# AN ADAPTIVE HOMOTOPY APPROACH FOR NON-SELFADJOINT EIGENVALUE PROBLEMS

C. CARSTENSEN[†⋆], J. GEDICKE[†‡], V. MEHRMANN[†], AND A. MIEDLAR[‡†]

ABSTRACT. This paper presents three different adaptive algorithms for eigenvalue problems associated with non-selfadjoint partial differential operators. The basis for the developed algorithms is a homotopy method. The homotopy method starts from a well-understood selfadjoint problem, for which well-established adaptive methods are available. Apart from the adaptive grid refinement, the progress of the homotopy as well as the solution of the iterative method are adapted to balance the contributions of the different error sources. The first algorithm balances the homotopy, discretization and approximation errors with respect to a fixed step-size $\tau$ in the homotopy. The second algorithm combines the adaptive step-size control for the homotopy with an adaptation in space that ensures an error below a fixed tolerance $\varepsilon$. The third algorithm allows the complete adaptivity in space, homotopy step-size as well as the iterative algebraic eigenvalue solver. All three algorithms are compared in numerical examples.

September 3, 2010

## 1. INTRODUCTION

Non-selfadjoint eigenvalue problems associated with partial differential operators arise in a large number of applications, such as acoustic field computations [ADRPR01], structural analysis of buildings or vehicles [HZS+04], electric and magnetic field computation [BFGP99]. Today, in almost all applications the space is discretized first which leads to a linear or nonlinear matrix eigenvalue problem. To solve these algebraic eigenvalue problems, classical eigenvalue methods [BDD+00, GV96, LSY98, Par98] are used. Typically the problems are discretized in space on very fine grids that lead to a high computational effort for the matrix eigenvalue solver.

In recent years there have been tremendous research activities to design adaptive eigenvalue methods that adapt the grid to the behavior of the eigenfunctions in order to avoid unnecessarily fine grids. For selfadjoint elliptic problems the progress in the analysis and computational methods has been substantial.

A priori error estimates for eigenvalues and eigenvectors of elliptic operators and compact operators were developed , e.g., in [BO89, OB91, Cha83, Kny97, RT83, SF73, XZ01]. All these approaches, although optimal, contain mesh size restrictions, which cannot be verified or quantified, neither a priori nor a posteriori. Verifiable a priori error estimates for symmetric eigenvalue problems were presented in [AKPP08, KA10, KO06], see also [LT03, RT83]. A first approach on a posteriori error analysis for symmetric second order elliptic eigenvalue problems can be found

*2010 Mathematics Subject Classification.* 65F15, 65N15, 65N25, 65N50, 65M60, 65H20.

*Key words and phrases.* eigenvalue problem, adaptive finite element method, homotopy.

† Supported by the DFG Research Center MATHEON "Mathematics for Key Technologies" in Berlin.

‡ Supported by the DFG graduate school BMS "Berlin Mathematical School" in Berlin.

⋆ World Class University (WCU) program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology R31-2008-000-10049-0.

in [Ver96]. A combination of a posteriori and a priori analysis was used in [Lar00] to prove reliable and efficient a posteriori estimates for $H^2$ regular problems. For non-smooth solutions a posteriori error estimators were given in [DPR03, MSZ06, Ney02]. Recent results include [CG08, GMZ09, GG09, GO09, Sau10]. Surveys about a posteriori error estimation can be found in [AO00, BS02, Ver96].

Almost all these results are for elliptic selfadjoint problems, an approach for non-selfadjoint elliptic eigenvalue problems was presented in [HR01]. The difficulty with non-selfadjoint PDE eigenvalue problems is multifold, eigenvalues may be complex, or may have different algebraic and geometric multiplicity. The latter property is a particular difficulty for the discretization methods because in the finite dimensional approximation these properties may be destroyed, and even though the approximation error is small the computed eigenvalues/eigenfunctions may have much larger errors due to the ill-conditioning of the problem. Even when the discretization retains the multiplicities of the eigenvalues, the algebraic eigensolvers have difficulties with the ill-conditioning of multiple eigenvalues. At this stage the adaptive solution of general non-selfadjoint eigenvalue problems remains a real challenge.

In this paper we will study only the restricted class of convection-diffusion eigenvalue problems, where for the pure diffusion problem the discussed adaptive methods work nicely. To design an adaptive algorithm for the convection-diffusion problem we will employ a homotopy method. Homotopy methods are well established for nonsymmetric matrix eigenvalue problems [LZ92, LZC92, LKK97, LZ99]. The homotopy approach will be used not only on the matrix level but on the level of the differential operator as well. In order to combine the adaptive homotopy with mesh adaptivity and iterative matrix eigenvalue solvers, we have to deal with three different types of errors. These are the *discretization error* $\eta$ that arises when the infinite dimensional variational problems is considered in a finite dimensional subspace [GC09, HR01], the *homotopy error* $\nu$ that arises because the diffusion problem is slowly transferred to the convection-diffusion problem [BE03] and the *approximation error* $\mu$ that arises from the iterative matrix eigensolver in finite precision arithmetic [BDD+00, Par98, SS90, HL06]. Since the goal is to design methods that are both accurate and efficient we will develop algorithms that are able to provide adaptivity in all three directions by a suitable balancing of all three errors.

As a model problem we consider the following convection-diffusion eigenvalue problem:
*Determine a non-trivial eigenpair* $(\lambda, u) \in \mathbb{C} \times H_0^1(\Omega;\mathbb{C}) \cap H_{loc}^2(\Omega;\mathbb{C})$ *with* $\|u\|_{L^2(\Omega;\mathbb{C})} = 1$ *such that*

$$-\Delta u + \beta \cdot \nabla u = \lambda u \text{ in } \Omega \qquad \text{and} \qquad u = 0 \text{ on } \partial\Omega \tag{1.1}$$

*for some bounded Lipschitz domain* $\Omega \subseteq \mathbb{R}^2$ *and a constant coefficient vector* $\beta \in \mathbb{R}^2$.

