AN ITERATIVE FINITE ELEMENT METHOD FOR ELLIPTIC EIGENVALUE PROBLEMS

P. SOLIN* AND S. GIANI†

Abstract. We consider the task of resolving accurately the nth eigenpair of a generalized eigenproblem rooted in some elliptic partial differential equation (PDE), using an adaptive finite element method (FEM). Conventional adaptive FEM algorithms call a generalized eigensolver after each mesh refinement step. This is not practical in our situation since the generalized eigensolver needs to calculate n eigenpairs after each mesh refinement step, it can switch the order of eigenpairs, and for repeated eigenvalues it can return an arbitrary linear combination of eigenfunctions from the corresponding eigenspace. In order to circumvent these problems, we propose a novel adaptive algorithm that only calls the eigensolver once at the beginning of the computation, and then employs an iterative method to pursue a selected eigenvalue-eigenfunction pair on a sequence of locally refined meshes. Both Picard's and Newton's variants of the iterative method are presented. The underlying partial differential equation (PDE) is discretized with higher-order finite elements (hp-FEM) but the algorithm also works for standard low-order FEM. The method is described and accompanied with theoretical analysis and numerical examples. Instructions on how to reproduce the results are provided.

Key words. Partial differential equation, Eigenvalue problem, Iterative method, Adaptive higher-order finite element method, hp-FEM, Reproducible research

AMS subject classifications. 65N25, 65N30, 65N50, 35B45

1. Introduction.

Eigenvalue problems for partial differential equations (PDE) are of considerable theoretical and practical interest in engineering and sciences. To name a few applications, let us mention automated multilevel substructuring methods for noise prediction in acoustics [14], analysis of photonic crystals [10, 13] and plasmonic guides [5], and stability analysis of fluid systems [6].

The most common approach to solving eigenproblems is using eigensolvers. For larger problems it is practical to employ iterative eigensolvers such as ARPACK [15]. A characteristic common to all eigensolvers is that even if the user is interested in one particular eigenpair only, several additional eigenvalues and possibly eigenvectors need to be computed. These auxiliary eigenpairs are the byproduct of techniques such as deflation or orthogonalization that are used to filter out unwanted solutions.

The majority of eigensolvers are not specifically designed to work with adaptive finite element methods (FEM), and their application on sequences of locally refined meshes can lead to substantial problems. In this paper we illustrate some of these problems and propose a novel iterative method that alleviates them. We are going to adapt the Picard's and Newton's methods to solve eigenvalue problems in order to minimize the number of unwanted eigenpair exploiting the orthogonality between eigenvectors. In contrast to conventional adaptive methods that call an eigensolver after each mesh refinement, the present method is capable of following reliably a selected eigenpair on a sequence of adapted meshes. This is particularly useful with multiple eigenvalues when only a particular eigenfunction in the eigenspace is wanted.

1.1. Outline of the Paper.

Section 2 illustrates some difficulties associated with conventional adaptive FEM algorithms. Section 3 introduces notations and

*Department of Mathematics and Statistics, University of Nevada, Reno, USA, and Institute of Thermomechanics, Prague, Czech Republic (solin@unr.edu).

†School of Mathematical Sciences, University of Nottingham, United Kingdom (stefano.giani@nottingham.ac.uk).

preliminaries. Section 4 presents a simple Picard's method and shows that it does not work as expected. The problem is fixed in Section 5 by adding orthogonalization. Section 6 presents a Newton's method with orthogonalization. Section 7 presents an outline of the adaptivity algorithm. Section 8 discusses a reconstruction technology for spectra perturbed by discretization. This is used to devise an algorithm for computing reference solutions for *hp*-adaptivity in Section 9. Moreover, in Section 10 we present improved versions of our iterative methods. A priori convergence results are presented in Section 11. Numerical results are presented in Section 12. Reproducibility of results is discussed in Section 13. Conclusion and outlook are presented in Section 14.

2. Motivation. We will illustrate a typical outcome of repeated eigensolver calls on a simple eigenproblem of the form

$$-\Delta u = \lambda u, \quad u = 0 \text{ on } \partial\Omega \tag{2.1}$$

that is solved in a square domain with a hole, i.e., $\Omega = (0,\, 1)^2 \setminus [1/3,\, 2/3]^2$.

Equation (2.1) is discretized via the finite element method on a mesh consisting of 16 quadratic ($p = 2$) triangular elements shown in Fig. 2.1.

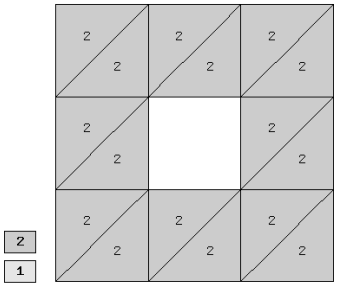

FIG. 2.1. *Initial mesh for automatic adaptivity.*

In this case the second and the third eigenvalues are repeated, i.e., $\lambda_2 = \lambda_3$. The corresponding approximate eigenfunctions are shown in Fig. 2.2.

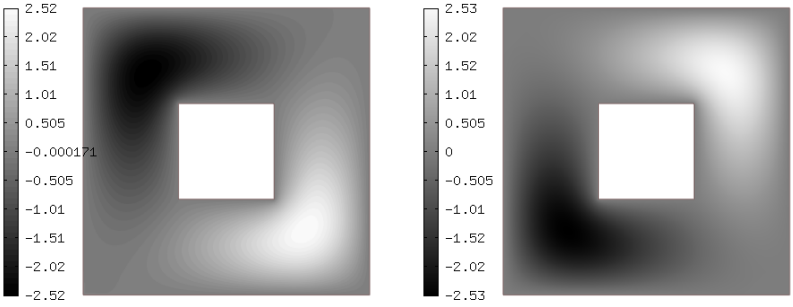

FIG. 2.2. *Approximate eigenfunctions for λ_2, λ_3 on initial mesh.*

Let us assume that the objective of the computation is to resolve accurately the eigenfunction corresponding to λ_3 (target eigenfunction). This eigenfunction contains singularities at the upper-right and lower-left corners of the hole and it is very smooth

(almost constant) in the vicinity of the remaining two corners. Accordingly, in the first mesh adaptation step, the mesh is refined towards the upper-right and lower-left corners of the hole.

On the refined mesh, a generalized eigensolver is called again. The new approximate eigenfunctions for λ_2, λ_3 are shown in Fig. 2.3.

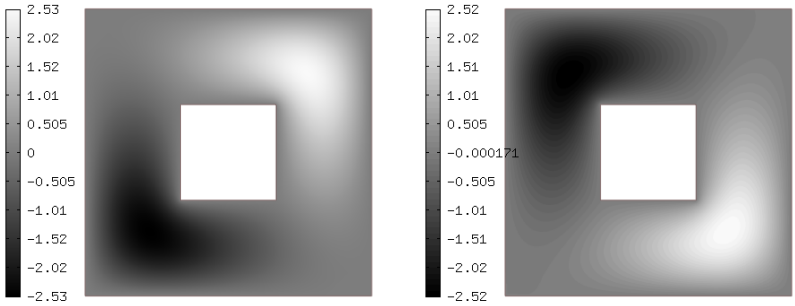

FIG. 2.3. *Approximate eigenfunctions for λ_2, λ_3 after the second call to the generalized eigensolver (compare to Fig. 2.2).*

The reader can observe that the second call to the generalized eigensolver switched the order of the eigenfunctions. This means that the second round of mesh refinements goes towards the upper-leftt and lower-right corners of the hole, i.e., into regions where the target eigenfunction is flat, and no refinements are done where it has singularities. Obviously, this is inefficient. In a worst-case scenario, the eigenfunctions are not switched back, and thus all following mesh refinements go into regions where the target eigenfunction is flat, i.e., its singularities are never resolved.

To our best knowledge, contemporary generalized eigensolvers do not offer any systematic way to avoid this problem. There are only a few publications that deal specifically with repeated eigenvalues, such as [8, 11]. Their authors adapt the mesh considering an entire basis of the eigenspace of the repeated eigenvalue. This means that for a double eigenvalue such as, e.g., $\lambda_2 = \lambda_3$ the mesh would be refined using both the eigenfunctions u_2 and u_3, even if one is just interested in the eigenpair (λ_3, u_3). This, however, does not solve the problem illustrated in Figs. 2.2 and 2.3. In addition, the degrees of freedom that are targeting the eigenpair (λ_2, u_2) are not going to improve significantly the accuracy of the eigenpair (λ_3, u_3) and in fact they are wasted.

3. Preliminaries. Throughout, $L^2(\Omega)$ denotes the usual space of square-integrable real valued functions equipped with the standard norm

$$\|f\|_0 \;:=\; \left(\int_\Omega |f|^2\right)^{\frac{1}{2}}. \tag{3.1}$$

The symbol $H^1(\Omega)$ denotes the usual space of functions in $L^2(\Omega)$ with square-integrable weak first partial derivatives. The H^1-norm is denoted by $\|f\|_1$.

The variational formulation of problem (2.1) is: *Find eigenpairs of the form* $(\lambda_j, u_j) \in \mathbb{R} \times H^1_0(\Omega)$ *such that*

$$\left.\begin{array}{rcll} a(u_j, v) & = & \lambda_j\, b(u_j, v), \quad \text{for all} \quad v \in H^1_0(\Omega) \\ \|u_j\|_0 & = & 1 \end{array}\right\} \tag{3.2}$$

where

$$a(u,v) := \int_\Omega \nabla u(x) \cdot \nabla v(x), \tag{3.3}$$

and

$$b(u,v) := \int_\Omega u(x)v(x). \tag{3.4}$$

Now, to discretize (3.2), let $\mathcal{T}_n, n = 1, 2, \ldots$ denote a family of meshes on Ω. The meshes can be irregular with multiple levels of hanging nodes, and they can combine possibly curvilinear triangular and quadrilateral elements. These meshes may be obtained using automatic adaptivity.

