Procedia Computer Science
Volume 29, 2014, Pages 565–575
ICCS 2014. 14th International Conference on Computational Science

Experiments on a Parallel Nonlinear Jacobi–Davidson Algorithm

Yoichi Matsuo[1]*, Hua Guo[2], and Peter Arbenz[3]

[1] Keio University, School of Fundamental Science and Technology, Graduate School of Science and Technology, 3-14-1 Hiyoshi, Kohoku, Yokohama, Kanagawa, 223-8522, Japan
matsuo@math.keio.ac.jp
[2] Systransis AG, Riedstrasse 1, 6343 Risch, Switzerland
[3] ETH Zurich, Computer Science Department, Universitätstr. 6, 8092 Zurich, Switzerland
arbenz@inf.ethz.ch

Abstract
The Jacobi–Davidson (JD) algorithm is very well suited for the computation of a few eigenpairs of large sparse complex symmetric nonlinear eigenvalue problems. The performance of JD crucially depends on the treatment of the so-called correction equation, in particular the preconditioner, and the initial vector. Depending on the choice of the spectral shift and the accuracy of the solution, the convergence of JD can vary from linear to cubic. We investigate parallel preconditioners for the Krylov space method used to solve the correction equation.

We apply our nonlinear Jacobi–Davidson (NLJD) method to quadratic eigenvalue problems that originate from the time-harmonic Maxwell equation for the modeling and simulation of resonating electromagnetic structures.

Keywords:

1 Introduction

In this paper, we consider solving constrained complex-valued nonlinear eigenvalue problems,

$$T(\lambda)\boldsymbol{x} = A\boldsymbol{x} + \lambda R\boldsymbol{x} - \lambda^2 M\boldsymbol{x} = \mathbf{0}, \qquad C^T\boldsymbol{x} = \mathbf{0}, \tag{1}$$

where A, R and M are large sparse complex symmetric matrices. Such eigenvalue problems occur in the analysis of resonant cavities that includes loss mechanisms in the model, of dielectric resonator antennas (DRA), or of plasmonic nanostructures, considering the dispersive dielectric properties of metals in the optical region of the electromagnetic spectrum [5, 9, 10, 22]. These models can be described by the time-harmonic Maxwell equations in 3-dimensional space [10],

$$\begin{aligned} \nabla \times \left(\mu_r^{-1}\nabla \times E(\boldsymbol{x})\right) + i\lambda\sigma Z_0 E(\boldsymbol{x}) - \lambda^2\varepsilon_r E(\boldsymbol{x}) = 0, \\ \nabla \times (\varepsilon_r E(\boldsymbol{x})) = 0, \boldsymbol{x} \in \Omega, \end{aligned} \tag{2}$$

*Most of the work was done during this author's visit to the Computer Science Department at ETH Zurich

doi:10.1016/j.procs.2014.05.051

where $\Omega \subset \mathbb{R}^3$ is a bounded domain, λ is the complex wavenumber in free space, ε_r and μ_r are magnetic permittivity and electric permeability, respectively. Then, a 3-dimensional finite element discretization of equation (2) by Nédélec tetrahedral elements yields complex-valued large sparse nonlinear eigenvalue problems of the form (1), cf. [11].

The Jacobi–Davidson (JD) algorithm suggested by Sleijpen and van der Vorst [16] is very well suited for the computation of a few eigenpairs of such problems. There are numerous variants available for linear and nonlinear eigenvalue problems [1–4, 7, 13, 20, 21]. In [2, 3], JD algorithms for large scale generalized real- and complex-valued symmetric eigenvalue problems $A\boldsymbol{x} = \lambda B\boldsymbol{x}$ have been considered.

The JD algorithm is related to the Newton as well as to the Rayleigh quotient iteration [13, 17]. Since the correction equation is in general solved only approximately these methods are 'inexact'. However, they are 'accelerated' by employing a search space instead of a single vector iterate. The treatment of the so-called correction equation is crucial for the success of JD, in particular, the shift strategy. Mostly, the correction equation is solved by a Krylov space method. Then, the choice of the preconditioner is important.

Betcke and Voss [4, 20] proposed a JD algorithm for solving nonlinear eigenproblems. We closely follow their algorithm but try to improve its behavior for our set of problems investigating mainly shifting strategies and preconditioners. In section 2, the NLJD algorithm is presented. In section 3, we discuss how the choice of the shift in the correction equation and the initial vector affect the speed of convergence. We show examples that explain the properties of the NLJD algorithm. In section 4 we present some realistic electromagnetics simulations. We draw our conclusions in section 5.