We will study the problem in its weak formulation:
*For two complex Hilbert spaces* $V := H_0^1(\Omega;\mathbb{C})$ *with norm* $|||\cdot||| := |\cdot|_{H^1(\Omega;\mathbb{C})}$ *and* $H := L^2(\Omega;\mathbb{C})$ *with norm* $\|\cdot\|_{L^2(\Omega;\mathbb{C})}$ *determine a non-trivial eigenpair* $(\lambda, u) \in \mathbb{C}\times V$ *with* $b(u,u) = 1$ *such that*

$$a(u,v) + c(u,v) = \lambda b(u,v) \quad \text{for all} \quad v \in V, \tag{1.2}$$

*where*

$$a(u,v) := \int_\Omega \nabla u \nabla \overline{v} dx, \quad c(u,v) := \int_\Omega \overline{v}(\beta \cdot \nabla u) dx, \quad b(u,v) := \int_\Omega u\overline{v} dx,$$

*and* $\overline{(.)}$ *denotes complex conjugation.*

As $\beta \in \mathbb{R}^2$ it is straightforward that $\beta$ is divergence free. Therefore, the Gauss theorem shows that

$$\int_\Omega (\beta \cdot \nabla v)\overline{v}\, dx = -\int_\Omega v(\beta \cdot \nabla \overline{v})\, dx \quad \text{for all} \quad v \in V,$$

and hence $\mathrm{Re}(a(v,v) + c(v,v)) = \mathrm{Re}(a(v,v)) = |\!|\!| v |\!|\!|^2$. This implies that the bilinear form $a(\cdot,\cdot) + c(\cdot,\cdot)$ is elliptic, with ellipticity constant 1 independent of $\beta$ and continuous in V, since $|\beta|$ is bounded. The bilinear form $b(\cdot,\cdot)$ is continuous, symmetric and positive definite, and hence induces a norm $\|\cdot\| := b(\cdot,\cdot)^{1/2}$ on $H$. For this model problem $|\!|\!|\cdot|\!|\!| = a(\cdot,\cdot)^{1/2}$ and $\|\cdot\| = \|\cdot\|_{L^2(\Omega;\mathbb{C})}$.

For the analysis and the adaptive method of this non-selfadjoint eigenvalue problems it is necessary to consider also the dual eigenvalue problem:
*Determine a non-trivial dual eigenpair* $(\lambda^\star, u^\star) \in \mathbb{C} \times V$ *with* $b(u^\star, u^\star) = 1$ *such that*

$$a(w, u^\star) + c(w, u^\star) = \overline{\lambda^\star} b(w, u^\star) \quad \textit{for all} \quad w \in V. \tag{1.3}$$

Note that the primal and dual eigenvalues are connected via $\lambda = \overline{\lambda^\star}$.

For a finite dimensional subspace $V_\ell \subseteq V$ the discretized primal and dual problems read:
*Determine non-trivial primal and dual eigenpairs* $(\lambda_\ell, u_\ell) \in \mathbb{C} \times V_\ell$ *and* $(\lambda_\ell^\star, u_\ell^\star) \in \mathbb{C} \times V_\ell$ *such that*

$$a(u_\ell, v_\ell) + c(u_\ell, v_\ell) = \lambda_\ell b(u_\ell, v_\ell) \quad \textit{for all} \quad v_\ell \in V_\ell, \tag{1.4}$$

$$a(w_\ell, u_\ell^\star) + c(w_\ell, u_\ell^\star) = \overline{\lambda_\ell^\star} b(w_\ell, u_\ell^\star) \quad \textit{for all} \quad w_\ell \in V_\ell. \tag{1.5}$$

In view of the difficulties for non-selfadjoint problems discussed before, we will focus in the following on the simpler special situation that the eigenvalue of interest $\lambda$ is simple and well-separated from the rest of the spectrum.

To distinguish continuous, discrete and approximated eigenvalues, some further notation is introduced. In the following $\lambda(t)$ will denote the continuous eigenvalue of interest at homotopy step $t$, $\lambda_\ell(t)$ the corresponding eigenvalue of the discrete problem, while $\tilde{\lambda}_\ell(t)$ denotes its approximation computed by an iterative eigenvalue solver in finite precision arithmetic. The corresponding eigenfunctions are denoted in a similar fashion, i.e., $u(t)$, $u_\ell(t)$, $\tilde{u}_\ell(t)$. In order to distinguish the eigenfunction $u_\ell(t)$ from the corresponding coefficient vector with respect to a given finite element basis, for this eigenvector $\mathbf{u}_\ell(t)$ bold letters will be used. For all these eigenvalues and eigenfunctions or eigenvectors $^\star$ denotes the solution of the corresponding dual problem, i.e., for the algebraic eigenvalue problem $\mathbf{u}_\ell^\star(t)$ denotes the corresponding left eigenvector.

We will also use the common notation $x \lesssim y$ for $x \leq Cy$ with a constant $C$ independent of the mesh size.

The paper is organized as follows: Section 2 reviews the adaptive finite element method (AFEM) and Section 3 discusses the homotopy method. The homotopy error is presented in Section 4. In Section 5 a complete a posteriori error estimator for all three different error sources is presented. In Section 6 several different adaptive homotopy algorithms are developed. A comparison of the performance for the different algorithms is presented in Section 7 via several numerical examples.

## 2. Adaptive finite element methods

In this section we review the basic concept of the adaptive finite element method (AFEM). Starting from an initial coarse triangulation $\mathcal{T}_0$, the AFEM generates a

sequence of nested triangulations $\mathcal{T}_0, \mathcal{T}_1, \ldots$ with corresponding nested spaces

$$V_0 \subseteq V_1 \subseteq \ldots \subseteq V_\ell \subset V.$$

A typical AFEM loop consists of the four steps

$$\texttt{Solve} \longrightarrow \texttt{Estimate} \longrightarrow \texttt{Mark} \longrightarrow \texttt{Refine}.$$

**Solve.** In the step `Solve` the primal and dual generalized algebraic eigenvalue problems

$$(A_\ell + C_\ell)\mathbf{u}_\ell = \lambda_\ell B_\ell \mathbf{u}_\ell \qquad \text{and} \qquad \mathbf{u}_\ell^\star (A_\ell + C_\ell) = \lambda_\ell^\star \mathbf{u}_\ell^\star B_\ell \tag{2.1}$$

are solved, where the coefficient matrices are the symmetric positive definite stiffness matrix $A_\ell$, the nonsymmetric convection matrix $C_\ell$ and the symmetric positive definite mass matrix $B_\ell$. The right and left eigenvectors $\mathbf{u}_\ell = [\mathbf{u}_{\ell,k}]$ and $\mathbf{u}_\ell^\star = [\mathbf{u}_{\ell,k}^\star]$ represent the eigenfunctions

$$u_\ell = \sum_{k=1}^{\dim(V_\ell)} \mathbf{u}_{\ell,k}\varphi_k \quad \text{and} \quad u_\ell^\star = \sum_{k=1}^{\dim(V_\ell)} \mathbf{u}_{\ell,k}^\star \varphi_k.$$

with respect to the basis $\operatorname{span}\{\varphi_1, \ldots, \varphi_{\dim(V_\ell)}\} = V_\ell$.