By $h_{n,\tau}$ we denote the diameter of element τ, we define $h_n := \max_{\tau\in\mathcal{T}_n}\{h_{n,\tau}\}$. Similarly with $p_{n,\tau}$ we denote the order of polynomials of element τ, we define $p_n := \min_{\tau\in\mathcal{T}_n}\{p_{n,\tau}\}$. On any mesh $\mathcal{T}_n$ we denote by $V_n \subset H_0^1(\Omega)$ the finite dimensional space of continuous functions v such that on any element τ we have that $v|_\tau \in \mathcal{P}_{p_{n,\tau}}(K)$. Here either $\mathcal{P}_{p_{n,\tau}}(K)$ is the space of polynomials of total degree at most $p_{n,\tau}$ if τ is a triangular element, or $\mathcal{P}_{p_{n,\tau}}(K)$ is the space of polynomials of degree at most $p_{n,\tau}$ in each variable if τ is a quadrilateral element.

The discrete version of (3.2) reads: *Find eigenpairs of the form* $(\lambda_{j,n}, u_{j,n}) \in \mathbb{R} \times V_n$ *such that*

$$\left.\begin{array}{rcll} a(u_{jn}, v_n) & = & \lambda_{j,n}\, b(u_{j,n}, v_n), \quad \text{for all} \quad v_n \in V_n \\ \|u_{j,n}\|_0 & = & 1 \end{array}\right\} \tag{3.5}$$

4. Picard's Method. Problem (3.5) can be reformulated in matrix form as: *Find eigenpairs of the form* $(\lambda, \mathbf{u}) \in \mathbb{R} \times \mathbb{R}^N$*, where* N *is the dimension of* V_n*, such that*

$$\left.\begin{array}{rcl} \mathbf{Au} & = & \lambda\mathbf{Bu}\,, \\ \mathbf{u}^t\mathbf{Bu} & = & 1 \end{array}\right\} \tag{4.1}$$

where the entries of the matrices $\mathbf{A}$ and $\mathbf{B}$ are

$$\mathbf{A}_{k,p} := a(\phi_k, \phi_p)\,, \quad \mathbf{B}_{k,p} := b(\phi_k, \phi_p)\,,$$

where ϕ_i are the basis functions spanning V_n.

The Picard's method, see Algorithm 1, takes as arguments the matrices $\mathbf{A}$ and $\mathbf{B}$, an initial guess $\tilde{u}$ for the eigenfunction, a relative tolerance Tol and an absolute tolerance AbsTol. The algorithm returns an approximated eigenpair (λ_n, u_n). Because we use this iterative method on a sequence of adaptively refined meshes, we normally set as initial guess the projection on the refined mesh of the eigenfunction of interest $u_{j,n-1}$.

Algorithm 1 Picard's method

$(\lambda_{j,n}, u_{j,n}) := \text{Picard}(\mathbf{A}, \mathbf{B}, \tilde{u}, \text{Tol}, \text{AbsTol})$
$\mathbf{u}^1 := \tilde{u}$
$\lambda^1 := \dfrac{u_{j,n}^t \mathbf{A} u_{j,n}}{u_{j,n}^t \mathbf{B} u_{j,n}}$
$m = 1$
repeat
  $\mathbf{u}^{m+1} := \mathbf{A}^{-1}\lambda^m \mathbf{B}\mathbf{u}^m$
  $\lambda^{m+1} := \dfrac{(\mathbf{u}^{m+1})^t \mathbf{A}\mathbf{u}^{m+1}}{(\mathbf{u}^{m+1})^t \mathbf{B}\mathbf{u}^{m+1}}$
  $m := m + 1$
until $\frac{\|\mathbf{u}^{m+1} - \mathbf{u}^m\|_1}{\|\mathbf{u}^m\|_1} > \text{Tol}$ **and** $|\lambda^{m+1} - \lambda^m| > \text{AbsTol}$
$u_n := \mathbf{u}^m$
$\lambda_n := \lambda^m$

The next theorem shows that the Picard's method always converges to the smallest eigenvalue.

THEOREM 4.1. *The Picard's method in exact arithmetic converges into the eigenspace which is not orthogonal to the initial guess* $\mathbf{u}^1$ *and whose eigenvalue has minimum module.*

Proof. Any vector $\mathbf{u}^m$ can be expressed as

$$\mathbf{u}^m = \sum_{i=1}^{N} c_i^m \mathbf{u}_i \ ,$$

where c_i^m are real coefficients, N is the size of the matrices $\mathbf{A}$ and $\mathbf{B}$ and the vectors $\mathbf{u}_i \equiv u_{i,n}$ are the eigenvectors of the discrete problem, which form an orthonormal basis. With no lost in generality we can assume that λ_1 is the eigenvalue of minimum module and that c_1^1 is different from 0.

In the case that λ_1 is simple we have from the definition of the problem:

$$\mathbf{u}^{m+1} = \mathbf{A}^{-1}\lambda^m \mathbf{B}\mathbf{u}^m = \Big(\Pi_{j=1}^m \lambda^j\Big)\Big(\mathbf{A}^{-1}\mathbf{B}\Big)^m \mathbf{u}^1 = \Big(\Pi_{j=1}^m \lambda^j\Big) \sum_{i=1}^{N} c_i^1 (\lambda_i)^{-m} \mathbf{u}_i \ ,$$

where λ_i are the eigenvalues corresponding to $\mathbf{u}_i$. Then

$$\mathbf{u}^{m+1} = \Big(\Pi_{j=1}^m \lambda^j\Big)(\lambda_1)^{-m}\Big(c_1^1 \mathbf{u}_1 + \sum_{i=2}^{N} c_i^1 \frac{(\lambda_1)^m}{(\lambda_i)^m}\mathbf{u}_i\Big) \ ,$$

where it is clear that, since $\lambda_1/\lambda_i < 1$, for $i \geq 2$, the direction of $\mathbf{u}^{m+1}$ tends toward the direction of $\mathbf{u}_1$. Furthermore, the Rayleigh quotient of $\mathbf{u}^{m+1}$

$$\lambda^{m+1} := \frac{(\mathbf{u}^{m+1})^t \mathbf{A}\mathbf{u}^{m+1}}{(\mathbf{u}^{m+1})^t \mathbf{B}\mathbf{u}^{m+1}} = \lambda_1 \frac{(c_1^1)^2 + \sum_{i=2}^{N} (c_i^1)^2 \left(\frac{\lambda_1}{\lambda_i}\right)^{m-1}}{(c_1^1)^2 + \sum_{i=2}^{N} (c_i^1)^2 \left(\frac{\lambda_1}{\lambda_i}\right)^{m}} \ ,$$

converges to λ_1.

In the case that λ_1 has multiplicity R and that c_r^1, for some $1 \leq r \leq R$, is not zero, we similarly have that for all $i > R$:

$$\mathbf{u}^{m+1} = \left(\Pi_{j=1}^{m}\lambda^j\right)(\lambda_1)^{-m}\left(\sum_{r=1}^{R} c_r^1 \mathbf{u}_r + \sum_{i=R+1}^{N} c_i^1 \frac{(\lambda_1)^m}{(\lambda_i)^m}\mathbf{u}_i\right) ,$$

and then

$$\lambda^{m+1} := \frac{(\mathbf{u}^{m+1})^t \mathbf{A}\mathbf{u}^{m+1}}{(\mathbf{u}^{m+1})^t \mathbf{B}\mathbf{u}^{m+1}} = \lambda_1 \frac{\displaystyle\sum_{r=1}^{R}(c_r^1)^2 + \sum_{i=2}^{N}(c_i^1)^2\left(\frac{\lambda_1}{\lambda_i}\right)^{m-1}}{\displaystyle\sum_{r=1}^{R}(c_r^1)^2 + \sum_{i=2}^{N}(c_i^1)^2\left(\frac{\lambda_1}{\lambda_i}\right)^{m}} ,$$

which converges again to λ_1.
□

Theorem 4.1 shows that even if the initial guess $\mathbf{u}^1$ is very close to a certain discrete eigenfunction $u_{i,n}$, for some i, the method can always converge to a different eigenfunction or a linear combinations of eigenfunctions with corresponding eigenvalues smaller in module than $\lambda_{i,n}$. In real arithmetic, even if the initial guess $\mathbf{u}^1$ is orthogonal to all eigenfunctions of indexes less than i, for some $m > 1$ the orthogonality could be perturbed, due to round-off errors, and the method can eventually converges anyway to a different eigenfunction or a linear combinations of eigenfunctions with corresponding eigenvalues smaller in module than $\lambda_{i,n}$.

To illustrate this behavior, we refer to the numerical simulations in Section 12.1, where we use the fifth eigenfunction corresponding to $\lambda_5 = 10$ (Fig. 4.1 left) as a starting guess for the Picard's method. However, the Picard's method converges to the first eigenfunction corresponding to $\lambda_1 = 2$ (Fig. 4.1 right).

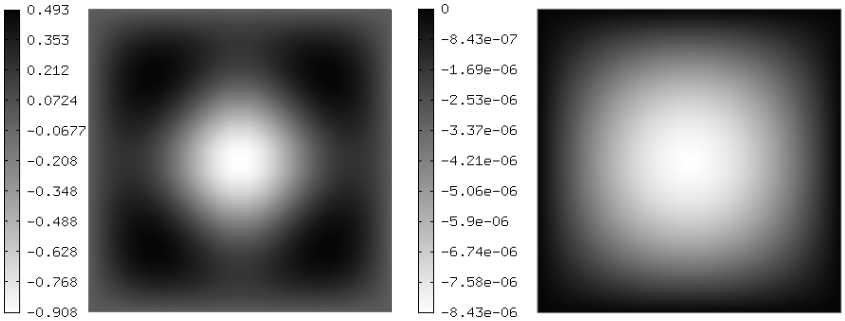

FIG. 4.1. *The Picard's method converges from an initial guess that is very close to the fifth eigenfunction (left) that corresponds to $\lambda_5 = 10$ to the first eigenfunction (right) that corresponds to $\lambda_1 = 2$.*

5. Picard's Method with Orthogonalization. In order to make the Picard's method suitable to approximate efficiently any discrete eigenpair, and not only the first one, we derived Algorithm 2, which has an orthogonalization procedure in it.