2 The nonlinear Jacobi–Davidson algorithm

The Jacobi–Davidson algorithm was suggested by Sleijpen and van der Vorst [16] for the computation of a few eigenvalues close to a given target τ. Shortly after this seminal paper, variants of JD for all kinds of linear and polynomial eigenvalue problems were derived [7, 21]. In this section, we discuss the NLJD algorithm suggested by Voss *et al.* [4, 20] for solving nonlinear eigenproblems.

Assume that $(\vartheta_k, \boldsymbol{y}_k)$ is the eigenpair of the *projected eigenvalue problem* $V_{k-1}^H T(\lambda) V_{k-1}\boldsymbol{y} = \boldsymbol{0}$ closest to the target τ. Then, $(\vartheta_k, \boldsymbol{u}_k)$ with $\boldsymbol{u}_k = V_{k-1}\boldsymbol{y}_k$ is a Ritz pair for the eigenvalue problem (1). The subspace V_{k-1} is expanded by the solution of the so-called correction equation for the NLJD algorithm,

$$\tilde{T}(\sigma_k)\,\boldsymbol{t} = \left(I - \frac{\boldsymbol{p}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{p}_k}\right) T(\sigma_k) \left(I - \frac{\boldsymbol{u}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{u}_k}\right)\boldsymbol{t} = -\boldsymbol{r}_k, \qquad \boldsymbol{t} \perp \boldsymbol{u}_k. \tag{3}$$

The solution $\boldsymbol{t}$ is orthogonalized against V_{k-1} and, after normalization, appended to V_{k-1} such that $V_k = [V_{k-1}, \boldsymbol{t}]$. This procedure is repeated until the residual $\|T(\vartheta_k)\boldsymbol{u}_k\|_2$ is small enough.

The NLJD algorithm has mostly been applied to nonlinear eigenproblems of the form (1). We developed our own parallel NLJD solver in Femaxx with only a few modifications from the algorithms of Voss [20]. Details of our NLJD method are given in Algorithm 1. The difference between this solver and Voss's [4, 20] are the projection steps 3 and 18, where we enforce the vectors to satisfy the divergence-free condition $C^T\boldsymbol{x} = \boldsymbol{0}$. The actual projection is

$$(I - YH^{-1}C^T)\,\boldsymbol{t} = \boldsymbol{t} - Y\boldsymbol{z}, \qquad H\boldsymbol{z} = C^T\boldsymbol{t}, \tag{4}$$

where $H\boldsymbol{z} = C^T\boldsymbol{t}$ is solved to high accuracy by Jacobi preconditioned Bi-CGstab. In step 7, the projected eigenvalue problem is solved by the method of successive linear problems [15] or

Algorithm 1: The nonlinear Jacobi–Davidson algorithm

```
   Data: Matrices $T, Y, H$, and $C$, target $\tau$, initial vector $\boldsymbol{v_0}$, tolerance $\varepsilon$
   Result: An eigenpair $(\lambda, \boldsymbol{x})$ closest to the target $\tau$
1  begin
2      Determine a preconditioner $M \approx T(\tau)$;
3      Project $\boldsymbol{v}_0 = \left(I - YH^{-1}C^T\right)\boldsymbol{v}_0$;
4      Set initial search space $V_0$=[$\boldsymbol{v}_0$];
5      while not converged do
6          $k = k + 1$;
7          compute a eigenpair $(\vartheta_k, \boldsymbol{x_k})$ of the projected eigenproblem $V^H T(\vartheta_{k-1}) V\boldsymbol{y} = \boldsymbol{0}$,
           closest to the target $\tau$.;
8          Ritz vector $\boldsymbol{u}_k = V\boldsymbol{x}_k$;
9          Residual $\boldsymbol{r}_k = T(\vartheta)\boldsymbol{u}_k$;
10         if $\|\boldsymbol{r}_k\| < \varepsilon$ then
11             accept the approximate eigenpair $(\vartheta_k, \boldsymbol{x_k})$;
12         end
13         $\boldsymbol{p}_k = T'(\vartheta_k)\boldsymbol{u}$;
14         Reduce the search space $V_k$ if necessary;
15         Choose $\sigma_k = \tau$ or $\vartheta_k$;
16         Solve the correction equation to find the vector expanding the search space by a
           Krylov space method with preconditioner $M$;
17         $\left(I - \frac{\boldsymbol{p}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{p}_k}\right) T(\sigma_k) \left(I - \frac{\boldsymbol{u}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{u}_k}\right)\boldsymbol{t}_k = -\boldsymbol{r}_k$;
18         Project $\boldsymbol{t}_k = \left(I - YH^{-1}C^T\right)\boldsymbol{t}_k$;
19         Orthogonalize $\boldsymbol{t}_k$ against the current search space $V_k$;
20         Expand the search space $V_{k+1} = [V_k, \boldsymbol{t}_k]$;
21     end
22 end
```

the QZ algorithm after linearization. In steps 16 and 17, the correction equation is solved by the Bi-CGstab method with preconditioner $\tilde{M}$. With $M \approx T(\sigma_k) \approx T(\tau)$ we set

$$\tilde{M} := \left(I - \frac{\boldsymbol{p}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{p}_k}\right) M \left(I - \frac{\boldsymbol{u}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{u}_k}\right). \tag{5}$$