**Estimate.** At this step the discretization error is estimated. The eigenvalue error is estimated a posteriori with a standard residual type error estimator using the residuals for both, the primal and dual, eigenfunctions. The proof of *reliability*, i.e., that the estimator is an upper bound of the eigenvalue error, can be found in [HR01, GC09], where it is shown, that

$$|\lambda - \lambda_\ell| \lesssim \sum_{T\in\mathcal{T}_\ell} \big(\eta_\ell^2(T) + \eta_\ell^{\star 2}(T)\big). \tag{2.2}$$

Here the primal $\eta_\ell$ and dual $\eta_\ell^*$ refinement indicators for a triangle $T \in \mathcal{T}_\ell$ are defined as

$$\begin{aligned}
\eta_\ell^2(T) &:= h_T^2 \|\beta\cdot\nabla u_\ell - \lambda_\ell u_\ell\|_{L^2(T)}^2 + \sum_{E\in\mathcal{E}_\ell(T)} h_E \|[\nabla u_\ell]\cdot n_E\|_{L^2(E)}^2, \\
\eta_\ell^{\star 2}(T) &:= h_T^2 \|-\beta\cdot\nabla \overline{u_\ell^\star} - \overline{\lambda_\ell^\star u_\ell^\star}\|_{L^2(T)}^2 + \sum_{E\in\mathcal{E}_\ell(T)} h_E \|[\nabla \overline{u_\ell^\star}]\cdot n_E\|_{L^2(E)}^2,
\end{aligned}$$

where $\mathcal{E}_\ell(T)$ denotes the set of all edges for an element $T \in \mathcal{T}_\ell$, $h_E$ is the length of the edge $E$, $h_T$ is the diameter of the triangle $T$, $n_E$ denotes an unit normal for the edge $E$, and $[\cdot]$ denotes the jump across some edge $E$ defined as $[v] := v|_{T_+} - v|_{T_-}$, $v \in V$, for two neighboring triangles $T_\pm \in \mathcal{T}_\ell$ with $E = T_+ \cap T_-$.

Note that the constant in the a posteriori error estimate (2.2) depends on the eigenvalue condition number $1/b(u, u^\star)$ [GC09].

**Mark.** Based on the refinement indicators the set of elements $\mathcal{M}_\ell \subseteq \mathcal{T}_\ell$ that are refined is specified in the algorithm `Mark`. Let $\mathcal{M}_\ell$ be the set of minimal cardinality for which the bulk criterion [Dör96],

$$\theta \sum_{T\in\mathcal{T}_\ell} \big(\eta_\ell^2(T) + \eta_\ell^{\star 2}(T)\big) \le \sum_{T\in\mathcal{M}_\ell} \big(\eta_\ell^2(T) + \eta_\ell^{\star 2}(T)\big)$$

is satisfied for a given bulk parameter $0 < \theta \le 1$. This minimal set $\mathcal{M}_\ell$ may be computed by a greedy algorithm. Sorting all the values $(\eta_\ell^2(T) + \eta_\ell^{\star 2}(T))_{T\in\mathcal{T}_\ell}$ in ascending order allows to add elements with largest values successively to the set $\mathcal{M}_\ell$ until the bulk criterion is fulfilled.

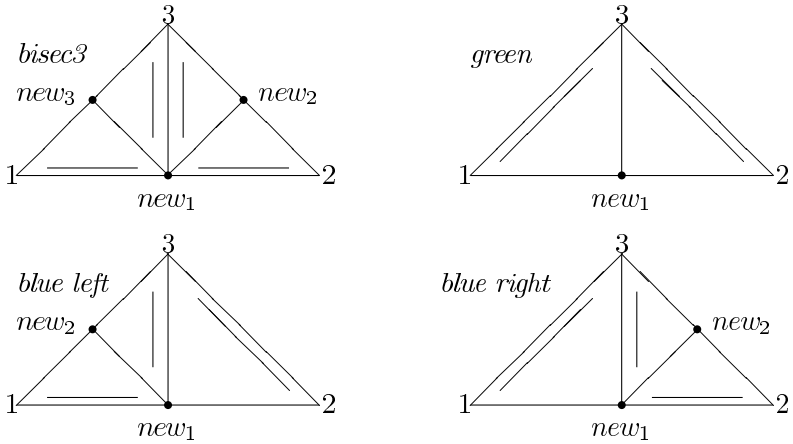


FIGURE 2.1. *Bisec3*, *green* and *blue* refinement. The new reference edge is marked through a second line in parallel opposite the new vertices $new_1$, $new_2$ or $new_3$.

**Refine.** Given a triangulation $\mathcal{T}_\ell$ on the level $\ell$, let $\mathcal{E}_\ell$ denote its set edges and let $E(T)$ denote the reference edge for a given triangle $T$. Note that the reference edge $E(T)$ will be the same edge of $T$ in all triangulations $\mathcal{T}_\ell$ which include $T$. However, once $T$ in $\mathcal{T}_\ell$ is refined, the reference edges will be specified for the different sub-triangles as indicated in Figure 2.1. To preserve the quality of the mesh, a closure algorithm computes the smallest subset $\widehat{\mathcal{M}}_\ell$ of $\mathcal{E}_\ell$ which includes all edges of elements in $\mathcal{M}_\ell$ and reference edges $E(T)$ such that

$$\Big\{E(T) : T \in \mathcal{T}_\ell \quad \text{with} \quad \mathcal{E}_\ell(T) \cap \widehat{\mathcal{M}}_\ell \neq \emptyset\Big\} \subseteq \widehat{\mathcal{M}}_\ell.$$

In other words, once an edge $E$ of an element $T$ is marked for refinement, the reference edge $E(T)$ of $T$ is marked as well. The mesh-refinement method `Refine` then consists of the following five different refinements. Elements with no marked edge are not refined, elements with one marked edge are refined *green*, elements with two marked edges are refined *blue*, and elements with three marked edges are refined *bisec3* as depicted in Figure 2.1.

