructions

We investigate the choices of an initial basis in the search space affects on the overall performance of the PJD algorithm. For the purpose of the comparison, addition to the proposed ***two-grid approach***, where an interpolated coarse solution is used as an initial vector on the fine grid, we also consider a random initial vector, using what is referred to as the ***random vector approach***. In this case, the initial search space is set to be $V_{ini} = [g]_{n\times 1}$, where $g = \text{normalize}(v + \text{normalize}(s))$. Here, the normalized constant, $v = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$, with a small random perturbation s, whose values drawn from the standard uniform distribution on the open interval $(0, 1)$. Note that a constant vector is one of the simplest choices, but such naive initial vector is not applicable for our acoustic eigenvalue problem when one of the eigenvalues is zero. Using such naive initial vector will make JD to converge to the trivial solution since the constant eigenvector is the one corresponding to zero eigenvalue. Hence, we perturb the constant vector by adding a random vector to assure such new perturbed vector, which is linearly independent to a constant vector.

	Random vector approach		Two-grid approach	
Target: $1281i$				
grid	JDIts (Avg.Corr.Its)	time	JDIts (Avg.Corr.Its.)	time
128×96	10(25.0)	0.27	3(25.0)	0.09
256×192	13(25.0)	2.16	5(25.0)	0.80
512×384	18(25.0)	10.69	8(25.0)	5.22
1024×768	28(25.0)	112.40	14(25.0)	40.29
Target: $3023i$				
grid	JDIts (Avg.Corr.Its)	time	JDIts(Avg.Corr.Its)	time
128×96	17(25.0)	0.49	8(25.0)	0.21
256×192	20(25.0)	3.11	15(25.0)	2.68
512×384	25(25.0)	17.49	21(25.0)	17.51
1024×768	130(25.0)	399.10	21(25.0)	69.81

Table 3: Effect of the selection of an initial search space on the convergence of PJD algorithm with respect to the grid size. $1281i$ and $3023i$ are set to be the targets.

Table 3 shows a comparison of the iteration number and the computing time of the two-grid and the random vector approaches for different sizes of the fine grids. The computational cost for constructing the coarse coefficient matrices and finding the coarse eigenvector are excluded. Two target eigenvalues are $1280i$ (lower frequency mode) and $3023i$ (higher frequency mode). Here, the correction equation is solved by the 25-step *fine* restricted additive Schwarz preconditioned GMRES using 12 processors. A fixed 32×24 coarse grid is used for the two-grid PJD algorithm.

As expected, the two-grid approach takes lesser iterations for convergence as well as its corresponding computing times, than the random vector approach. On the other hand, for the two-grid approach, the number of PJD iterations still grows as the ratio of the fine to the coarse grids increases. This result suggests that only employing good initial search space is not sufficient; more efficient correction equation solvers may be needed.

We look into more details of the two-level approach for setting the initial search space. Table 4 shows the convergence history of the Ritz value, the JD residual, and the number of iterations for solving the correction equation. We first solve the 32×24 coarse grid problem by the PJD algorithm using 12 processors. From the top part of the table, we find that at the beginning, the

level=H, coarse grid: 32×24			
i	JD iter Ritz-value	residual	KSP iter
0	-4.4613e+02+2.2480e+03i	2.18E+04	25
1	-3.0114e+02+2.1833e+03i	1.63E+03	25
2	-3.6043e+02+1.9630e+03i	3.09E+04	25
3	-9.7772e+01+1.4879e+03i	9.93E+03	22
4	-8.9670e+01+1.2873e+03i	7.55E+02	22
5	-8.9954e+01+1.2817e+03i	1.72E+00	22
6	-8.9952e+01+1.28175e+03i	2.42E-05	
level=h, fine grid: 256×192			
i	JD iter Ritz-value	residual	KSP iter
0	-8.9952e+01+1.2817e+03i	1.12E+02	25
1	-8.9952e+01+1.2813e+03i	1.18E-02	25
2	-8.9953e+01+1.2813e+03i	4.17E-04	25
3	-8.9953e+01+1.2813e+03i	3.74E-04	25
4	-8.9953e+01+1.2813e+03i	4.30E-05	25
5	-8.9953e+01+1.28135e+03i	4.45E-06	