Notice that both $\tilde{T}(\sigma_k)$ and $\tilde{M}$ are maps from span$\{\boldsymbol{u}_k\}^\perp$ to span$\{\boldsymbol{p}_k\}^\perp$. Therefore it makes sense to precondition (3) by $\tilde{M}$,

$$\tilde{M}^+\tilde{T}(\sigma_k)\,\boldsymbol{t} = -\tilde{M}^+\boldsymbol{r}, \qquad \boldsymbol{t}_k \perp \boldsymbol{u}_k. \tag{6}$$

We start the Bi-CGstab iteration for solving (5) with the initial guess $\boldsymbol{0}$ which trivially satisfies the constraint. In each step, we compute $\boldsymbol{z} = \tilde{M}^+\tilde{T}\boldsymbol{w}$. Since $\left(I - \frac{\boldsymbol{u}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{u}_k}\right)\boldsymbol{w} = \boldsymbol{0}$, we get

$$\tilde{\boldsymbol{w}} = \tilde{T}\boldsymbol{w} = \left(I - \frac{\boldsymbol{p}_k\boldsymbol{u}_k^H}{\boldsymbol{u}_k^H\boldsymbol{p}_k}\right) T(\sigma_k)\boldsymbol{w} = T(\sigma_k)\boldsymbol{w} - \frac{\boldsymbol{u}_k^H T(\sigma_k)\boldsymbol{w}}{\boldsymbol{u}_k^H\boldsymbol{p}_k}\boldsymbol{p}_k. \tag{7}$$

Then we solve $\tilde{M}\boldsymbol{z} = \tilde{\boldsymbol{w}}$, $\boldsymbol{z} \perp \boldsymbol{u}_k$. Again, since $\boldsymbol{z} \perp \boldsymbol{u}_k$, we get

$$\boldsymbol{z} = M^{-1}\tilde{\boldsymbol{w}} - \frac{\boldsymbol{u}_k^H M^{-1}\tilde{\boldsymbol{w}}}{\boldsymbol{u}_k^H M^{-1}\boldsymbol{p}_k} M^{-1}\boldsymbol{p}_k. \tag{8}$$

We can solve equation (5) without computing the pseudoinverse of $\tilde{M}$ by means of equations (7)–(8). In the first few NLJD iterations the approximation far from the desired eigenpair. Therefore, it is not necessary to solve the correction equation accurately. We stop the iteration after a small number of steps or if the variable stopping criterion by Fokkema *et al.* [7],

$$\|\tilde{\boldsymbol{r}}_i\|_2 < 1.2^{-k}\|\tilde{\boldsymbol{r}}_i\|_2, \tag{9}$$

is satisfied.

3 Solving the correction equation

Let us for a moment assume that $T(\sigma_k)$ in the correction equation (3) is symmetric and that the correction equation can be solved exactly by Gaussian elimination. If $\sigma_k = \vartheta_k$ then

$$\boldsymbol{t}_k = \alpha_k T(\vartheta_k)^{-1}\boldsymbol{p}_k - T(\vartheta_k)^{-1}\boldsymbol{r}_k, \qquad \alpha_k = \frac{\boldsymbol{u}_k^H T(\vartheta_k)^{-1}\boldsymbol{r}_k}{\boldsymbol{u}_k^H T(\vartheta_k)^{-1}\boldsymbol{p}_k}, \tag{10}$$

where α_k is determined by the constraint $\boldsymbol{t}_k^H\boldsymbol{u}_k = 0$. Equation (10) is mathematically equivalent with a step of the cubically convergent Rayleigh quotient iteration [13]. The Rayleigh quotient iteration works very well if $\boldsymbol{u}_k$ has a large component in the direction of the sought eigenvector. Otherwise, there is slow or even no convergence. In other words, if $\boldsymbol{u}$ is far from the desired eigenvector, the search space is poorly expanded and the iteration stagnates.

To stabilize the iteration we keep the shift fixed in the early phase of the NLJD iteration, $\sigma_k = \tau$. This makes (10) a shift-and-invert iteration that converges stably but slowly, i.e., linearly to the eigenvector associated with the eigenvalue closest to the target τ. For the performance of NLJD it is important to get a suitable approximation of the desired eigenpair and switch from the fixed to variable shifts at the right moment. In the next section we experiment with variants on how to do this, cf. [8].