The Picard's method with orthogonalization takes as arguments the matrices $\mathbf{A}$ and $\mathbf{B}$ of (3.5), an initial guess $\tilde{u}$ for the eigenfunction, the tolerances AbsTol andTol and it also takes the $j-1$ eigenfunctions $u_{1,n}, \ldots, u_{j-1,n}$. Then it returns the eigenpair $\lambda_{j,n}, u_{j,n}$ on the refined mesh.

Algorithm 2 Picard's method with orthogonalization

$(\lambda_{j,n}, u_{j,n}) := \text{PicardOrtho}(\mathbf{A}, \mathbf{B}, \tilde{u}_{j,n-1}, \text{Tol}, \text{AbsTol}, u_{1,n}, \ldots, u_{j-1,n})$
$\mathbf{u}^1 := \tilde{u}_{j,n-1}$
$\lambda^1 := \dfrac{u_{j,n}^t \mathbf{A} u_{j,n}}{u_{j,n}^t \mathbf{B} u_{j,n}}$
$m = 1$
repeat
  $\mathbf{u}^{m+1} := \mathbf{A}^{-1} \lambda^m \mathbf{B} \mathbf{u}^m$
  for $i = 1$ to $j - 1$ **do**
    $\mathbf{u}^{m+1} := \mathbf{u}^{m+1} - (u_{i,n}^t \mathbf{B} \mathbf{u}^{m+1}) u_{i,n}$ {Orthogonalization}
  end for
  $\mathbf{u}^{m+1} = \dfrac{\mathbf{u}^{m+1}}{((\mathbf{u}^{m+1})^t \mathbf{B} \mathbf{u}^{m+1})^{1/2}}$ {Normalize}
  $\lambda^{m+1} := \dfrac{(\mathbf{u}^{m+1})^t \mathbf{A} \mathbf{u}^{m+1}}{(\mathbf{u}^{m+1})^t \mathbf{B} \mathbf{u}^{m+1}}$
  $m := m + 1$
until $\frac{\|\mathbf{u}^{m+1} - \mathbf{u}^m\|_1}{\|\mathbf{u}^m\|_1} > \text{Tol}$ **and** $|\lambda^{m+1} - \lambda^m| > \text{AbsTol}$
$u_{j,n} := \mathbf{u}^m$
$\lambda_{j,n} := \lambda^m$

As can be seen in Algorithm 2, the orthogonalization is done in each iteration. This is necessary in real arithmetic to guarantee that $\mathbf{u}^m$ is orthogonal to all eigenfunctions $u_{1,n}, \ldots, u_{j-1,n}$, for all m. Otherwise in exact arithmetic it would be enough to orthogonalize only $\mathbf{u}^1$. Moreover, a normalization step is necessary in all iterations because due to the orthogonalization procedure, this version of the Picard's method does not conserve the norm of the vectors and possible underflows or overflows could happen with no normalization.

THEOREM 5.1. *Algorithm 2 never converges to an eigenvalue of index smaller than j.*

Proof. The proof comes straightforwardly from the arguments used to prove Theorem 4.1. The fact that $\mathbf{u}^m$ is orthogonal to all eigenfunctions $\mathbf{u}_1, \ldots, \mathbf{u}_{j-1}$, implies that the coefficients c_i^m, with $m = 1, \ldots, j-1$, are zeros. Then, the Rayleigh quotient converges to λ_j by the same arguments use before. □

Remark: To make the Picard's method usable in practice, it is recommended to enhance it with Anderson acceleration [1]. This method combines a number of last iterates in a GMRES-like fashion. The result is equivalent to a Jacobian-free quasi-Newton (Broyden) method.

6. Newton's Method with Orthogonalization. The second iterative method that we are going to propose is based on the Newton's method applied to eigenvalue problems. Denoting by $\tilde{x} := (x, \lambda)$, we have that problem (3.2) can be rewritten in the form

$$0 = f(\tilde{x}) := \begin{pmatrix} Ax & - & \lambda_j \, Bx \\ x^T Bx & - & 1 \end{pmatrix},$$

then denoting by $\tilde{h} := (h, \delta)$ the increment, we have that the truncated Taylor series of the problem is

$$f(\tilde{x} + \tilde{h}) \approx f(\tilde{x}) + J_f(\tilde{x}) \cdot \tilde{h}, \tag{6.1}$$

where the Jacobian matrix is defined as

$$J_f(\tilde{x}) := \begin{pmatrix} A - B\lambda & -Bx \\ 2Bx^T & 0 \end{pmatrix}.$$

Then when $\tilde{x} + \tilde{h}$ is a solution of (3.2), we have from (6.1) that

$$J_f(\tilde{x}) \cdot \tilde{h} = -f(\tilde{x}),$$

which defines the linear problem of the Newton's method that we are solving.

Algorithm 3 Newton's method

$(\lambda_{j,n}, u_{j,n}) :=$ Newton$(\mathbf{A}, \mathbf{B}, \tilde{u}, \text{Tol}, \text{AbsTol})$
$\mathbf{u}^1 := \tilde{u}$
$\lambda^1 := \dfrac{u_{j,n}^t \mathbf{A} u_{j,n}}{u_{j,n}^t \mathbf{B} u_{j,n}}$
$m = 1$
repeat
  Solve $J_f(\mathbf{u}^m, \lambda^m) \cdot \tilde{h} = -f(\mathbf{u}^m, \lambda^m)$
  $\mathbf{u}^{m+1} := \mathbf{u}^m + h$
  $\lambda^{m+1} := \lambda^m + \delta$
  $m := m + 1$
until $\frac{\|\mathbf{u}^{m+1} - \mathbf{u}^m\|_1}{\|\mathbf{u}^m\|_1} > \text{Tol}$ **and** $|\lambda^{m+1} - \lambda^m| > \text{AbsTol}$
$u_n := \mathbf{u}^m$
$\lambda_n := \lambda^m$

In order to make the method suitable for all eigenpairs, we are going write a version of the Newton's method that uses an orthogonalization procedure, similarly to what we have already done for the Picard's method.

Algorithm 4 Newton's method with orthogonalization

$(\lambda_{j,n}, u_{j,n}) := \text{NewtonOrtho}(\mathbf{A}, \mathbf{B}, \tilde{u}_{j,n-1}, \text{Tol}, \text{AbsTol}, u_{1,n}, \ldots, u_{j-1,n})$
$\mathbf{u}^1 := \tilde{u}_{j,n-1}$
$\lambda^1 := \dfrac{u_{j,n}^t \mathbf{A} u_{j,n}}{u_{j,n}^t \mathbf{B} u_{j,n}}$
$m = 1$
repeat
  Solve $J_f(\mathbf{u}^m, \lambda^m) \cdot h = -f(\mathbf{u}^m, \lambda^m)$
  $\mathbf{u}^{m+1} := \mathbf{u}^m + h$
  $\lambda^{m+1} := \lambda^m + \delta$
  $m := m + 1$
  for $i = 1$ to $j - 1$ **do**
    $\mathbf{u}^{m+1} := \mathbf{u}^{m+1} - (u_{i,n}^t \mathbf{B} \mathbf{u}^{m+1}) u_{i,n}$ {Orthogonalization}
  end for
  $\mathbf{u}^{m+1} := \dfrac{\mathbf{u}^{m+1}}{((\mathbf{u}^{m+1})^t \mathbf{B} \mathbf{u}^{m+1})^{1/2}}$ {Normalize}
  $\lambda^{m+1} := \dfrac{(\mathbf{u}^{m+1})^t \mathbf{A} \mathbf{u}^{m+1}}{(\mathbf{u}^{m+1})^t \mathbf{B} \mathbf{u}^{m+1}}$
  $m := m + 1$
until $\frac{\|\mathbf{u}^{m+1} - \mathbf{u}^m\|_1}{\|\mathbf{u}^m\|_1} > \text{Tol}$ **and** $|\lambda^{m+1} - \lambda^m| > \text{AbsTol}$
$u_{j,n} := \mathbf{u}^m$
$\lambda_{j,n} := \lambda^m$

THEOREM 6.1. *Algorithm 4 converges always to an eigenvalue greater or equal to* λ_j.

Proof. This result is a direct consequence of the orthogonalization step in Algorithm 4. We are using again the fact that any vector $\mathbf{u}^{m+1}$ can be expressed as

$$\mathbf{u}^{m+1} = \sum_{i=1}^{N} c_i^{m+1} \mathbf{u}_i,$$

where c_i^{m+1} are real coefficients, N is the size of the matrices $\mathbf{A}$ and $\mathbf{B}$ and the vectors $\mathbf{u}_i \equiv u_{i,n}$ are the eigenvectors of the discrete problem, which are sorted accordingly the magnitude of the corresponding eigenvalues λ_i. In particular, when $\mathbf{u}^{m+1} := \mathbf{u}^m + h$, where h is the solution of $J_f(\mathbf{u}^m, \lambda^m) \cdot h = -f(\mathbf{u}^m, \lambda^m)$, we have that, after the application of the orthogonalization step, the resulting vector is

$$\hat{\mathbf{u}}^{m+1} = \sum_{i=j}^{N} c_i^{m+1} \mathbf{u}_i.$$

Then, it is straightforward to see that the Rayleigh quotient

$$\lambda^{m+1} := \frac{(\hat{\mathbf{u}}^{m+1})^t \mathbf{A} \hat{\mathbf{u}}^{m+1}}{(\hat{\mathbf{u}}^{m+1})^t \mathbf{B} \hat{\mathbf{u}}^{m+1}} \geq \lambda_j.$$

□

7. Automatic *hp*-Adaptivity. With the Picard's and Newton's methods in hand, we can now proceed to automatic *hp*-adaptivity. This part of the paper is not new but we need to present it to make the paper self-contained. We use an

algorithm from [20] that is an analogy to embedded higher-order ODE methods: In each adaptivity step it constructs an approximation pair with different orders of accuracy and uses their difference as an a-posteriori error estimator.