For further details on adaptive finite element methods for several problems see [AO00, BR03, BS02, Ver96].

## 3. Homotopy methods

In this section we will discuss homotopy methods and extend them from the matrix eigenvalue problem to the operator problem. Homotopy methods in the context of nonsymmetric matrix eigenvalue problems are discussed in [LZ92, LZ99, LZC92, LKK97]. In [LG95] an extension to the eigenvalue problem for selfadjoint partial differential operators is presented.

In the matrix case, from the eigenvalues and eigenvectors of some known matrix $A_0$, for a given function $f : [0,1] \to [0,1]$ with $f(0) = 0,\ f(1) = 1$, the eigenvalues and eigenvectors of

$$\mathcal{H}(t) = (1 - f(t))A_0 + f(t)A_1 \quad \text{for} \quad 0 \le t \le 1 \tag{3.1}$$

can be computed by following their paths from 0 to 1. In the following we will discuss the case $f(t) = t$, but in practice, the function $f$ should grow faster towards $t = 1$ to improve the convergence of the homotopy method.

The homotopy concept can be easily extended to the convection-diffusion operator eigenvalue problem. Starting from the spectrum of some known operator, e.g.,

from $\mathcal{L}_0 u := -\Delta u$, one may use a continuation method to obtain the eigenpairs for the convection-diffusion operator $\mathcal{L}_1 u := -\Delta u + \beta \cdot \nabla u$.

Throughout the paper the following homotopy equation is considered for the model problem (1.1)

$$\mathcal{H}(t) = (1-t)\mathcal{L}_0 + t\mathcal{L}_1 \quad \text{for} \quad 0 \le t \le 1. \tag{3.2}$$

Since for $t = 0$ we have

$$\mathcal{H}(0) = \mathcal{L}_0,$$

the eigenpairs of $\mathcal{H}(0)$ are the eigenpairs for the Laplace eigenvalue problem. The continuation method uses a 'time'-stepping procedure with nodes $t_0 = 0 < t_1 < \ldots < t_N = 1$ to compute the eigenvalues and eigenvectors of

$$-\Delta u + t_i \beta \cdot \nabla u = \lambda u \quad \text{in} \quad \Omega.$$

Finally when the homotopy reaches its final value 1, the eigenpairs of $\mathcal{H}(1) = \mathcal{L}_1$, are the eigenpairs of the desired problem,

$$-\Delta u + \beta \cdot \nabla u = \lambda u \quad \text{in} \quad \Omega.$$

For each step $t_i$ the corresponding weak finite dimensional primal and dual problems

$$\begin{aligned} a(u_\ell, v_\ell) + t_i\, c(u_\ell, v_\ell) &= \lambda_\ell b(u_\ell, v_\ell) \quad \text{for all} \quad v_\ell \in V_\ell, \\ a(w_\ell, u_\ell^\star) + t_i c(w_\ell, u_\ell^\star) &= \overline{\lambda_\ell^\star} b(w_\ell, u_\ell^\star) \quad \text{for all} \quad w_\ell \in V_\ell, \end{aligned}$$

lead to the generalized primal and dual matrix eigenvalue problems

$$(A_\ell + t_i\, C_\ell)\mathbf{u}_\ell = \lambda_\ell B_\ell \mathbf{u}_\ell, \tag{3.3}$$

$$\mathbf{u}_\ell^\star (A_\ell + t_i\, C_\ell) = \lambda_\ell^\star \mathbf{u}_\ell^\star B_\ell, \tag{3.4}$$

corresponding to the discrete homotopy equation

$$\mathcal{H}_\ell(t) = (1-t)A_\ell + t(A_\ell + C_\ell) = A_\ell + tC_\ell.$$

For the case considered here, of simple and well-separated eigenvalues that do not bifurcate during the homotopy process, it is known [Kat82] that every eigenvalue $\lambda_\ell(t)$ of the generalized eigenvalue problems (3.3) and (3.4) is an analytic function in $t$. Hence by choosing appropriate homotopy step-sizes, the eigenvalues can be continued on an analytic path towards the eigenvalues of $(A_\ell + C_\ell, B_\ell)$, see [LKK97, LZ99]. The evolution of an eigenpair as a function of $t$ is called an *eigenpath* and is denoted by $(\lambda_\ell(t), \mathbf{u}_\ell(t))$ and $(\lambda_\ell^\star(t), \mathbf{u}_\ell^\star(t))$, respectively.

## 4. Homotopy Error

In this section we analyze the homotopy error which in another context is called *modeling error* [BE03]. As we solve at the beginning of the homotopy process first the selfadjoint problem and on the matrix level symmetric generalized eigenvalue problem we need to understand how the real eigenvalues of the symmetric problem move to the (potentially complex conjugate) eigenvalues of the final problem. For this we need to bound the homotopy error between the eigenvalues of the initial symmetric and of the final nonsymmetric problem via some a posteriori error estimator, i.e.,

$$|\lambda(1) - \lambda(t)| \lesssim \nu(t) \quad \text{for} \quad 0 \le t \le 1.$$

We have the following bound for the operator eigenvalue problem.

**Lemma 4.1.** *For the model problem (1.1), the difference between the exact eigenvalues $\lambda(t)$ of the homotopy $\mathcal{H}(t)$ in (3.2) and $\lambda(1)$ can be estimated via*

$$|\lambda(1) - \lambda(t)| \lesssim \nu(t) := (1-t)|\beta|_\infty \left( |\!|\!| u(t) |\!|\!| + |\!|\!| u^\star(t) |\!|\!| \right) \quad \textit{for} \quad 0 \le t \le 1. \tag{4.1}$$

*The constant in the inequality tends to $1/(2b(u(1), u^\star(1))$ as $t \to 1$.*

*Proof.* For the homotopy parameter $0 \le t \le 1$, the primal and dual weak eigenvalue problems have the form

$$\begin{aligned} a(u(t),v) + tc(u(t),v) &= \lambda(t) b(u(t),v) \quad \text{for all} \quad v \in V, \\ a(w,u^\star(t)) + tc(w,u^\star(t)) &= \overline{\lambda^\star(t)} b(w,u^\star(t)) \quad \text{for all} \quad w \in V. \end{aligned}$$