By means of three small examples we illustrate how the choice of the shift in the correction equation affects the convergence of the NLJD algorithm. These examples are inspired by [6,16] and turned into quadratic eigenproblems. Note that we do not impose constraints on the solutions. We investigate several variants of the NLJD algorithm: (1) we keep the shift fixed throughout the iteration, $\sigma_k = \tau$; (2) we vary the shift $\sigma_k = \vartheta_k$ in all steps; (3) we keep the shift fixed initially and start varying it at step 5. All examples have been executed with MATLAB 7.10.0. The tolerance ε is set to 10^{-8} for Examples 1, 2, and 3, and to 10^{-6} for Example 4, respectively. The initial vector is a random vector except Example 4. In each example, each procedure finds the same eigenpair.

Example 1. Here, $T \in \mathbb{R}^{1000\times1000}$ is a symmetric sparse matrix. The matrix A has diagonal elements $a_{jj} = j$. The elements on sub- and superdiagonal are all equal to 0.5. The matrix R is tridiagonal with elements $(-1, 2, -1)$. The matrix M has nonzero diagonal elements $m_{j,j} = 1$ and two further nonzeros $m_{1000,1} = m_{1,1000} = 0.5$. We set the target $\tau = 40$ and obtain the eigenvalue 38.32. The closest eigenvalue is 34.99. We employ the NLJD algorithm with variable shift (NLJD-1), with fixed shift (NLJD-2), and with switching the shift at step 5 (NLJD-3). The correction equation is solved exactly in each step. Figure 1 gathers the convergence histories.

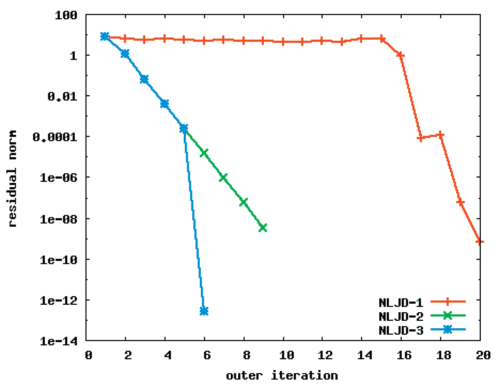

Figure 1: Convergence history for Example 1.

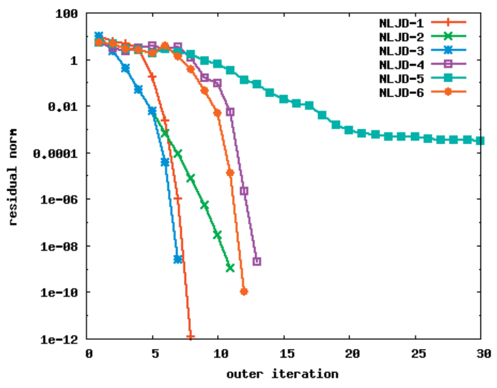

Figure 2: Convergence history for Examples 2 and 3.

The NLJD algorithm with the variable shift converges cubically. However, since the initial vector is far from the desired eigenvector, NLJD-1 needs many iteration steps to get into the region of fast converge. NLJD-2 converges linearly. The initial steps of NLJD-3 coincide with those of NLJD-2. After 5 steps the shift is adapted, $\sigma_k = \vartheta_k$. From this moment we observe cubic convergence.

Example 2. Now, $T \in \mathbb{C}^{1000\times 1000}$. The matrices A and R are tridiagonal. The nonzero elements in the jth row of A are $(0.5 + 0.5\imath, j - 0.5\imath, 0.5 + 0.5\imath)$, those in the jth row of R are $(-1, 2 + j\imath, -1)$. The matrix M has nonzeros on the diagonal $m_{j,j} = 1 + \imath$ and two more nonzeros $m_{1000,1} = m_{1,1000} = 0.5+0.5\imath$. We set $\tau = 0$. All variants of NLJD find the eigenvalue $-0.126 + 0.341\imath$. The nearest eigenvalue is $-0.319 + 0.478\imath$, They behave very similarly as in Example 1, see Figure 2. However, NLJD-1 gets in the region of cubic convergence much quicker. Therefore, NLJD-2 is clearly the slowest solver, in terms of iteration count.