Algorithm 5 Automatic *hp*-Adaptivity

Let $\mathcal{T}_0^c$ be an initial coarse mesh. We construct an initial fine mesh $\mathcal{T}_0^f$ by refining all elements in space and moreover increasing their polynomial degrees by one. A generalized eigensolver is called one time only, to obtain a solution pair (λ_0^c, u_0^c) on the initial coarse mesh $\mathcal{T}_0^c$.
Set $k := 0$
repeat
Project the approximation u_k^c to the mesh $\mathcal{T}_k^f$. The projection is denoted by $P_k^f u_k^c$. Since the finite element spaces on meshes $\mathcal{T}_k^c$ and $\mathcal{T}_k^f$ are embedded, there is no projection error.
Calculate an initial guess $\tilde{\lambda}_k^f$ for the eigenvalue on the mesh $\mathcal{T}_k^f$ using the relation

$$\tilde{\lambda}_k^f = \frac{(P_k^f u_k^c)^T \mathbf{A}_k^f P_k^f u_k^c}{(P_k^f u_k^c)^T \mathbf{B}_k^f P_k^f u_k^c},$$

where $\mathbf{A}_k^f$ and $\mathbf{B}_k^f$ are the stiffness and mass matrices on the mesh $\mathcal{T}_k^f$, respectively.
The pair $(\tilde{\lambda}_k^f, P_k^f u_k^c)$ is **not a solution** to the generalized eigenproblem on the mesh $\mathcal{T}_k^f$, but it is used as an initial guess.
Apply the Picard's or Newton's method as described in Sections 5 and 6, to obtain a solution pair (λ_k^f, u_k^f) on the mesh $\mathcal{T}_k^f$.
Project the approximation u_k^f back to the coarse mesh $\mathcal{T}_k^c$ to obtain $P_k^c u_k^f$.
Calculate an a-posteriori error estimate e_k^c,

$$e_k^c = u_k^f - P_k^c u_k^f.$$

Note: e_k^c is a function, not a number.
Use e_k^c to guide one step of automatic *hp*-adaptivity [20] that yields a new coarse mesh $\mathcal{T}_{k+1}^f$.
Update $k := k + 1$
until The H^1-norm of e_{k-1}^c is sufficiently small.

8. Reconstruction Technology. It is well known that the discretization process perturbs the spectrum, in particular the eigenspace $E(\lambda_j)$ of multiple eigenvalue λ_j can be split in more than one discrete eigenspace $E(\lambda_{j,n}), E(\lambda_{j+1,n}), \ldots, E(\lambda_{j+m,n})$ with correspondent discrete eigenvalues $\lambda_{j,n}, \lambda_{j+1,n}, \ldots, \lambda_{j+m,n}$ forming a small cluster for sufficiently rich finite element spaces, also under the same assumption we have that

$$\dim E(\lambda_j) = \sum_{i=0}^{m} \dim E(\lambda_{j+i,n}).$$

This phenomenon is already well documented in literature, see [21, 3, 12].

Different finite element spaces can split the same multiple eigenspace in different ways, this also happens with adaptively refined meshes. It is not rare that the same

multiple eigenspace is split differently on the coarse and on the refined meshes. A different split corresponds to different discrete eigenfunctions, then it is not always possible to find for the same eigenvalue on the refined mesh an eigenfunction similar to the one on the coarse mesh.

We propose a way to always construct on a refined mesh, an approximation of the same eigenfunction as on the coarse mesh. The idea is based on the fact that for a sufficiently rich finite element space, the space $M_n(\lambda_j) = \text{span}\{E(\lambda_{j,n}), E(\lambda_{j+1,n}), \ldots, E(\lambda_{j+m,n})\}$ is an approximation of the space $E(\lambda_j)$, see [21]. Let us denote the space $M_{n,1}(\lambda_j)$ as the subspace of $M_n(\lambda_j)$ of functions with unit norm in the L^2. So for any function $U_{n-1} \in M_{n-1,1}(\lambda_j)$, we propose the function $U_n \in M_{n,1}(\lambda_j)$ that minimize the $\|U_{n-1} - U_n\|_{0,\Omega}$ as an approximation of U_{n-1} on the refined mesh. For a sufficiently rich finite element space the minimizer is unique. By construction

$$U_n = \sum_{i=1}^{R} c_i \; u_{i,n}, \tag{8.1}$$

where $u_{1,n}, u_{2,n}, \ldots, u_{R,n}$, with $R = \dim\, E(\lambda_j)$, are eigenfunctions of the discrete problem forming an orthonormal basis for $M_{n,1}$ and where the coefficients c_i satisfy

$$\sum_{i=1}^{R} c_i^2 = 1. \tag{8.2}$$

From the definition of problem (3.2) we have that the reconstructed eigenvalue is defined as

$$\Lambda_n = \frac{a(U_n, U_n)}{b(U_n, U_n)}.$$

The couple (Λ_n, U_n) is not a discrete eigenpair of problem (3.2) in general, in section 11 we prove that (Λ_n, U_n) converges a priori at the same rate as any other discrete eigenpair of (3.2) to a continuous eigenpair.

9. Computing Reference Solution via Reconstruction. In this section we present two algorithms to compute approximations of eigenpairs. Each algorithm is based on a different method to compute the discrete spectrum, but all of them use the reconstruction technology to keep track of the eigenfunction of interest.

In all algorithms we are going to use on the initial mesh an iterative eigensolver with calling interface $\{(\lambda_{j,n}, u_{j,n})_{j=1}^{i}\} := \text{Eigensolver}(\mathbf{A}, \mathbf{B}, i, \text{Tol}, \text{MaxIter})$, that computes the set of discrete eigenpairs $\{(\lambda_{j,n}, u_{j,n})\}_{j=1}^{i}$ and where $\mathbf{A}$ is the stiffness matrix of the problem, $\mathbf{B}$ is the mass matrix of the problem i is the number of eigenpairs to compute, Tol is the requested tolerance for the eigenpairs and MaxIter is the maximum number of iterations.

All algorithm we describe below are based on the reconstruction technology which is guided by two parameters: DTE and FIE. The parameter DTE should be equal to the multiplicity of the continuous eigenvalue λ that the user want to approximate. All the algorithms work also when DTE contains an upper bound of the multiplicity of λ, so in practise the multiplicity of the target eigenvalue is not necessary to be known exactly. The parameter FIE should be equal to the index i of the first discrete eigenvalue on the initial mesh $\lambda_{i,0}$ that approximates λ. The reconstruction technology is described in Algorithm (6).

Algorithm 6 Reconstruction algorithm

$(\Lambda_n, U_n) :=$ Reconstruction$(\{(\lambda_{j,n}, u_{j,n})\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{n-1}, U_{n-1}))$
Compute $(\Lambda_n, U_n) := \displaystyle\sum_{i=\text{FIE}}^{\text{FIE+DTE}} b(u_{i,n}, U_{n-1}) u_{i,n}$
$U_n := \dfrac{U_n}{\sqrt{b(U_n, U_n)}}$ {Normalize}
$\Lambda_n := a(U_n, U_n)$

The first method is based on the Picard's method. The only three parameters not yet defined are M which is the maximum number of mesh adaptation requested, $0 < \text{FIETE} \leq \text{FIE} + \text{DTE}$ which is the index of the eigenvalue that the user want to target and err which is tolerance for the residual.

Algorithm 7 Adaptive method based on the Picard's method

$(\Lambda_M, U_M) :=$ PicardAdapt$(\mathcal{T}_0, V_0, M, \text{err}, \text{Tol}, \text{AbsTol}, \text{MaxIter}, \text{DTE}, \text{FIE}, \text{TE})$
Construct $\mathbf{A}_0$ and $\mathbf{B}_0$
$\{(\lambda_{j,0}, u_{j,0})\}_{j=1}^{\text{DTE+FIE}} :=$ Eigensolver$(\mathbf{A}_0, \mathbf{B}_0, \text{DTE} + \text{FIE},$
Tol, MaxIter)
$(\Lambda_0, U_0) := (\lambda_{\text{TE},0}, u_{\text{TE},0})$
$m := 1$
repeat
  Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
  Construct $\mathbf{A}_m$ and $\mathbf{B}_m$
  $(\lambda_{1,m}, u_{1,m}) :=$ Picard$(\mathbf{A}_m, \mathbf{B}_m, u_{1,m-1}, \text{Tol}, \text{AbsTol})$
  $j = 1$
  for $j = 2$ to DTE + FIE **do**
    $(\lambda_{j,m}, u_{j,m}) :=$
      PicardOrtho$(\mathbf{A}_m, \mathbf{B}_m, u_{j,m-1}, \text{Tol}, \text{AbsTol}, u_{j,m-1}, \ldots, u_{j-1,m-1})$
  end for
  $(\Lambda_m, U_m) :=$ Reconstruction$(\{(\lambda_{j,m}, u_{j,m})\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}))$
  $m := m + 1$
until $m \leq M$ OR $\eta \leq$ err

Similarly we define the adaptive method based on the Newton's method.

Algorithm 8 Adaptive method based on the Newton's method

$(\Lambda_M, U_M) :=$ NewtonAdapt$(\mathcal{T}_0, V_0, M,$ err, Tol, AbsTol, MaxIter, DTE, FIE, TE)
Construct $\mathbf{A}_0$ and $\mathbf{B}_0$
$\{(\lambda_{j,0}, u_{j,0})\}_{j=1}^{\mathrm{DTE+FIE}} :=$ Eigensolver$(\mathbf{A}_0, \mathbf{B}_0,$ DTE + FIE, Tol, MaxIter)
$(\Lambda_0, U_0) := (\lambda_{\mathrm{TE},0}, u_{\mathrm{TE},0})$
$m := 1$
repeat
  Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
  Construct $\mathbf{A}_m$ and $\mathbf{B}_m$
  $(\lambda_{1,m}, u_{1,m}) :=$ Newton$(\mathbf{A}_m, \mathbf{B}_m, u_{1,m-1},$ Tol, AbsTol)
  $j = 1$
  for $j = 2$ to DTE + FIE **do**
    $(\lambda_{j,m}, u_{j,m}) :=$ NewtonOrtho$(\mathbf{A}_m, \mathbf{B}_m, u_{j,m-1},$ Tol, AbsTol, $u_{1,m-1}, \ldots, u_{j-1,m-1})$
  end for
  $(\Lambda_m, U_m) :=$ Reconstruction$(\{(\lambda_{j,m}, u_{j,m})\}_{j=\mathrm{FIE}}^{\mathrm{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}))$
  $m := m + 1$
until $m \leq M$ OR $\eta \leq$ err

10. Iterative methods with improved orthogonalization. The main disadvantage of the algorithms presented in Sections 5 and 6 is the cost of the method. Those methods ensure that the eigenpair with the correct index m is computed, but, in order to ensure that, all eigenpairs of indices from 1 to $n-1$ are also computed. What we present now is a more cheaper way to ensure the computation of the target eigenpair. The key idea is a smarter way to use the orthogonalization only when it is really necessary and this is possible mainly because we can use information from the previous mesh to identify unwanted eigenpairs.