Table 4: Two-grid PJD monitor for the cubic acoustic eigenvalue problem.

convergence of the Ritz-value on the coarse grid exhibits the oscillatory behavior, mainly because the random constant initial search space is used. Later, once the PJD algorithm locates the right Ritz value (for example, at 4th JD iteration in this particular case), the Ritz value converges quadratically. Then we interpolate the converged coarse eigenvector to fine grid and use it as the basis for the fine grid initial search space. As a result, the Ritz value converges very fast, taking only 3 iterations to achieve 5- to 6- digit accuracy. The rest of the PJD iterations mainly correct the error of the eigenvector.

Furthermore, the initial JD residual on the fine grid is not small compared to that of the last JD iterations on the coarse grid (10^2 compared to 10^{-5}). This seems to be counterintuitive: A smooth eigenvector on the fine grid should be well represented by the data on the coarse grid. For example, let us consider a special case with only the Neumann boundary condition imposed. In this case, without the damping effect, the cubic acoustic eigenvalue problem is reduced to the generalized eigenvalue problem by setting $\omega = \lambda^2$, and all eigenvalues are purely real and negative. Table 5 shows the convergence history of the Ritz values on the coarse and fine grids, respectively. Different from the cubic eigenvalue problem, in this case, the interpolated coarse eigenvector (See Fig. 3) is a good initial guess for the fine grid so that the initial residual starts at 10^{-4}, which means that both of the eigenvalue and eigenvector have already been corrected by coarse grid. Therefore, PJD converges quite fast on the fine grid.

But for the cubic acoustic eigenvalue problem, we have a different story. Fig. 4 includes nine eigenvector plots for the coarse grid eigenvectors (first row), the interpolated coarse grid eigenvector (second row), and the fine grid eigenvector (third row) in the magnitude (first column), their real part (second column), and their imaginary part (third column). Observing from these plots, we find that three vectors have the same magnitude, but due to complex-arithmetic, the interpolated eigenvector and the fine grid eigenvector are quite different, if we look at them in real part and imaginary part individually. That explains why the initial JD residual is not small, and the coarse eigenvalue is quite close to the fine eigenvalue.

level= H, coarse grid: 32×24.			
i	JD iter Ritz-value	residual	KSP iter
0	-9.4425e+04+0.0000e+00i	4.65E-02	17
1	-2.8834e-02+0.0000+00i	5.00E-06	19
2	-1.5090e+06+0.0000+00i	3.33E-02	23
3	-1.1588e+06+0.0000e+00i	5.07E-03	22
4	-1.1418e+06+0.0000e+00i	1.35E-05	22
5	-1.14184e+06+0.0000e+00i	3.17E-10	
level= h, fine grid: 256×192			
i	JD iter Ritz-value	residual	KSP iter
0	-1.1418e+06+0.0000e+00i	3.90E-04	25
1	-1.1409e+06+0.0000e+00i	1.89E-09	

Table 5: Two-grid PJD monitor for the generalized acoustic eigenvalue problem, which is no damping effect.

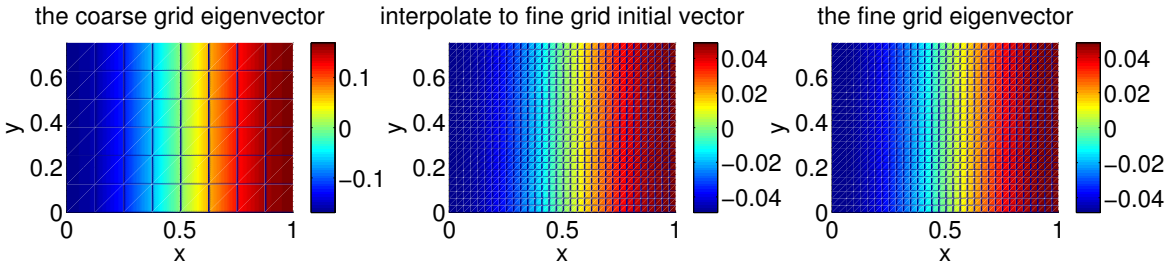

Figure 3: Reduced acoustic generalized eigenvalue problem. The contour plot for the coarse grid eigenvector (left), the interpolate coarse vector to the fine grid (middle), and the fine grid eigenvector (right). These three vectors look almost identical.