Algebraic manipulations yield

$$\begin{aligned} &(\lambda(1)-\lambda(t))\Big(b(u(1),u^\star(1)) + b(u(t),u^\star(t)) - b(u(1)-u(t),u^\star(1)-u^\star(t))\Big) \\ &\quad = (\lambda(1)-\lambda(t))\big(b(u(1),u^\star(t)) + b(u(t),u^\star(1))\big) \\ &\quad = \lambda(1) b(u(1),u^\star(t)) + \overline{\lambda^\star(1)} b(u(t),u^\star(1)) \\ &\qquad - \overline{\lambda^\star(t)} b(u(1),u^\star(t)) - \lambda(t) b(u(t),u^\star(1)) \\ &\quad = (1-t)c(u(1),u^\star(t)) + (1-t)c(u(t),u^\star(1)). \end{aligned}$$

Since $\beta$ is divergence free, it follows that

$$c(u(1),u^\star(t)) = -c(u^\star(t),u(1)).$$

Then the Hölder inequality implies that

$$\begin{aligned} c(u(t),u^\star(1)) - c(u^\star(t),u(1)) &\le \|\beta\cdot\nabla u(t)\|\|u^\star(1)\| + \|\beta\cdot\nabla u^\star(t)\|\|u(1)\| \\ &\le |\beta|_\infty \left(|\!|\!| u(t) |\!|\!| + |\!|\!| u^\star(t) |\!|\!|\right). \end{aligned}$$

Since $b(u(t),u^\star(t))$ tends to $b(u(1),u^\star(1))$ and since $b(u(1)-u(t),u^\star(1)-u^\star(t))$ tends to zero as $t \to 1$, the constant in the eigenvalue error estimate tends to $1/(2b(u(1),u^\star(1))$. □

## 5. A Posteriori Error Estimator

In this section we discuss the a posteriori estimation of the eigenvalue error during the homotopy process. Since in the next section three algorithms are presented which use the homotopy process in combination with mesh adaptivity and inexact algebraic eigenvalue solvers, it is of particular interest to bound the difference between the exact eigenvalue of the original problem at homotopy step $t = 1$ and the inexact iterative solution for a homotopy step $t \le 1$. Since the exact solution is unknown, this bound should be only based on the computed inexact approximations of right and left eigenvectors and the approximated eigenvalue of $\mathcal{H}_\ell(t)$ .

Using the a posteriori error bound for the discretization error from [HR01, GC09], we obtain that for any $0 \le t \le 1$,

$$\begin{aligned} &|\!|\!| u(t)-u_\ell(t) |\!|\!|^2 + |\!|\!| u^\star(t) - u_\ell^\star(t) |\!|\!|^2 + |\lambda(t) - \lambda_\ell(t)| \\ &\qquad \lesssim \eta^2(\lambda_\ell(t),u_\ell(t),u_\ell^\star(t)) := \sum_{T\in\mathcal{T}_\ell} \left(\eta^2(\lambda_\ell(t),u_\ell(t);T) + \eta^{\star 2}(\lambda_\ell(t),u_\ell^\star(t);T)\right). \end{aligned}$$

Here and throughout this paper,

$$\begin{aligned} \eta^2(\lambda_\ell(t),u_\ell(t);T) &:= h_T^2 \|\beta\cdot\nabla u_\ell(t) - \lambda_\ell(t) u_\ell(t)\|_{L^2(T)}^2 \\ &\quad + \sum_{E\in\mathcal{E}_\ell(T)} h_E \|[\nabla u_\ell(t)]\cdot n_E\|_{L^2(E)}^2, \\ \eta^{\star 2}(\lambda_\ell(t),u_\ell^\star(t);T) &:= h_T^2 \|-\beta\cdot\nabla \overline{u_\ell^\star(t)} - \lambda_\ell(t)\overline{u_\ell^\star(t)}\|_{L^2(T)}^2 \\ &\quad + \sum_{E\in\mathcal{E}_\ell(T)} h_E \|[\nabla \overline{u_\ell^\star(t)}]\cdot n_E\|_{L^2(E)}^2. \end{aligned}$$

Following [Par98, HL06, MM10], for the algebraic errors we have the estimate

$$\begin{aligned}&|||u_\ell(t)-\tilde{u}_\ell(t)|||^2 + |||u_\ell^\star(t)-\tilde{u}_\ell^\star(t)|||^2 + |\lambda_\ell(t)-\tilde{\lambda}_\ell(t)|\\ &\qquad\lesssim \mu^2(\tilde{\lambda}_\ell(t),\tilde{u}_\ell(t),\tilde{u}_\ell^\star(t)) := \left(\frac{\|\mathbf{r}_\ell\|_{B_\ell^{-1}}}{\|\mathbf{u}_\ell\|_{B_\ell}}\right)^2 + \left(\frac{\|\mathbf{r}_\ell^\star\|_{B_\ell^{-1}}}{\|\mathbf{u}_\ell^\star\|_{B_\ell}}\right)^2,\end{aligned}$$

with the algebraic residuals

$$\mathbf{r}_\ell := (A_\ell + C_\ell)\mathbf{u}_\ell - \lambda_\ell B_\ell \mathbf{u}_\ell, \quad \mathbf{r}_\ell^\star := \mathbf{u}_\ell^\star(A_\ell + C_\ell) - \lambda_\ell^\star \mathbf{u}_\ell^\star B_\ell,$$

and $\|\mathbf{u}_\ell\|_M := \sqrt{\mathbf{u}_\ell^* M \mathbf{u}_\ell}$. The constants for the algebraic error estimators depend on the condition number of the considered eigenvalue and the gap in the spectrum. However, in our numerical examples the eigenvalue of interest is well-conditioned and well-separated from the rest part of the spectrum.