The reason why we introduce the algorithms in Sections 5 and 6 was to cure the downside of the iterative methods to possibly converge to an eigenpair different from the target one. The answer to this problem presented in Sections 5 and 6 was to compute all possible eigenpair to which the method could erroneously converge to, and then use all of them to force the method, by the orthogonalization, to produce an approximation of the wanted eigenpair. There is a better way which consists in starting with no-orthogonalization and then every time that the iterative method produces an unwanted eigenpair, save it to be used next time in the orthogonalization process to prevent the method to converge again to the same unwanted solution. This is possible only if a way to distinguish between wanted and unwanted solution is available. In the adaptive setting this is always possible because the orthogonality of any newly computed eigenpair against the results on the previous mesh can be computed.

The Algorithms 9 and 10 are the incarnations of the improved orthogonality technology applied with the either the Picard's or the Newton's method respectively. Since these two algorithms are identical except for the call to either PicardOrtho or to NewtonOrtho, in the rest we are gong to describe only Algorithm 9.

Algorithm 9 compute a set of eigenpairs $\{(\lambda_{j,n}, u_{j,n})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}}$, approximating the target continuous eigenspace, on the mesh $\mathcal{T}_n$. The arguments that it needs are: the matrices $\mathbf{A}$ and $\mathbf{B}$, the approximation of the target eigenspace $\{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}}$ computed on the previous mesh $\mathcal{T}_{n-1}$, and then projected on the refined mesh $\mathcal{T}_n$,

and a real value $0 < \text{ThO} < 1$ which is used to decide wether a computed eigenfunction is part of the approximation of the target eigenspace or not. The set $\mathcal{D}$ is empty at the beginning, but then it is fed with all computed eigenfunctions. Then $\mathcal{D}$ is passed to every call to PicardOrtho and so it guarantees that the same eigenfunction is never computed twice. The key part of the algorithm is just after the call to PicardOrtho, where the newly computed eigenfunction is analyzed. The analysis consists in checking how orthogonal the newly computed eigenfunction $u_{j,n}$ is respect to the span of $\{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1})\}_{j=\text{FIE}}^{\text{DTE+FIE}}$. If the resulting value is smaller than ThO, then $u_{j,n}$ is not considered part of the target eigenspace and a new approximation of $u_{j,n}$ is done. Otherwise, $u_{j,n}$ is kept and the algorithm pass to approximate the next eigenfunction in the target eigenspace. The algorithm ends when all eigenpair in $\{(\lambda_{j,n}, u_{j,n})\}_{j=\text{FIE}}^{\text{DTE+FIE}}$ are computed.

Algorithm 9 Picard's method with improved orthogonalization

$\{(\lambda_{j,n}, u_{j,n})\}_{j=\text{FIE}}^{\text{DTE+FIE}} := \text{PicardImpOrtho}(\mathbf{A}, \mathbf{B}, \{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1})\}_{j=\text{FIE}}^{\text{DTE+FIE}}, \text{ThO})$
$\mathcal{D} := \emptyset$
$j := \text{DTE} + \text{FIE}$
repeat
  $(\lambda_{j,n}, u_{j,n}) := \text{PicardOrtho}(\mathbf{A}, \mathbf{B}, u_{j,n-1}, \text{Tol}, \text{AbsTol}, \mathcal{D})$
  Add $u_{j,n}$ to $\mathcal{D}$
  inner := 0
  for $i = \text{FIE} \rightarrow \text{DTE} + \text{FIE}$ **do**
    inner := inner + $u_{j,n}^t \mathbf{B} \tilde{u}_{i,n-1}$
  end for
  if inner > ThO **then**
    $j := j - 1$
  end if
until $j < \text{FIE}$

Algorithm 10 Newton's method with improved orthogonalization

$\{(\lambda_{j,n}, u_{j,n})\}_{j=\text{FIE}}^{\text{DTE+FIE}} := \text{NewtonImpOrtho}(\mathbf{A}, \mathbf{B}, \{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1})\}_{j=\text{FIE}}^{\text{DTE+FIE}}, \text{ThO})$
$\mathcal{D} := \emptyset$
$j := \text{DTE} + \text{FIE}$
repeat
  $(\lambda_{j,n}, u_{j,n}) := \text{NewtonOrtho}(\mathbf{A}, \mathbf{B}, u_{j,n-1}, \text{Tol}, \text{AbsTol}, \mathcal{D})$
  Add $u_{j,n}$ to $\mathcal{D}$
  inner := 0
  for $i = \text{FIE} \rightarrow \text{DTE} + \text{FIE}$ **do**
    inner := inner + $u_{j,n}^t \mathbf{B} \tilde{u}_{i,n-1}$
  end for
  if inner > ThO **then**
    $j := j - 1$
  end if
until $j < \text{FIE}$

So the number of computed eigenfunction may vary: in the best case scenario when the method is used to approximate an eigenspace of dimension DTE, only DTE

eigenfunctions are computed. In the worst case scenario, DTE+FIE eigenfunctions are computed, which is the number of computed eigenfunctions by Algorithms 2, 4 on the same space. Because almost never the worst case scenario is achieved, Algorithms 9 and 10 are more efficient than Algorithms 2, 4.

We conclude this section stating the adaptive algorithms with improved orthogonality.

Algorithm 11 Adaptive method based on the Picard's method with improved orthogonality

$(\Lambda_M, U_M) :=$ PicardImpAdapt$(\mathcal{T}_0, V_0, M, \text{err}, \text{Tol}, \text{MaxIter}, \text{DTE}, \text{FIE}, \text{TE}, \text{ThO})$
Construct $\mathbf{A}_0$ and $\mathbf{B}_0$
$\{(\lambda_{j,0}, u_{j,0})\}_{j=1}^{\text{DTE+FIE}} :=$ Eigensolver$(\mathbf{A}_0, \mathbf{B}_0, \text{DTE} + \text{FIE},$
Tol, MaxIter)
$(\Lambda_0, U_0) := (\lambda_{\text{TE},0}, u_{\text{TE},0})$
$m := 1$
repeat
Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
Construct $\mathbf{A}_m$ and $\mathbf{B}_m$
$\{(\lambda_{j,m}, u_{j,m})\}_{j=\text{FIE}}^{\text{DTE+FIE}} :=$
PicardImpOrtho$(\mathbf{A}_m, \mathbf{B}_m, \{(\lambda_{j,m-1}, u_{j,m-1})\}_{j=\text{FIE}}^{\text{DTE+FIE}}, \text{ThO})$
$(\Lambda_m, U_m) :=$ Reconstruction$(\{(\lambda_{j,m}, u_{j,m})\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}))$
$m := m + 1$
until $m \leq M$ OR $\eta \leq \text{err}$

Similarly we define the adaptive method based on the Newton's method.

Algorithm 12 Adaptive method based on the Newton's method with improved orthogonality

$(\Lambda_M, U_M) :=$ NewtonImpAdapt$(\mathcal{T}_0, V_0, M, \text{err}, \text{Tol}, \text{MaxIter}, \text{DTE}, \text{FIE}, \text{TE}, \text{ThO})$
Construct $\mathbf{A}_0$ and $\mathbf{B}_0$
$\{(\lambda_{j,0}, u_{j,0})\}_{j=1}^{\text{DTE+FIE}} :=$ Eigensolver$(\mathbf{A}_0, \mathbf{B}_0, \text{DTE} + \text{FIE},$
Tol, MaxIter)
$(\Lambda_0, U_0) := (\lambda_{\text{TE},0}, u_{\text{TE},0})$
$m := 1$
repeat
Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
Construct $\mathbf{A}_m$ and $\mathbf{B}_m$
$\{(\lambda_{j,m}, u_{j,m})\}_{j=\text{FIE}}^{\text{DTE+FIE}} :=$
NewtonImpOrtho$(\mathbf{A}_m, \mathbf{B}_m, \{(\lambda_{j,m-1}, u_{j,m-1})\}_{j=\text{FIE}}^{\text{DTE+FIE}}, \text{ThO})$
$(\Lambda_m, U_m) :=$ Reconstruction$(\{(\lambda_{j,m}, u_{j,m})\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}))$
$m := m + 1$
until $m \leq M$ OR $\eta \leq \text{err}$

11. A Priori Convergence Results. In this section we gather together some a priori estimates for eigenvalue problems. The framework used in this section is an extension to the $hp-$case of the a priori results in [9, 10]. Lemma 11.4 contains the a priori convergence results for eigenvalues and eigenfunctions in the *hp* context.

Moreover, in Theorem 11.5 we proved convergence a priori results for the reconstructed pair (Λ_n, U_n).

It follows from the coercivity of the bilinear form $a(\cdot,\cdot)$ that all eigenvalues of (3.2) and all $N = \dim V_n$ eigenvalues of (3.5) are positive. We can order them as $0 < \lambda_1 \leq \lambda_2 \ldots$ and $0 < \lambda_{1,n} \leq \lambda_{2,n} \ldots \leq \lambda_{N,n}$. Moreover, we know (e.g., [4]) that $\lambda_{j,n} \to \lambda_j$, for any j, as $V_n \to H^1(\Omega)$ and (by the minimax principle) that $\lambda_{j,n}$ is monotone non-increasing, i.e.,

$$\lambda_{j,n} \;\geq\; \lambda_{j,m} \;\geq\; \lambda_j, \quad \text{for all} \quad j = 1, \ldots, N, \quad \text{and all} \quad m \geq n. \tag{11.1}$$

The distance of an approximate eigenfunction from the true eigenspace is a crucial quantity in the convergence analysis for eigenvalue problems especially in the case of non-simple eigenvalues.