Example 3. Since it is not practical for large scale problems to solve the correction equation exactly we now experiment with an iterative solver. We solve the correction equation for the problem of Example 2 by GMRES(25) with ILU(0) preconditioner. We build the preconditioner M from $T(\tau)$. We stop GMRES if either $\|\tilde{\boldsymbol{r}}_i\|_2 < 1.2^{-k}\|\tilde{\boldsymbol{r}}_0\|_2$ or after 50 iteration steps. So, we restart at most once. We denote the NLJD algorithm with variable shift by “NLJD-4”, with fixed shift by “NLJD-5”, and with switching shift by “NLJD-6”. Again, we switch from fixed to variable shift at step 5. The convergence history is given in Figure 2. We can see similarities between Example 2 and Example 3. The speed of convergence of NLJD-2 and NLJD-5 is linear. However, NLJD-5 is so slow that it does not converge within 30 outer iteration steps. If we set the tolerance for the GMRES to be smaller, for instance $\|\tilde{\boldsymbol{r}}_i\|_2 < 2.0^{-k}\|\tilde{\boldsymbol{r}}_0\|_2$, the speed increases. On the other hand, we can still see quadratic convergence for NLJD-6 and also NLJD-4 since the initial guess is close enough to the solution.

Example 4. We also show how a good initial vector can promote the convergence. According to Voss [20], if no information on the desired eigenvector is available, it may be helpful to execute a few Arnoldi steps for the linear eigenproblem

$$T(\tau)\boldsymbol{y} = \vartheta\boldsymbol{y}, \tag{11}$$

and choose the eigenvector corresponding to the smallest eigenvalue as the initial vector. In these experiments, we executed 5 steps to get an initial vector. We solved again the problem of

Example 2 and employed NLJD-4, NLJD-5, and NLJD-6 with the improved initial vector. The convergence history is illustrated in Figure 3. NLJD-4 converges cubically to the eigenvalue in just 3 steps. NLJD-6 also converges cubically after switching the target. The speed of convergence of NLJD-5 is improved, but it is still not competitive.

These results show that if we solve the correction equation by an iterative solver accurately enough, we can still see superlinear convergence. Although these problems are artificial they show quite nicely the convergence behavior of Jacobi–Davidson. We can say that switching the shift from fixed to variable can improve the speed of convergence considerably. We however did not investigate when to actually change the strategy. An initial phase with a fixed shift is certainly beneficial for a stable convergence.

4 Experiments with realistic problems

In this section, we experiment with problems that originate in electromagnetics simulations. The computations are executed with Femaxx, a software package that has continuously been developed since 2002, in a collaboration between ETH Zurich and the Paul Scherrer Institute (PSI) [1, 2, 8–10]. The time-harmonic Maxwell equations (2) are solved in geometrically complicated domains Ω with curved boundaries. The problems are discretized by the finite element method employing linear and quadratic Nédélec finite elements [11], which are mandatory in order to properly represent the electromagnetic field vector. The large sparse eigenvalue problems are solved by the Jacobi–Davidson QZ algorithm [18] or the NLJD algorithm of Algorithm 1. All solvers are parallelized for distributed memory parallel architectures by means of the Trilinos framework which uses the MPI for communication [19].

In the previous version of Femaxx, only the NLJD algorithm with fixed shift and diagonal (Jacobi) preconditioner are implemented [9, 10]. We implemented new preconditioners and incorporated variable shift strategies into NLJD. The problems that we discuss now have been executed on 32 cores of the Merlin4 cluster at PSI. Merlin4 has 30 compute nodes (with a total of 360 cores). The nodes have dual Intel Xeon X5670 2.93 GHz CPUs with 24 or 48 GB RAM. We show numerical experiments with fixed shift (indicated by "f"), and with variable shift (indicated by "s"). "blkj-" refers to the Block-Jacobi preconditioner. Here, a block is the local diagonal portion of the respective matrix. In the tables, "nit_{out}" and "nit_{in}" denote the number of outer and inner iteration steps, respectively. "t_{prec}" denotes the time for building the preconditioner. In all experiments we compute a single eigenpair.

4.1 Ohmically lossy material

We first consider the half-filled rectangular resonator [9, 22], where one half of the complete volume is filled with ohmically lossy material and the other half is vacuated. It is modeled by the Maxwell equations (2). We apply perfect electric conductor (PEC) boundary conditions which implies that the tangential electric field vanishes identically on the cavity wall. We approximate the problem by quadratic Nédélec elements with 1'852'810 degrees of freedom and 76'854'188 non-zero elements. The finite element discretization yields a constrained complex-valued symmetric quadratic eigenproblem of the form (1). The matrix A is real-valued, the matrix M is complex-valued, and R is a purely imaginary matrix. C is a complex-valued rectangular matrix. Setting $\tau = 100$ we find the eigenvalue $119.704 + 108.913\imath$. The nearest eigenvalue is $137.883 + 80.981\imath$. As indicated in Algorithm 1, the computed eigenpairs $(\lambda, \boldsymbol{x})$ satisfy the divergence-free condition, i.e., $C^T\boldsymbol{x} = \mathbf{0}$. We use the M-orthogonal projector P

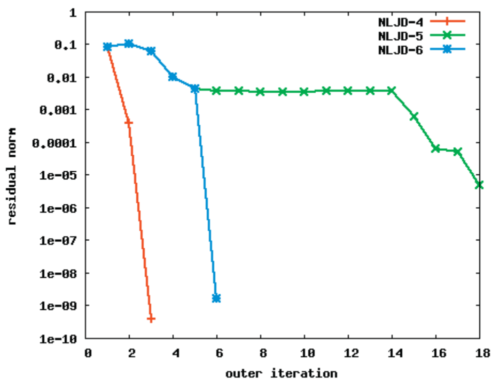

Figure 3: Convergence history for Example 4.