**Lemma 5.1.** *Suppose that* $|\lambda_\ell(t)-\tilde{\lambda}_\ell(t)| < 1$. *Then, for a fixed* $0 \le t \le 1$, *the perturbation of the a posteriori error estimator for the discretization error satisfies*

$$|\eta(\lambda_\ell(t),u_\ell(t),u_\ell^\star(t)) - \eta(\tilde{\lambda}_\ell(t),\tilde{u}_\ell(t),\tilde{u}_\ell^\star(t))|^2 \lesssim \mu^2(\tilde{\lambda}_\ell(t),\tilde{u}_\ell(t),\tilde{u}_\ell^\star(t)).$$

*Proof.* Using the triangle inequality, we have

$$\begin{aligned}&|\eta(\lambda_\ell(t),u_\ell(t),u_\ell^\star(t)) - \eta(\tilde{\lambda}_\ell(t),\tilde{u}_\ell(t),\tilde{u}_\ell^\star(t))|^2\\ &\quad\le \sum_{T\in\mathcal{T}_\ell} h_T^2 \|\beta\cdot\nabla(u_\ell(t)-\tilde{u}_\ell(t)) - \lambda_\ell(t)u_\ell(t) + \tilde{\lambda}_\ell(t)\tilde{u}_\ell(t)\|_{L^2(T)}^2\\ &\quad+ \sum_{E\in\mathcal{E}_\ell} h_E \|[\nabla(u_\ell(t)-\tilde{u}_\ell(t))]\cdot n_E\|_{L^2(E)}^2\\ &\quad+ \sum_{T\in\mathcal{T}_\ell} h_T^2 \|-\beta\cdot\nabla(\overline{u_\ell^\star(t)}-\overline{\tilde{u}_\ell^\star(t)}) - \lambda_\ell(t)\overline{u_\ell^\star(t)} + \tilde{\lambda}_\ell(t)\overline{\tilde{u}_\ell^\star(t)}\|_{L^2(T)}^2\\ &\quad+ \sum_{E\in\mathcal{E}_\ell} h_E \|[\nabla(\overline{u_\ell^\star(t)}-\overline{\tilde{u}_\ell^\star(t)})]\cdot n_E\|_{L^2(E)}^2.\end{aligned}$$

The local discrete inverse inequality [BS02] for $v_\ell \in V_\ell$ reads

$$h_T^2 \|D^2 v_\ell\|_{L^2(T)}^2 \lesssim \|\nabla v_\ell\|_{L^2(T)}^2.$$

Let $\omega_E := T_+ \cup T_-$ denote the edge patch for two neighboring triangles $T_\pm \in \mathcal{T}_\ell$ such that $E = T_+ \cap T_-$. The trace inequality [BS02] for $v \in V$

$$\|v\|_{L^2(E)}^2 \lesssim h_E^{-1}\|v\|_{L^2(\omega_E)}^2 + h_E\|\nabla v\|_{L^2(\omega_E)}^2$$

together with another application of the triangle inequality yields

$$\begin{aligned}&|\eta(\lambda_\ell(t),u_\ell(t),u_\ell^\star(t)) - \eta(\tilde{\lambda}_\ell(t),\tilde{u}_\ell(t),\tilde{u}_\ell^\star(t))|^2\\ &\quad\lesssim \sum_{T\in\mathcal{T}_\ell} h_T^2 \|\lambda_\ell(t)u_\ell(t) - \tilde{\lambda}_\ell(t)\tilde{u}_\ell(t)\|_{L^2(T)}^2 + h_T^2\|\lambda_\ell^\star(t)u_\ell^\star(t) - \tilde{\lambda}_\ell^\star(t)\tilde{u}_\ell^\star(t)\|_{L^2(T)}^2\\ &\quad+ \sum_{T\in\mathcal{T}_\ell} h_T^2 |\beta|_\infty \left(\|\nabla u_\ell(t) - \nabla\tilde{u}_\ell(t)\|_{L^2(T)}^2 + \|\nabla u_\ell^\star(t) - \nabla\tilde{u}_\ell^\star(t)\|_{L^2(T)}^2\right)\\ &\quad+ \sum_{E\in\mathcal{E}_\ell} \|\nabla u_\ell(t) - \nabla\tilde{u}_\ell(t)\|_{L^2(\omega_E)}^2 + \|\nabla u_\ell^\star(t) - \nabla\tilde{u}_\ell^\star(t)\|_{L^2(\omega_E)}^2.\end{aligned}$$

The finite overlap of the edge patches $\omega_E$ and the Poincaré inequality [BS02] lead to

$$\begin{aligned}
&|\eta(\lambda_\ell(t), u_\ell(t), u_\ell^\star(t)) - \eta(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t))|^2 \\
&\quad\lesssim |\!|\!| u_\ell(t) - \tilde u_\ell(t) |\!|\!|^2 + |\!|\!| u_\ell^\star(t) - \tilde u_\ell^\star(t) |\!|\!|^2 \\
&\qquad + ||\lambda_\ell(t) u_\ell(t) - \tilde\lambda_\ell(t)\tilde u_\ell(t)||^2 + ||\lambda_\ell^\star(t) u_\ell^\star(t) - \tilde\lambda_\ell^\star(t)\tilde u_\ell^\star(t)||^2 \\
&\quad\lesssim |\!|\!| u_\ell(t) - \tilde u_\ell(t) |\!|\!|^2 + |\!|\!| u_\ell^\star(t) - \tilde u_\ell^\star(t) |\!|\!|^2 + |\lambda_\ell(t) - \tilde\lambda_\ell(t)|^2.
\end{aligned}$$

Using the assumption $|\lambda_\ell(t) - \tilde\lambda_\ell(t)| < 1$ completes the proof. □

**Lemma 5.2.** *For the model problem (1.1), the difference between the iterative eigenvalue $\tilde\lambda_\ell(t)$ in the homotopy $\mathcal{H}_\ell(t)$ and the continuous eigenvalue $\lambda(1)$ of the original problem* (1.1) *can be estimated a posteriori via*

$$|\lambda(1) - \tilde\lambda_\ell(t)| \lesssim \nu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) + \eta^2(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) + \mu^2(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t))$$

*in terms of*

$$\begin{aligned}
\nu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) :=& (1-t)|\beta|_\infty \left( |\!|\!| \tilde u_\ell(t) |\!|\!| + |\!|\!| \tilde u_\ell^\star(t) |\!|\!| \right) \\
&+ (1-t)|\beta|_\infty \left( \eta(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) + \mu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) \right).
\end{aligned}$$

*Proof.* The triangle inequality gives

$$|\lambda(1) - \tilde\lambda_\ell(t)| \le |\lambda(1) - \lambda(t)| + |\lambda(t) - \lambda_\ell(t)| + |\lambda_\ell(t) - \tilde\lambda_\ell(t)|.$$