DEFINITION 11.1. *Given a function $v \in L^2(\Omega)$ and a finite dimensional subspace $\mathcal{P} \subset L^2(\Omega)$, we define:*

$$\text{dist}(v, \mathcal{P})_{0,B} \;:=\; \min_{w \in \mathcal{P}} \|v - w\|_0.$$

Similarly, given a function $v \in H_0^1(\Omega)$ and a finite dimensional subspace $\mathcal{P} \subset H_0^1(\Omega)$, we define:

$$\text{dist}(v, \mathcal{P})_1 \;:=\; \min_{w \in \mathcal{P}} \|v - w\|_1.$$

Now let λ_j be any eigenvalue of (3.2), let $E(\lambda_j)$ denote the (finite dimensional) space spanned by the eigenfunctions of λ_j and set $E_1(\lambda_j) = \{u \in E(\lambda_j) : \|u\|_0 = 1\}$. Let T_{λ_j} denote the orthogonal projection of H^1 onto $E(\lambda_j)$ with respect to the inner product $a(\cdot,\cdot)$.

The next lemma is already in [10] and it shows that both distances have the same minimizer.

LEMMA 11.2. *Let $(\lambda_{j,n}, u_{j,n})$ be an eigenpair of (3.5). Then*

$$\|u_{j,n} - u_j\|_0 = \text{dist}(u_{j,n}, E_1(\lambda_j))_0, \tag{11.2}$$

if and only if

$$\|u_{j,n} - u_j\|_1 = \text{dist}(u_{j,n}, E_1(\lambda_j))_1. \tag{11.3}$$

In order to make further progress we need some assumptions on regularity of solutions of elliptic problems associated with $a(\cdot,\cdot)$. Also from now on we assume that the eigenfunctions of (3.2) are at least in $H^2(\Omega)$ and that the sequence of adapted meshes are at most 1-irregular.

ASSUMPTION 11.3. *We assume that there exists a constant $C_{\text{ell}} > 0$ with the following property. For $f \in L^2(\Omega)$ and with the solution operator $\mathcal{S}$, we have that if $v := \mathcal{S}f \in H_0^1(\Omega)$ solves the problem $a(v,w) = b(f,w)$ for all $w \in H_0^1(\Omega)$, then*

$$\|\mathcal{S}f\|_2 \leq C_{\text{ell}} \|f\|_0, \tag{11.4}$$

where $\|\cdot\|_2$ is the norm in the Sobolev space $H^2(\Omega)$. This is a standard assumption which is satisfied in a wide number of applications such as problems with discontinuous coefficients (see eg. [9] for more references).

From now on we shall let C denote a generic constant which may depend on the true eigenvalues and vectors of (3.2) and other constants introduced above, but is always independent of n, as well as h_n and p_n. The next lemma is an extension of Theorem 3.5 in [10] based on the same arguments and where $hp-$results like [17, Theorem 4.72] are used.

LEMMA 11.4. *Suppose* $1 \leq j \leq \dim V_n$. *Let* λ_j *be an eigenvalue of* (3.2) *with corresponding eigenspace* $E(\lambda_j) \subset H^{1+\mu}(\Omega)$, *for* $\mu > 1$, *of any (finite) dimension and let* $(\lambda_{j,n}, u_{j,n})$ *be an eigenpair of* (3.5). *Then, for a finite element space* V_n *sufficiently rich,*

(i)

$$|\lambda_j - \lambda_{j,n}| \ \leq \ (\mathrm{dist}(u_{j,n}, E_1(\lambda_j))_1)^2; \quad \textit{and} \quad |\lambda_j - \lambda_{j,n}| \ \leq \ C\frac{h_n^{2\mu}}{p_n^{2\mu}}; \quad (11.5)$$

(ii)

$$\mathrm{dist}(u_{j,n}, E_1(\lambda_j))_0 \ \leq \ C\frac{h_n}{p_n}\mathrm{dist}(u_{j,n}, E_1(\lambda_j))_1 \ ; \quad (11.6)$$

(iii)

$$\mathrm{dist}(u_{j,n}, E_1(\lambda_j))_1 \ \leq C\frac{h_n^{\mu}}{p_n^{\mu}}, \quad (11.7)$$

with $1 \leq \mu \leq p_n$

Finally, the next and last theorem shows that also the reconstructed couple (Λ_n, U_n) converges in a similar way to standard computed eigenpair. It is interesting to remind that in general the reconstructed couple (Λ_n, U_n) is not an eigenpair of the discrete problem (3.5).

THEOREM 11.5. *Suppose* $1 \leq j \leq \dim V_n$. *Let* λ_j *be an eigenvalue of* (3.2) *with corresponding eigenspace* $E(\lambda_j)$ *of any (finite) dimension and let* (Λ_n, U_n) *be a reconstructed couple of* (3.5). *Then, for a finite element space* V_n *sufficiently rich,*

(i)

$$|\lambda_j - \Lambda_n| \ \leq \ C\frac{h_n^{2\mu}}{p_n^{2\mu}}; \quad (11.8)$$

(ii)

$$\mathrm{dist}(U_n, E_1(\lambda_j))_0 \ \leq \ C\frac{h_n^{\mu+1}}{p_n^{\mu+1}} \ ; \quad (11.9)$$

(iii)

$$\mathrm{dist}(U_n, E_1(\lambda_j))_1 \ \leq C\frac{h_n^{\mu}}{p_n^{\mu}}, \quad (11.10)$$

with $1 \leq \mu \leq p_n$

Proof. Recalling (8.1), let us denote by $u_i \in E(\lambda_j)$, for each $i \leq R$, where R is the multiplicity of λ_j, the eigenfunctions that minimize both $\mathrm{dist}(u_{i,n}, E_1(\lambda_j))_0$ and $\mathrm{dist}(u_{i,n}, E_1(\lambda_j))_1$. Then denoting by

$$U = \sum_{i=1}^{R} c_i \ u_i,$$

we have that

$$\text{dist}(U_n, E_1(\lambda_j))_0 \leq \|U - U_n\|_0 \leq \sum_{i=1}^{R} \text{dist}(u_{i,n}, E_1(\lambda_j))_0,$$

and similarly we have

$$\text{dist}(U_n, E_1(\lambda_j))_0 \leq \|U - U_n\|_1 \leq \sum_{i=1}^{R} \text{dist}(u_{i,n}, E_1(\lambda_j))_1.$$

Then results (ii) and (iii) comes straightforwardly form Lemma 11.4(ii-iii).

By construction we have that $\Lambda := \sum_{i=1}^{R} c_i^2 \, \lambda_{i,n}$ and from the minimum-maximum principle we have that for all i, $\lambda_j - \lambda_{i,n} \leq 0$, then from (8.1) we have

$$\sum_{i=1}^{R} c_i^2 \, |\lambda_j - \lambda_{i,n}| = \sum_{i=1}^{R} c_i^2 \, (\lambda_{i,n} - \lambda_j) = \Lambda_n - \lambda_j = |\lambda_j - \Lambda_n|,$$

so it is also clear that $\Lambda_n \geq \lambda_j$. Then (11.8) come directly from Lemma 11.4(i). □

12. Numerical Results.

12.1. Orthogonality technologies. In this first set of examples, we would like to show the advantages of the orthogonality technologies presented in Sections 5, 6 and 10. We want to approximate the fifth eigenvalue, and the corresponding eigenfunction, on the square domain $[0, \pi]^2$ just calling the Picard's method as in Algorithm 1 with no orthogonalization and starting with four quadratic elements forming the initial structured mesh. So, as always, we compute the approximation of the first fifth eigenpair on the initial coarse mesh with an eigensolver, then we adapt the mesh for the fifth eigenpair and from that point on we use the Picard's method. As can be seen from the piece of output below, already on the first adapted mesh the Picard's method goes away from the correct value 10 for the eigenvalue and converges to the first eigenvalue:

```
---- Adaptivity step 1:
ndof: 9, ndof_ref: 121
Projecting coarse mesh solution to reference mesh.
Assembling matrices S and M on reference mesh.
Initial guess for eigenvalue on reference mesh: 14.049870798404
---- Picard iter 1, ndof 121, eigenvalue: 10.089937400818,
       picard_err_rel 56.2457%,  picard_abs_rel 3.95993
---- Picard iter 2, ndof 121, eigenvalue: 8.617533724041,
       picard_err_rel 23.0615%,  picard_abs_rel 1.4724
---- Picard iter 3, ndof 121, eigenvalue: 3.262292956407,
       picard_err_rel 92.3173%,  picard_abs_rel 5.35524
---- Picard iter 4, ndof 121, eigenvalue: 2.059320963619,
      picard_err_rel 59.4885%,  picard_abs_rel 1.20297
---- Picard iter 5, ndof 121, eigenvalue: 2.002388093093,
      picard_err_rel 13.1799%,  picard_abs_rel 0.0569329
---- Picard iter 6, ndof 121, eigenvalue: 2.000099686846,
       picard_err_rel 2.64435%,  picard_abs_rel 0.00228841
---- Picard iter 7, ndof 121, eigenvalue: 2.000008353170,
       picard_err_rel 0.5283%,  picard_abs_rel 9.13337e-05
```

```
---- Picard iter 8, ndof 121, eigenvalue: 2.000004708920,
       picard_err_rel 0.105528%,  picard_abs_rel 3.64425e-06
---- Picard iter 9, ndof 121, eigenvalue: 2.000004563515,
       picard_err_rel 0.0210793%,  picard_abs_rel 1.45406e-07
---- Picard iter 10, ndof 121, eigenvalue: 2.000004557713,
       picard_err_rel 0.00421058%,  picard_abs_rel 5.80168e-09
---- Picard iter 11, ndof 121, eigenvalue: 2.000004557482,
       picard_err_rel 0.000841063%,  picard_abs_rel 2.31489e-10
```

This is a clear example of what was predicted in Theorem 4.1.