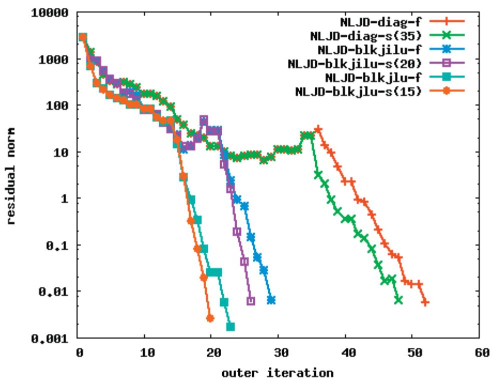

Figure 4: Convergence history of the half-filled rectangular resonator

Table 1: Statistics for the half-filled rectangular resonator

Prec	nit_{out}	nit_{in}	t	t_{prec}	Prec	nit_{out}	nit_{in}	t	t_{prec}
diag-f	52	1306	538.67	0.07	diag-s(35)	48	757	534.13	0.08
blkjilu-f	29	366	409.00	7.3	blkjilu-s(20)	27	394	427.17	7.66
blkjlu-f	23	160	453.44	46.58	blkjlu-s(15)	20	244	582.85	41.71

in (4) to enforce the constraint [8]. This implies that all vectors in the search space satisfy the divergence-free condition.

We solve the half-filled rectangular resonator by the NLJD with fixed or switching shift. The correction equation is solved by the Bi-CGstab method with Jacobi (diagonal) and Block-Jacobi preconditioners. With Block-Jacobi preconditioners, each local block is solved by either the ILU(0) or the LU factorization. In Femaxx we use hierarchical bases for the Nédélec and the Lagrange basis functions. Therefore it is natural to use hierarchical basis preconditioners for solving the correction equation [1]. We employ the NLJD and restart after every 10 iteration steps. The tolerance ε for the residual norm is set to 10^{-2}. In Bi-CGstab we execute at most 50 iterations for the correction equation. For the Jacobi preconditioner, we switch to variable shifts after 35 outer iteration. For the Block-Jacobi ILU and LU preconditioners, the switch is after 20 and 15 steps, respectively.

The numerical results are presented in Table 1. The convergence histories are found in Figure 4. Considering these results, the switching shift strategy reduces the number of outer iterations. In this example it is much more difficult to find a good approximation of the desired eigenpair than in the previous examples. Although we can reduce the number of outer iteration steps by switching shifts, the gain is little. Since the preconditioner is recomputed in every iteration step, the single step becomes much more expensive. This is in particular visible with the LU factorizations (blkj-lu). Nevertheless, if the reduction of the iteration steps is big enough, then also the overall execution time is reduced. Clearly, it is very important to switch from fixed to variable shift at the right moment. The blkj-ilu approach is a good compromise as the (re-)computation of the preconditioner is not excessively expensive.

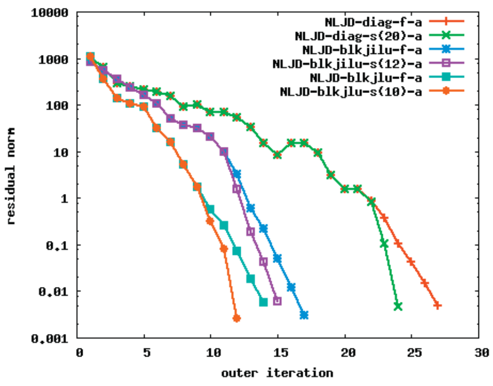

Figure 5: Convergence history for the half-filled rectangular resonator.

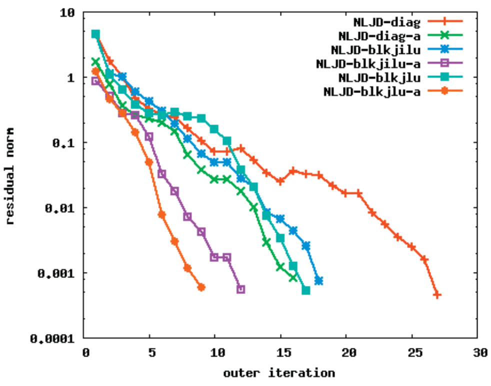

Figure 6: Convergence history for DRA problem.