The first term is estimated via Lemma 4.1 as

$$\begin{aligned}
|\lambda(1) - \lambda(t)| &\lesssim (1-t)|\beta|_\infty \left( |\!|\!| u(t) |\!|\!| + |\!|\!| u^\star(t) |\!|\!| \right) \\
&\le (1-t)|\beta|_\infty \left( |\!|\!| \tilde u_\ell(t) |\!|\!| + |\!|\!| \tilde u_\ell^\star(t) |\!|\!| \right) \\
&\quad + (1-t)|\beta|_\infty \left( |\!|\!| u(t) - u_\ell(t) |\!|\!| + |\!|\!| u_\ell(t) - \tilde u_\ell(t) |\!|\!| \right) \\
&\quad + (1-t)|\beta|_\infty \left( |\!|\!| u^\star(t) - u_\ell^\star(t) |\!|\!| + |\!|\!| u_\ell^\star(t) - \tilde u_\ell^\star(t) |\!|\!| \right).
\end{aligned}$$

The a posteriori error bound and Lemma 5.1 lead to

$$\begin{aligned}
|\!|\!| u(t) - u_\ell(t) |\!|\!| &\lesssim \eta(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) + \mu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)), \\
|\!|\!| u^\star(t) - u_\ell^\star(t) |\!|\!| &\lesssim \eta(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) + \mu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)).
\end{aligned}$$

The algebraic error estimates

$$\begin{aligned}
|\!|\!| u_\ell(t) - \tilde u_\ell(t) |\!|\!| &\lesssim \mu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)), \\
|\!|\!| u_\ell^\star(t) - \tilde u_\ell^\star(t) |\!|\!| &\lesssim \mu(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t))
\end{aligned}$$

then complete the estimate of the first term. The second term is estimated with Lemma 5.1 as

$$|\lambda(t) - \lambda_\ell(t)| \lesssim \eta^2(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)) + \mu^2(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t))$$

and the third term again with the algebraic error estimate

$$|\lambda_\ell(t) - \tilde\lambda_\ell(t)| \lesssim \mu^2(\tilde\lambda_\ell(t), \tilde u_\ell(t), \tilde u_\ell^\star(t)).$$

□

## 6. Algorithms

In this section we will combine the homotopy method with the adaptive finite element method. We will balance the homotopy error, the discretization error and the error in the iterative solution of the generalized algebraic eigenvalue problems (3.3)–(3.4). An important factor in the presented algorithms is the step-size control for the homotopy steps. The homotopy method strongly depends on the step-size $\tau$, which influences the convergence and accuracy of the method. A very small $\tau$ will assure that the homotopy method gives a good approximation of the desired

eigenvalues and eigenvectors, but will lead to large computational costs. On the other hand, if $\tau$ is too large, then the method may not capture a crossing or joining of eigenvalues. Therefore, the goal is to choose $\tau$ in an optimal way, such that it will assure accuracy of the approximation, minimize the computational effort and keep track of the eigenpath. To achieve this, we may employ adaptive step-size control techniques that are well established in the numerical solution of ordinary differential equations [HNW93], e.g., predictor-corrector procedures as they are commonly used [LZ99]. Combining the homotopy approach with the adaptive finite element method, however, requires a modification of the adaptive step-size control techniques.

At this stage, under the given assumptions, the following simple step-size control can be applied. If the number of required refinement steps for the homotopy parameters $t_i$ and $t_i + \tau$ differs significantly, then the homotopy step for $t_i + \tau$ is rejected and $t_i + q\tau$ is used, where $0 < q < 1$, e.g., $q = \frac{1}{2}$. If the number of refinements is small, then the step-size $\tau$ is preserved or even increased by choosing, e.g., $\tau = q^{-1}\tau$. This simple idea allows to describe the dependence of the step-size not only on the solution but also on the mesh adaptation process.

As test case, we consider only the eigenvalue with smallest real part, which is known to be simple and well-separated [Eva00] for all $0 \leq t \leq 1$. Thus it will not bifurcate and the evolution of the eigenvalue follows an analytic path.

The best choice of the maximal number of refinement steps is still an open question. Future work will have to include the combination of the presented concepts with methods that detect multiple eigenvalues, bifurcation in the paths, ill-conditioning, or the treatments of jumps in the eigenpaths.

In the following we present three different adaptive algorithms for the homotopy driven eigenvalue problem.

In Algorithm 1, a fixed step-size $\tau$ for the homotopy is considered in order to analyze the influence of the homotopy error on the mesh adaptation process and the accuracy of the solution. Algorithm 2 considers an adaptive step-size control for the homotopy, based on the number of refinements required to balance the discretization error $\eta_\ell$ and the desired accuracy $\varepsilon$. Algorithm 3 then finally combines the two concepts from Algorithms 1 and 2.

In all three algorithms, $\rho$ will denote the accuracy for the matrix eigensolver, $0 < \omega < 1$ is the parameter in the relative accuracy condition for the algebraic approximation error, $0 < \delta < 1$ is the parameter balancing the discretization and homotopy error estimators, $0 < \theta < 1$ is the marking parameter for the bulk marking strategy and $\gamma$ denotes the maximal number of refinement steps in each homotopy step of Algorithms 2 and 3. In Algorithm 1, $\tau$ is the fixed step-size, while in the other two algorithms it is the starting step-size for refinement.

In all three homotopy methods, the basic mesh adaptation method given by the procedures `Estimate & Solve`, `Mark` and `Refine` as described in Section 2 is used. In the `Estimate & Solve` function (see below) for the given mesh and parameters in each refinement step, the generalized algebraic eigenvalue problem (AEVP) for $((A_\ell + tC_\ell), B_\ell)$ has to be solved. The approximation of the eigenpair is considered to be accurate if the estimate for the complete algebraic approximation error $\mu$, (both for the left and right eigenvectors), is smaller then the discretization error $\eta$, up to some fixed constant $\omega$ (see line 5). To ensure that the algebraic approximation error itself is small, the tolerance parameter $\rho$ for the iterative solver depends on the discretization error $\eta$ and is also adapted (lines 4–6). The algebraic eigenvalue problem is solved using the ARPACK [LSY98] implementation of the implicitly restarted Arnoldi method for nonsymmetric eigenvalue problems. The size of the constructed Krylov subspaces is chosen to be as small as possible and

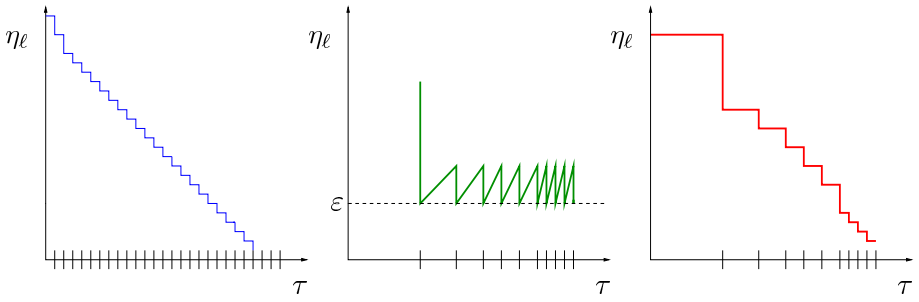


FIGURE 6.1. Schematic view of three homotopy-based Algorithms.

the approximations of the right and left eigenvectors from the previous iteration are taken as starting values for the new Arnoldi step. Note that here the final accuracy $\varepsilon$ of the solution is not required at every step, only the relation between the discretization error and the algebraic approximation error is used to stop the procedure.