On the other hand, using the standard orthogonality (Section 5) we cure this problem and the method converges to the correct eigenpair. In Figure 12.1 we show the convergence rate for the fifth eigenvalue. As can be seen, the convergence curve is almost a straight line which means an exponential convergence rate.

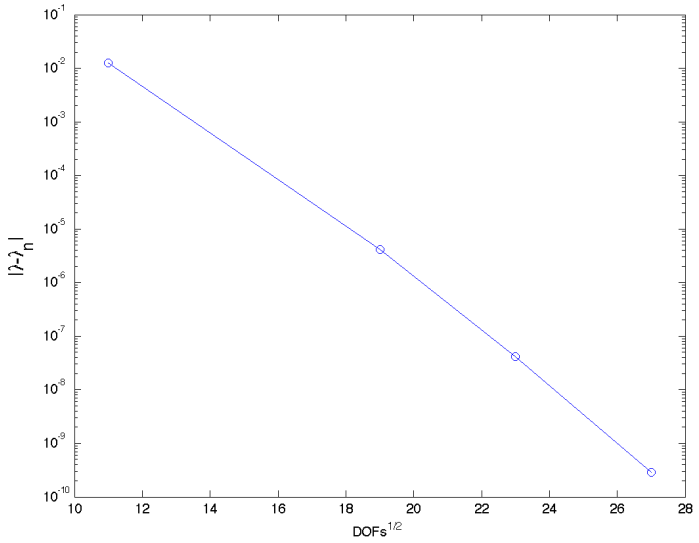

FIG. 12.1. *Convergence plot for the fifth eigenvalue.*

When the standard orthogonality is used, five eigenpairs are computed on each adapted mesh, this means that overall 20 eigenpairs are computed. This is quite expensive, but the cost can be reduced using the improved orthogonality (Section 10). The convergence rate is exactly the same as in Figure 12.1, but the number of computed eigenpairs is dramatically smaller. In Table 12.1 we reported the number of computed eigenpairs on each adapted mesh using either the standard or the improved orthogonality. It is worth mentioning that if we had used the eigensolver on the meshes, we would have computed five eigenpairs for each mesh, as with the standard orthogonality.

12.2. Reconstruction technologies. In the next example we present the benefits of using the reconstruction technology, introduced in Section 8, which makes possible to follow the approximation of the same continuous eigenfunction on a series of adaptively refined meshes, even if the corresponding eigenvalue has multiplicity greater than 1. The fact that, via the reconstruction, there are no changes in the

n	Standard	Improved
1	5	2
2	5	2
3	5	2
4	5	2
Total	20	8

TABLE 12.1
Number of computed eigenpairs for different orthogonality technologies.

approximations, should lead to a decrease in the computational cost, because changes in the approximations, like the one presented in Figures 2.2, 2.3, could mislead the adaptive procedure to introduce more degrees of freedom in regions that are not going to be useful to reduce the error once the approximation is changed.

In Table 12.2 we compare the results between Algorithm 12 and using an eigensolver without the reconstruction technology on each refined mesh, both with hp-adaptivity. In particular in Table 12.2 we compare for each adaptively refined mesh of index n the DOFs and the error in percentage for the 8th eigenpair (which belongs to a double eigenvalue) of problem (2.1), starting with a mesh of 64 square elements and starting with order of polynomials 1. We also set the target tolerance for the error to 0.3. As can be seen the Newton method takes only 12 refinement of the mesh to reach the target tollerance, compared to 15 for using the eigensolver. Also, between the 3rd and the 4th refined meshes, the error actually increase using the eigensolver, this is due to the fact that the approximation is changed dramatically between those two meshes and so the 4th mesh, which was refined using the information from the approximation on the 3rd mesh, is not very good to describe the approximation on the computed in the 4th mesh. Since the approximated continuous eigenfunction never changes using the reconstruction, there is no sign of any oscillation in the error for the Newton method.

12.3. Approximating eigenfunctions on individual meshes. Next we would like to illustrate that in general each eigenfunction should be approximated on its own mesh. We choose a classical L-shape domain example for the Laplace operator where the first eigenfunction exhibits a singularity in the gradient at the re-entrant corner while the second one is completely smooth. The differences in the regularity are reflected in the adapted meshes: For the first eigenfunction a great amount of h-refinement takes place at the re-entrant corner. For the second eigenfunction we have an adapted mesh mostly characterized by p-refinement. These results are shown in Figures 12.2 and 12.3.

12.4. Domains with few reentering corners. We conclude the numerics section considering the model problem (2.1) on the square domain Ω with a square hole. Assuming that we are interested in the third eigenfunction, see Figure 2.2(b), we want to compare adapted meshes using either the eigensolver on each refined mesh or Algorithm 12, which is the most sophisticated incarnation of the Newton's method presented in this paper. In Figure 12.4 we reported the resulting adapted meshes. As can be seen the mesh in Figure 12.4(b) present a gradient in the size of the elements toward the reentering corners where the singularities are located (compare the mesh with Figure 2.2(b)). On the other hand the mesh in Figure 12.4(a) does not present any particular adaptive pattern around the reentering corners. This is due to the

	Eigensolver		Newton	
n	Dofs	Error %	Dofs	Error %
1	961	38.8308	961	31.0305
2	1233	21.5207	1201	21.8555
3	1461	16.5425	1501	16.0894
4	1693	20.9316	1777	10.9197
5	1813	13.9295	2081	6.55891
6	2033	7.59609	2361	4.28255
7	2305	3.99428	2717	3.01209
8	2661	3.16526	3149	2.0103
9	3037	2.76441	3421	1.32503
10	3257	1.87542	3721	0.782314
11	3541	1.4761	4161	0.401284
12	3721	0.7925	4565	0.293117
13	4033	0.4206	-	-
14	4569	0.3087	-	-
15	4925	0.2673	-	-

TABLE 12.2
Comparison between the Newton's method and an eigensolver.

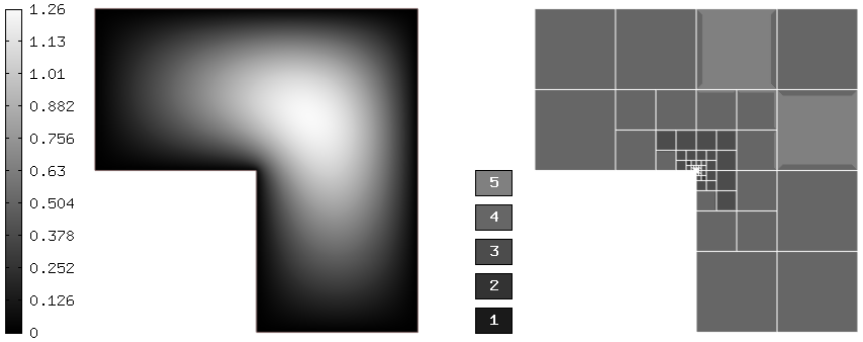

FIG. 12.2. *First eigenfunction for the L-shaped domain and corresponding adapted mesh.*

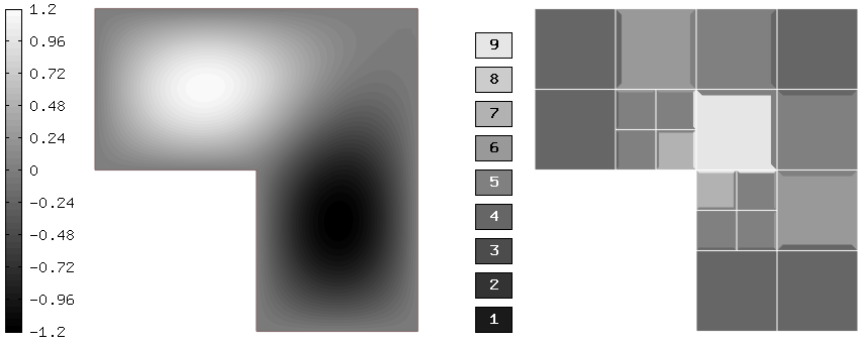

FIG. 12.3. *Second eigenfunction for the L-shaped domain and corresponding adapted mesh.*

phenomenon already described in the motivation of this paper, Section 2, where it was explained that the position in the spectrum of the second and third eigenfunctions could change refining the mesh. Because such phenomenon has happened very often in the simulation, the adaptive procedure was unable to target any particular singularity. Instead, using the reconstruction technology, the target eigenfunction was fixed and so, it was very easy for the adaptive procedure to adapt the mesh accordingly. This phenomenon has also implications in the convergence rate, as can be seen in Figure 12.5 where the number of degrees of freedoms are plotted against the error for the third eigenvalue. It is clear from that picture that Algorithm 12 converges much faster compared to a standard eigensolver. The curves in Figure 12.5 seem to approximate straight lines, which suggests exponential convergence rate. In order to confirm that we included Figure 12.6, where we plot the same curves, but in a log-log scale. Looking at Figure 12.6 it is clear that the convergence in both cases is faster than polynomial. Finally, in Figure 12.7 we compare the convergence rates of both Algorithm 12 and the eigensolver in term of number of mesh refinements n. The gap in this case between the two methods is even bigger and it is clear that Algorithm 12 performs much better.