Experiments with an improved initial vector

Now we discuss numerical experiments with an improved initial vector. Voss [20] suggested to generate a 'good' initial vector by executing a few steps of the Arnoldi algorithm for the linear eigenvalue problem

$$T(\tau)\boldsymbol{y} = \vartheta \boldsymbol{y}, \tag{12}$$

and select the eigenvector corresponding to the smallest eigenvalue as the initial vector for the nonlinear problem. In our experiments, we executed 5 Arnoldi steps to get an initial vector. Execution times in the tables include the time for constructing the initial vector. Constructing an initial vector is referred to by "a".

We solve again the half-filled rectangular resonator [9, 22] with quadratic elements with 1'852'810 degrees of freedom and 76'854'188 non-zero elements by the Bi-CGstab algorithm with Jacobi and Block-Jacobi preconditioners. In this experiment the shift is fixed for all solvers. Except for the initial vector, every parameter is the same as with the half-filled rectangular resonator. The results are found in Table 2. Here we can see improvements. Comparing with the results with fixed shift in Table 1, the NLJD with diagonal preconditioning the number of outer iterations is reduced by about 40%, the overall time by about 10%. The NLJD with Block-Jacobi preconditioning reduces the number of the outer iteration by about 35% and the overall time by about 5%. Since we need time to construct a initial vector, the reduction of the overall time is less. Nevertheless, by executing the Arnoldi steps, the initial vector can get significant components in the direction of the desired eigenvector and convergence to an unwanted eigenvector is less probable.

Then we executed the NLJD with switching shifts and Arnoldi type initial vector. The results are listed on the right side of Table 2. Comparing with the left side of Table 1, the number of outer iteration steps is further reduced. However, again, the increase of the number of inner iteration steps per outer iteration step counteracts this improvement. Figure 5 shows the convergence histories of NLJD with fixed and switching shift, and improved initial vector. The speed of convergence of the NLJD with switching shift and good initial vector is improved.

Table 2: Statistics for the half-filled rectangular resonator with a good initial vector

Prec	nit_{out}	nit_{in}	t	t_{prec}	Prec	nit_{out}	nit_{in}	t	t_{prec}
diag-f-a	27	597	500.85	0.07	diag-s(20)-a	24	677	540.13	0.08
blkjilu-f-a	17	340	398.43	8.53	blkjilu-s(12)-a	15	396	487.11	7.8
blkj-lu-f-a	14	150	419.62	43.63	blkjlu-s(10)-a	12	230	626.12	42.56

Table 3: Statistics of the DRA with a random (left) or improved (right) initial vector. The shift is fixed.

Prec	nit_{out}	nit_{in}	t	t_{prec}	Prec	nit_{out}	nit_{in}	t	t_{prec}
diag-f	27	474	130.27	1.18	diag-a	16	258	80.45	1.15
blkilu-f	16	316	71.46	1.92	blkilu-a	9	270	56.94	2.02
blkjlu-f	17	223	85.36	2.92	blkjlu-a	11	179	60.73	3.12

4.2 Dielectric resonant antenna with a good initial vector

Now we consider the problem of rectangular dielectric resonator antennas (DRAs) in the microwave region. Here we solve the time-harmonic Maxwell equations (2) with first order absorbing boundary conditions (ABC) [11]

$$\mathbf{n} \times \nabla \times \mathbf{E}(\mathbf{x}) = -ik\,\mathbf{n}\times(\mathbf{n} \times \mathbf{E}(\mathbf{x})),$$

where $k = k_0\sqrt{\mu_r\varepsilon_r}$ is the wavenumber of the surrounding medium. DRAs have been analyzed theoretically [12,14]. The complex-valued matrices A, R, and M are composed of real symmetric element matrices multiplied by a (varying) complex factor. With quadratic Nédélec elements there are 439'188 degrees of freedom and 1'895'460 non-zero elements. With the target $\tau = 0.1$ we find the eigenvalue $0.117 + 0.00134\imath$. The closest eigenvalue is $0.162 + 0.0392\imath$. Again, we execute 5 Arnoldi steps to get an improved initial vector. The numerical results are presented in Table 3. The convergence histories are found in Figure 6. By using a good initial vector, the NLJD with diagonal preconditioning reduces about 40% the number of outer and inner iteration steps, as well as the overall time. The NLJD with Block-Jacobi preconditioning also exhibit considerable improvements with respect to the number of outer and inner iterations, and overall time.