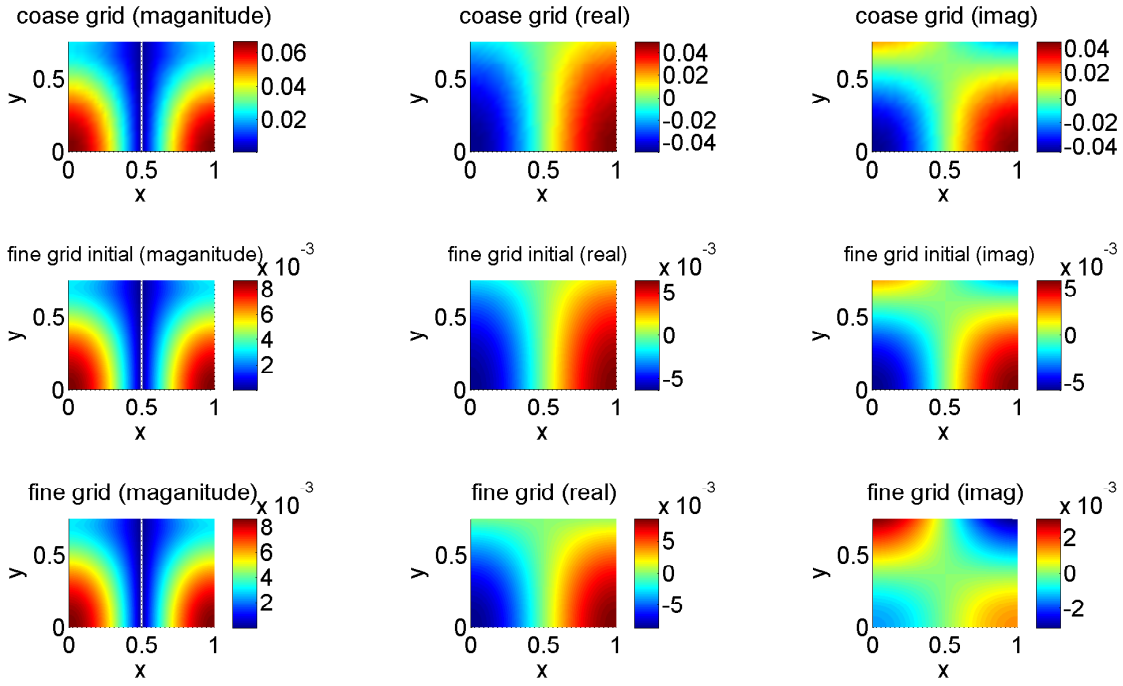

Figure 4: Cubic acoustic eigenvalue problem. The contour plots for the coarse eigenvector (first row), interpolated vector from the coarse grid to the fine grid (second row), the fine eigenvector (third row). The magnitude of the eigenvectors (first column), the real part of the eigenvectors (second column), and the imaginary part of the eigenvectors (third column).

4.4. A comparison of the coarse solution quality for the two-level method

We next study how the coarse solution quality related to the two-level Krylov-Schwarz solver for the correction equation impacts on the overall PJD convergence. For all test runs, a fixed 25-step GMRES method is employed for solving the correction equation. A fixed coarse grid: 32×24 is used. The HRAS preconditioner defined in (4) for performing the operator B^{-1} with a vector. Here, the RAS_c preconditioner or a coarse RAS_c preconditioned GMRES is used for the coarse grid correction. For the RAS_c preconditioned GMRES method, we adopt two kinds of stopping strategies, using the relative residual (10^{-2} and 10^{-5}) or a fixed number of iterations (5 or 10 steps). An incomplete sparse LU decomposition with three levels of fill-ins was employed as a subdomain solver and the overlapping size was set to 1 for all Schwarz type preconditioners. For the coarse grid eigenvalue problem, a random vector was used for the initial search basis, one may or may not always find the eigenvalue close to the one on the fine grid. Hence, we found two coarse eigenvalues, then chose the one closest to our target as our initial search candidate on the fine grid. Table 6 indicates the HRAS preconditioner with RAS_c is more efficient compared to the one with RAS_c preconditioned GMRES with a variant of the stopping criterion, especially for the finest grid case with a large number of processors. The number of JD iterations is mildly dependent on the number of processors.