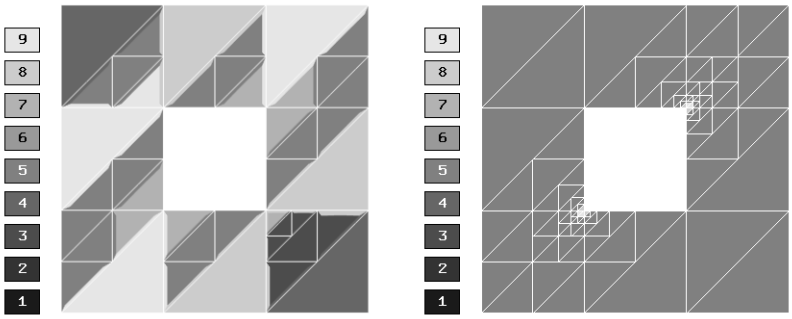

FIG. 12.4. *Adapted meshes using either (a) the eigensolver or (b) Algorithm 12.*

13. Reproducibility of Results. The method presented in this paper is part of the open source C++ library Hermes (http://hpfem.org/hermes), and it can be found in example "eigen-adapt-iter". If the reader has a problem with building the library on his/her computer, or with running this example, should send a message to the mailing list hermes2d@googlegroups.com. For a model implementation of the Anderson acceleration technique mentioned in Section 5 visit the Networked Computing Laboratory (NCLab) at http://nclab.com where it is part of the Published Worksheet "Fixed Point Iteration (with Acceleration)". In NCLab, anyone can freely experiment with this technique and other numerical methods in his/her web browser.

14. Conclusion and Outlook. We presented a novel adaptive higher-order finite element method (*hp*-FEM) for PDE eigenproblems. As opposed to conventional methods, it does not need to call an eigensolver in each adaptivity step. This eliminates standard problems associated with repeated eigenvalues. The technique also makes it possible to approximate different eigenfunctions on individual meshes. The fact that one mesh cannot be optimal for multiple eigenfunctions at the same time was the original motivation for this study. Another motivation is that using the present method, many eigenfunctions can be calculated separately in parallel, without a parallel eigensolver. In our next steps we will continue along this line and adjust the

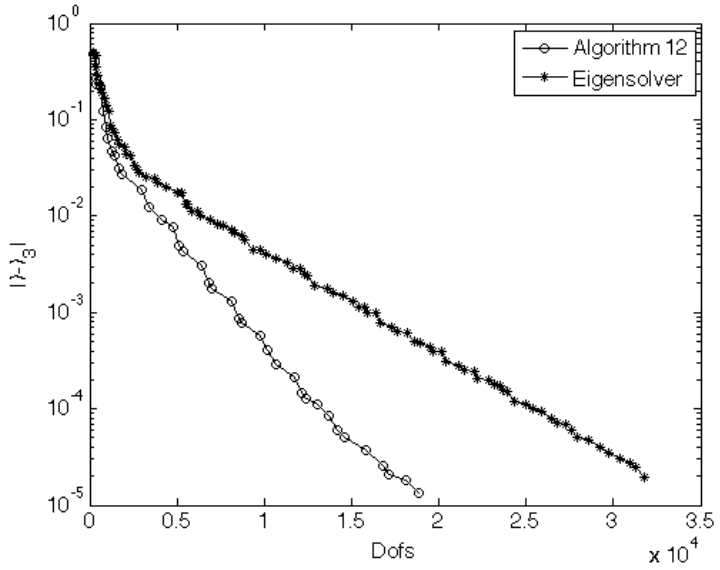

FIG. 12.5. *Convergence plot for the third eigenvalue.*

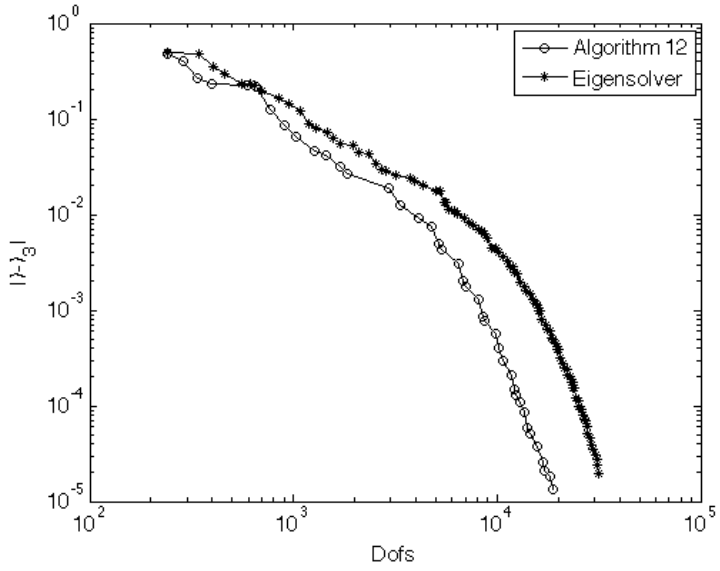

FIG. 12.6. *Convergence plot for the third eigenvalue in log-log scale.*

adaptive multimesh *hp*-FEM technique [19] to approximate multiple eigenfunctions adaptively on individual meshes.

Acknowledgments. The first author was supported by the Subcontract No. 00089911 of Battelle Energy Alliance (U.S. Department of Energy intermediary) as well as by the Grant No. IAA100760702 of the Grant Agency of the Academy of Sciences of the Czech Republic.

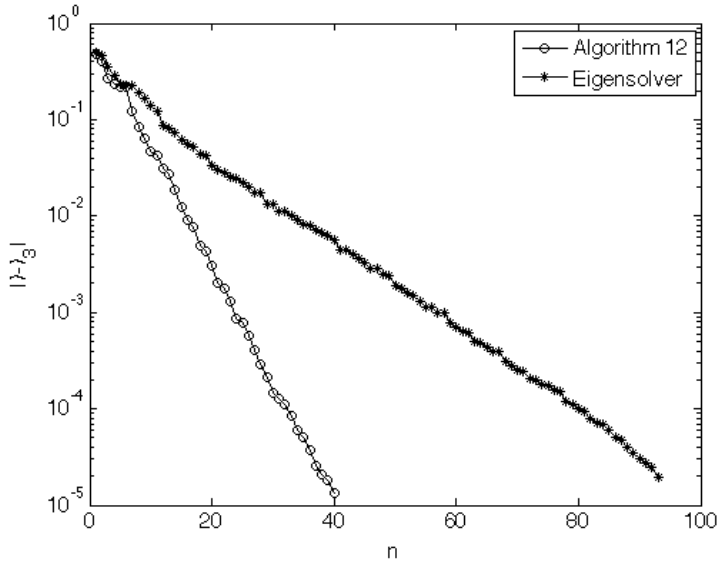

FIG. 12.7. *Convergence plot for the third eigenvalue in term of number of mesh refinements.*

REFERENCES

[1] D. G. ANDERSON, *Iterative procedures for nonlinear integral equations,* J. Assoc. Comput. Machinery, 12 (1965), pp. 547–560.

[2] I. BABUŠKA AND J. OSBORN, *Estimates for the errors in eigenvalue and eigenvector approximation by Galerkin methods, with particular attention to the case of multiple eigenvalues*, SIAM J. Numer. Anal., 24 (1987), pp. 1249–1276.

[3] I. BABUŠKA AND J. OSBORN, *Eigenvalue problems*, in Handbook of Numerical Analysis Vol II, eds P.G. Cairlet and J.L. Lions, North Holland, 1991.

[4] I. BABUŠKA AND J. OSBORN, *Finite element-Galerkin approximation of the eigenvalues and eigenvectors of selfadjoint problems*, Math. Comput. 186 (1989), pp. 275–297.

[5] P. BERINI, *Plasmon-polariton waves guided by thin lossy metal films of finite width: Bound modes of symmetric structures*, Physical Review B 61:15 (2000), pp.10484+.

[6] A. CLIFFE, E. HALL AND P. HOUSTON, *Adaptive discontinuous Galerkin methods for eigenvalue problems arising in incompressible fluid flows*, SIAM Journal on Scientific Computing, 31:6 (2010), pp. 4607–4632.

[7] L. DUBCOVA, P. SOLIN, G. HANSEN AND H. PARK, *Comparison of multimesh hp-fem to interpolation and projection methods for spatial coupling of reactor thermal and neutron diffusion calculations*, J. Comput. Phys. 230 (2011), pp. 1182–1197.

[8] S. GIANI, *Convergence of Adaptive Finite Element Methods for Elliptic Eigenvalue Problems with Application to Photonic Crystals*, Ph.D. thesis, Department of Mathematical Sciences, University of Bath, 2008.

[9] S. GIANI AND I. G. GRAHAM, *A convergent adaptive method for elliptic eigenvalue problems*, SIAM J. Numer. Anal. 47:2 (2009), pp. 1067–1091.

[10] S. GIANI AND I. G. GRAHAM, *Adaptive finite element methods for computing band gaps in photonic crystals*, Numerische Mathematik, submitted.

[11] L. GRUBIŠIĆ AND J. S. OVALL, *On estimators for eigenvalue/eigenvector approximations*, Mathematics of Computation, 78:266 (2008), pp. 739–770.

[12] W. HACKBUSCH, *Elliptic Differential Equations*, Springer, Berlin, 1992.

[13] J. D. JOANNOPOULOS, R. D. MEADE, AND J. N. WINN, *Photonic crystals. Molding the flow of light* Princeton Univ. Press, Princeton, NJ, 1995.

[14] N. LALOR AND H-H. PRIEBSCH, *The prediction of low- and mid-frequency internal road vehicle noise: a literature survey*, in Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering 221:3 (2007), pp. 245–269.

[15] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted arnoldi methods*, SIAM, 1998.
[16] H. Röck, *Finfding an eigenvector and eigenvalue, with Newtons method for solving systems of nonlinear equations*, Report CMDIE WS2002/03
[17] C. Schwab, *p- and hp- finite element methods*, Oxford University Press, Oxford, 1998.
[18] P. Solin, D. Andrs, J. Cerveny And M. Simko, *PDE-independent adaptive hp-fem based on hierarchic extension of finite element spaces*, J. Comput. Appl. Math., 233 (2010), pp. 3086–3094.
[19] P. Solin, J. Cerveny, L. Dubcova And D. Andrs, *Monolithic discretization of linear thermoelasticity problems via adaptive multimesh hp-FEM*, J. Comput. Appl. Math, 234 (2010), pp. 2350–2357.
[20] P. Solin, K. Segeth and I. Dolezel, *Higher-order finite element methods*, Chapman & Hall, CRC Press, London, 2003.
[21] G. Strang and G. J. Fix, *An analysis of the finite element method*, Prentice-Hall, 1973.