5 Conclusions

In this paper, we discussed how the shift in the correction equation and the initial vector affect the speed of convergence of the NLJD. Clearly, switching the shift reduces the number of outer iteration steps and improves the speed of convergence. However, the recomputation of the preconditioner consumes a lot of computing time. Therefore, choosing the right instance when to switch from the fixed to the variable shift is critical. In our experiments, the switching helped to improve the execution times of the solver, however by at most 20%.

On the other hand, improving the initial vector by solving an auxiliary linear eigenvalue problem was very successful. For instance in the DRA problem, choosing a good initial vector reduced the overall solution time by 40%.

Compared with plain diagonal preconditioning, Block-Jacobi preconditioning improves the speed of convergence. It reduces the number of inner and outer iterations leading to substantial reductions of execution time – in spite of the fact that constructing these preconditioners needs more time.

We plan to apply the techniques developed in this paper to improve solution times to the simulation of further resonating electromagnetic structures such as plasmonic nanostructures. In particular we will investigate more sophisticated techniques for obtaining good starting vectors.

References

[1] P. Arbenz, M. Bečka, R. Geus, U. Hetmaniuk, and T. Mengotti. On a parallel multilevel preconditioned Maxwell eigensolver. *Parallel Comput.*, 32(2):157–165, 2006.

[2] P. Arbenz and R. Geus. Multilevel preconditioned iterative eigensolvers for Maxwell eigenvalue problems. *Appl. Numer. Math.*, 54(2):107–121, 2005.

[3] P. Arbenz and M. E. Hochstenbach. A Jacobi–Davidson method for solving complex-symmetric eigenvalue problems. *SIAM J. Sci. Comput.*, 25(5):1655–1673, 2004.

[4] T. Betcke and H. Voss. A Jacobi–Davidson-type projection method for nonlinear eigenvalue problems. *Future Gener. Comput. Syst.*, 20(3):363–372, 2004.

[5] S.J. Cooke and B. Levush. Eigenmodes of microwave cavities containing high-loss dielectric materials. In *17th Particle Accelerator Conference*, pages 2431–2433, May 1997.

[6] M. Crouzeix, B. Philippe, and M. Sadkane. The Davidson method. *SIAM J. Sci. Comput.*, 15:62–76, 1994.

[7] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst. Jacobi–Davidson style QR and QZ algorithms for the partial reduction of matrix pencils. *SIAM J. Sci. Comput.*, 20(1):94–125, 1998.

[8] R. Geus. *The Jacobi–Davidson algorithm for solving large sparse symmetric eigenvalue problems.* PhD thesis no. 14734, ETH Zurich, 2002.

[9] H. Guo. *3-dimensional eigenmodal analysis of electromagnetic structures.* PhD thesis, ETH Zurich, 2013.

[10] H. Guo, B. Oswald, and P. Arbenz. 3-dimensional eigenmodal analysis of plasmonic nanostructures. *Opt. Express*, 20(5):5481–5500, 2012.

[11] J. Jin. *The Finite Element Method in Electromagnetics.* John Wiley, New York, NY, 2nd edition, 2002.

[12] R. K. Mongia and A. Ittipiboon. Theoretical and experimental investigations on rectangular dielectric resonator antennas. *IEEE Trans. Antennas Propag.*, 45(9):1348–1356, 1997.

[13] Y. Notay. Combination of Jacobi–Davidson and conjugate gradients for the partial symmetric eigenproblem. *Numer. Linear Algebra Appl.*, 9:21–44, 2002.

[14] A. Okaya and L. F. Barash. The dielectric microwave resonator. *Proc. IRE*, 50:2081–2092, 1962.

[15] A. Ruhe. Algorithms for the nonlinear eigenvalue problem. *SIAM J. Numer. Anal.*, 10(4):674–689, 1973.

[16] 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(2):401–425, 1996.

[17] G. L. G. Sleijpen and H. A. van der Vorst. The Jacobi–Davidson method for eigenvalue problems and its relation with accelerated inexact Newton scheme. In S. D. Margenov and P. S. Vassilevski, editors, *Second IMACS International Symposium on Iterative Methods in Linear Algebra*, pages 377–389, New Brunswick, NJ, 1996. IMACS.

[18] G. L. G. Sleijpen and H. A. van der Vorst. Jacobi–Davidson method. In Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, pages 238–246. SIAM, Philadelphia, PA, 2000.

[19] The Trilinos Project Home Page. `http://trilinos.sandia.gov/`.

[20] H. Voss. A Jacobi–Davidson method for nonlinear and nonsymmetric eigenproblems. *Comput. Struct.*, 85(17-18):1284–1292, 2007.

[21] J. Demmel Z. Bai, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, PA, 2000.

[22] Y. Zhu and A. C. Cangellaris. Robust finite-element solution of lossy and unbounded electromagnetic eigenvalue problems. *IEEE Trans. Microwave Theory Tech.*, 50(10):2331–2338, 2002.