Coarse part	$(\mathcal{A}^H)^{-1}$	RAS_c-GMRES				RAS_c
Fine grid	relative tol.	1.00E-05	1.00E-02	—	—	—
np	maximum ite.	—	—	10	5	—
256×192	JDit(Avg.KSPit)	2(8.7)	2(7.3)	7(8.0)	4(9.8)	2(25.0)
12	Time	0.32	0.25	0.69	0.48	0.43
512×384	JDit(Avg.KSPit)	2(15.7)	2(1.7)	6(11.7)	4(15.2)	2(25.0)
48	Time	0.74	0.51	1.09	1.65	0.72
1024×768	JDit(Avg.KSPit)	2(25.0)	2(17.7)	8(17.7)	4(23.2)	3(25.0)
192	Time	7.05	3.46	5.73	3.74	2.77
2048×1536	JDit(Avg.KSPit)	3(25.0)	3(24.8)	6(25.0)	5(25.0)	6(25.0)
768	Time	12.50	7.03	7.89	5.32	5.11

Table 6: Effect on the quality of coarse grid solution. The coarse problem is solved by one-level restricted additive coarse Schwarz preconditioned GMRES with different stopping conditions and one step one-level restricted additive coarse Schwarz Richardson iterative method. Coarse grid: 32×24. The degree of overlapping is set to 1 and the subdomain problem is solved by ILU(0) for both cases.

4.5. Parallel performance analysis

In this section, we investigate the parallel performance of the proposed two-level PJD algorithms based on the fixed-problem size scalability as the metric on a cluster of computers. A fixed 32×24 coarse grid was used. Let the time running the parallel code on 12 cores be denoted by T_{12} and the time running the same code on *np* processors be denoted by T_n; then the speedup is defined by

$$\mathrm{Sp} = \frac{T_{12}}{T_{np}}$$

Note that the test problem is too large to fit into a single core, hence we used the timing results obtained by using 12 cores as the baseline for comparison. Efficiency is then calculated using the following formula:

$$\mathrm{Ef} = \frac{T_{12} \times 12}{T_n \times np}.$$

From Table 7, we found the following:

1. Both methods are quite scalable in terms of the total number of iterations (outer iterations times inner iterations). All remain nearly constant as the number of cores increases.
2. About the timing results, without taking the cost of the solution of the coarse eigenvalue problem into account, the two-level PJD method is about eight times or above faster than the one-level PJD method. The only exception is the problem with dimension 1024×768, which is solved by 768 cores. This problem is too small to be solved by a large number of cores. Notice that the PJD on the coarse grid is not quite scalable and even slower using more processors. One resolution is to solve the coarse problem redundantly using a subset of processors.
3. The two-level PJD algorithm achieved 43.8% parallel efficiency up to $np = 768$, slightly worse than the one-level algorithm. The breakdown analysis of each key component in the two-level PJD algorithm (**Table 8**) is useful for further improving the parallel performance of the two-level PJD algorithm.

	Two-level PJD						One-level PJD			
np	1st coarse	2nd coarse	JD.Its	Time (s)	Sp	Ef (%)	JD.Its	Time (s)	Sp	Ef(%)
Fine grid: 1024×768										
12	0.28	0.04	3	31.8			30	267.8		
48	0.35	0.04	3	6.2	5.2	129.1%	30	52.4	5.1	127.9%
192	0.35	0.08	3	1.1	30.0	187.4%	30	12.8	21.0	131.3%
768	0.48	0.30	3	2.2	14.5	22.7%	41	7.0	38.0	59.4%
Fine grid: 2048×1536										
12	1.22	0.08	4	141.1			63	1991.0		
48	0.13	0.04	5	40.8	3.5	86.5%	66	455.9	4.4	109.2%
192	0.33	0.08	6	10.9	12.9	80.7%	70	118.5	16.8	105.0%
768	0.37	0.26	6	5.0	28.1	43.8%	79	40.9	48.7	76.0%

Table 7: Fixed problem size scalability analysis: one-level PJD and two-level PJD cases. The timing results for finding the first and second coarse eigenvector are also included.

Component	np=12	np=48	np=192	np=768
Form M_i	5.02	1.71	0.32	0.12
GEVP solve	0.03	0.05	0.03	0.03
CorrEq solve	131.39	37.19	10.18	4.76
Compute u, p, r	1.92	0.72	0.19	0.06
Orthogonalization	1.24	0.65	0.09	0.03
Form $\mathcal{A}(\lambda)$	1.48	0.37	0.12	0.03
Totle time	141.1	40.8	10.9	5.0

Table 8: Timing breakdown of each key component in the two-level PJD algorithm with the coarse grid 32×24. The components include (a) Form M_i : M_i calculation; (b) GEVP solve: general eigenvalue problem solve; (c) CorrEq solve: correction equation solve; (d) Compute u, p, r : the vectors u,p,and r calculation; (e) Orthogonalization; (f) Form $\mathcal{A}(\lambda)$: $A(\lambda)$ calculation.

5. Conclusions

In this work, we developed a two-level Schwarz based polynomial Jacobi-Davidson algorithm for PDE polynomial eigenvalue problems and used a cubic acoustic eigenvalue problem as a numerical

example to investigate its performances regarding efficiency and parallel scalability on a cluster of computers. The proposed two-level PJD algorithm consisted of two important ingredients, including the use of the coarse information for constructing a better initial basis for the search space and a low-cost two-level restricted additive Schwarz preconditioner in conjunction with GMRES for the correction equation. After some careful, intense numerical experiments, we identify the optimal or nearly optimal parameters involved in the Krylov-Schwarz method for solving the correction equation. As a result, we achieve an excellent fixed-problem-size-scalability, up to 756 processors. In general, the two-level PJD method is ten times faster than the one-level PJD method proposed by [12].Finally, we close the paper by pointing out a few related project along this research direction. (a) It is interesting to compare the two-grid PJD algorithm with some other popular alternative approaches for solving cubic eigenvalue problems, e.g., Shift-and-invert linearization approach with the Krylov-Schur method. (b) Current studies only considered a quite simple model, where the motion of the structure was ignored, but the effect of the structure acting on the fluids was taken into accounts by considering the Robin-type boundary condition. To model more realistic situation, we need to consider a complicated acoustic-structural eigenvalue problem, and an efficient preconditioner for the problem is necessary to develop.

Acknowledgements

The work was supported in part by the Ministry of Science and Technology of Taiwan, MOST100-2115-M-008-008-MY2. The authors thank Prof. W. Wang and Prof. T.-M. Huang for useful discussion. We are grateful to the National Center for High-performance Computing in Taiwan for computer time and facilities.

References

[1] P. Arbenz, M. Bečka, R. Geus, U. Hetmaniuk, and T. Mengotti. On a parallel multilevel preconditioned Maxwell eigensolver. *Parallel Comput.*, 32:157–165, 2006.

[2] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, . C. McInnes, B. F. Smith, and H. Zhang. PETSc webpage, 2016. http://www.mcs.anl.gov/petsc.

[3] A.T. Barker and X.C. Cai. Two-level Newton and hybrid Schwarz preconditioners for fluid-structure interaction. *SIAM J. Sci. Comput.*, 32:2395–2417, 2010.

[4] A. Bermúdez, R. G. Durán, and J. Rodríguez, R. Solomin. Finite element analysis of a quadratic eigenvalue problem arising in dissipative acoustics. *SIAM J. Numer. Anal.*, 38:267–291, 2000.

[5] A. Bermúdez and R. Rodríguez. Modelling and numerical solution of elastoacoustic vibrations with interface damping. *Int. J. Numer. Meth. Engng.*, 46:1763–1779, 1999.

[6] X.-C. Cai and M. Sarkis. A restricted additive Schwarz preconditioner for general sparse linear systems. *SIAM J. Sci. Comput*, 21:792–797, 1999.

[7] Y.-T. Feng. An integrated Davidson and multigrid solution approach for very large scale symmetric eigenvalue problems. *Comput. Methods in Appl. Mech. Engrg.*, 190:3543–3563, 2001.

[8] M.B. Van Gijzen. The parallel computation of the smallest eigenpair of an acoustic problem with damping. *Int. J. Numer. Meth. Engng.*, 45:765–777, 1999.

[9] V. Hernandez, J.E. Roman, A. Tomas, and V. Vidal. SLEPc webpage, 2016. http://www.grycap.upv.es/slepc.

[10] V. Heuveline and C. Bertsch. On multigrid methods for the eigenvalue computation of nonselfadjoint elliptic operators. *East-West J. Numer. Math.*, 8:275–298, 2000.

[11] M. Hochbruck and D. Löchel. A multilevel Jacobi-Davidson method for polynomial PDE eigenvalue problems arising in plasma physics. *SIAM J. Sci. Comput.*, 32:3151–3169, 2010.

[12] T.-M. Huang, F.-N. Hwang, S.-H. Lai, W. Wang, and Z.-H. Wei. A parallel polynomial Jacobi-Davidson approach for dissipative acoustic eigenvalue problems. *Comput. Fluids*, 45:207–214, 2011.

[13] F.-N. Hwang, Z.-H. Wei, T.-M. Huang, and W. Wang. A parallel additive Schwarz preconditioned Jacobi-Davidson algorithm for polynomial eigenvalue problems in quantum dot simulation. *J. Comput. Phys.*, 229:2932–2947, 2010.
[14] T.-M. Hwang, W.-W. Lin, J.-L. Liu, and W. Wang. Jacobi-Davidson methods for cubic eigenvalue problems. *Numer. Linear Algebra Appl.*, 12:605–624, 2005.
[15] T.-M. Hwang, W.-W. Lin, W.-C. Wang, and W. Wang. Numerical simulation of three dimensional pyramid quantum dot. *J. Comput. Phys.*, 196:208–232, 2004.
[16] F Ihlenburg. *Finite Element Analysis of Acoustic Scattering*. Springer, 1998.
[17] L.E. Kinsler, A.R. Frey, A.B. Coppens, and Sanders J.V. *Fundamentals of Acoustics*. John Wiley & Sons, 2000.
[18] P.T. Lin, J.N. Shadid, M. Sala, R.S. Tuminaro, G.L. Hennigan, and R.J. Hoekstra. Performance of a parallel algebraic multilevel preconditioner for stabilized finite element semiconductor device modeling. *J. Comput. Phys.*, 228:6250–6267, 2009.
[19] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations*. Oxford University Press, 1999.
[20] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.
[21] S. Scacchi. A hybrid multilevel Schwarz method for the bidomain model. *Comput. Methods Appl. Mech. Engrg.*, 197:4051–4061, 2008.
[22] G.L.G. Sleijpen and H.A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 17:401–425, 1996.
[23] G.L.G. Sleijpen and H.A. van der Vorst. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM Rev.*, 42:267–293, 2000.
[24] G.L.G. Sleijpen and F.W. Wubs. Exploiting multilevel preconditioning techniques in eigenvalue computations. *SIAM J. Sci. Comput.*, 25:1249–1272, 2003.
[25] B.F. Smith, P.E. Bjørstad, and W. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge University Press, 1996.
[26] G. Strang and G.J. Fix. *An Analysis of the Finite Element Method*. Prentice-Hall, 1973.
[27] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43:235–286, 2001.
[28] A. Toselli and O.B. Widlund. *Domain Decomposition Methods-Algorithms and Theory*. Springer, 2005.
[29] Y. Wu and X.C. Cai. A parallel two-level method for simulating blood flows in branching arteries with the resistive boundary condition. *Comput. Fluids*, 45:92–102, 2011.
[30] T. Zhao, F-.N. Hwang, and X.-C Cai. Parallel two-level domain decomposition based Jacobi-Davidson algorithms for pyramidal quantum dot simulation. *Comput. Phy. Comm.*, 204:74–81, 2016.