In order to illustrate the differences between the three algorithms their main ideas are depicted in Figure 6.1.

6.1. **Algorithm 1.** The first algorithm introduces a homotopy method with fixed step-size $\tau$. For the initial homotopy parameter $t_0 = 0$, the corresponding Laplace eigenvalue problem is solved on the initial mesh $\mathcal{T}_0(t_0)$ (line 3), where the algebraic eigenvalue problem is solved up to tolerance $\rho(t_0)$ (line 4, see `Estimate & Solve` for details). This step is the same for all three algorithms. Based on the calculated initial approximation of the eigenpair at $t_0$ the corresponding discretization and homotopy error estimators $\eta_0(t_0)$, $\nu_0(t_0)$ are determined (line 6).

In order to balance the discretization error, the homotopy error, and the desired accuracy $\varepsilon$, the adaptive mesh refinement method is used (lines 9–15). The mesh adaptation process is repeated as long as the discretization error dominates over the homotopy error multiplied by a balancing factor $\delta$ or is larger than the desired accuracy $\varepsilon$ (line 9). Throughout the adaptive loop, sequences of meshes $\mathcal{T}_{\ell+j}(t_k)$, error estimators $\eta_j(t_k), \nu_j(t_k), \mu_j(t_k)$ and eigentriple approximations $(\tilde{\lambda}_j(t_k), \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k))$ are assembled. To avoid unnecessary computational work, at each step of the adaptation loop, the algebraic eigenvalue problem is solved only up to the

---

`Estimate & Solve`

**Input:** $\mathcal{T}$, $t$, $\rho$, $\omega$, $\tilde{\mathbf{u}}, \tilde{\mathbf{u}}^\star$

1: $[(A+C), B]$ = Create AEVP($\mathcal{T}, \beta, t$)
2: $[\mu, \tilde{\mathbf{u}}, \tilde{\mathbf{u}}^\star]$ = Solve AEVP($A+C$, $B$, $\rho$, $\tilde{\mathbf{u}}$, $\tilde{\mathbf{u}}^\star$)
3: Compute $\eta$
4: $\rho = 2\eta$
5: **while** $\mu > \omega\eta$ **do**
6: $\quad \rho = \frac{\rho}{2}$
7: $\quad [\mu, \tilde{\mathbf{u}}, \tilde{\mathbf{u}}^\star]$ = Solve AEVP($A+C$, $B$, $\rho$, $\tilde{\mathbf{u}}$, $\tilde{\mathbf{u}}^\star$)
8: $\quad$ Compute $\eta$
9: **end while**
10: Compute $\nu$

**Output:** $\eta$, $\nu$, $\mu$, $\lambda$, $\tilde{\mathbf{u}}, \tilde{\mathbf{u}}^\star$

---

accuracy $\rho(t_k)$, which depends on the discretization error $\eta_j(t_k)$ (see line 8 and the `Estimate & Solve` function for details). When the condition in line 9 does not hold, a new homotopy parameter $t_{k+1} = t_k + \tau$ is chosen and the new adaptation process starts with a previously obtained approximation taken as initial guess (line 13). Here $P_{j,j-1}$ denotes the prolongation matrix from the last coarse mesh $\mathcal{T}_{j-1}(t_k)$ to the refined mesh $\mathcal{T}_j(t_k)$ (line 12–13). Note that the final mesh derived for the former homotopy parameter is taken as the initial mesh for the new computations (line 3). After a fixed number of homotopy steps $t_k$ reaches its final value one and the algorithm returns the approximated eigenvalues and eigenvectors of the model problem.

The final number of refinement levels reached up to the parameter $t_k$ is denoted by $\ell$, while $j$ is a refinement index for the current parameter $t_k$. This distinction is made to separate a sequence of meshes for a single homotopy step from the final sequence obtained for the whole algorithm. It has particular importance for the next two algorithms.

Although controlling the homotopy error is beneficial, however, an arbitrary fixed choice of the homotopy step-size, in general, will not work, especially for more complicated problems. In the nonsymmetric case the eigenvalues move according to their condition number [Saa92]. Ill-conditioned eigenvalues, as a function of $t$, may move very fast. The lack of an analogue of the min-max theorem [GV96] for nonsymmetric problems makes the localization of the eigenvalue very hard. In particular, it may be difficult to guarantee fast convergence of the iterative eigensolver to the eigenvalue with smallest real part for the next homotopy parameter $t_k + \tau$ even with correct starting eigenvalue for a certain parameter $t_k$, if the step-size $\tau$ is chosen too large. On the other hand choosing $\tau$ very small leads to a large number of homotopy steps, and since for each step the whole adaptive mesh refinement loop has to be performed, this may led to large computational effort.

6.2. **Algorithm 2.** In contrast to Algorithm 1, in Algorithm 2 an adaptive step-size control for the homotopy is used. Starting with an initial step-size $\tau$ the first

**Algorithm 1**

**Input:** $t_0 = 0, \tau, \mathcal{T}_0(t_0), \rho, \varepsilon, \omega, \delta$, $\tilde{\mathbf{u}}_0(t_0)$, $\tilde{\mathbf{u}}_0^\star(t_0)$

1: $\ell = 1,\ k = 0$
2: **while** $t_k \leq 1$ **do**
3: $\quad \mathcal{T}_0(t_k) = \mathcal{T}_\ell(t_{k-1})$
4: $\quad \rho(t_k) = \rho$
5: $\quad [\tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k)] = [\tilde{\mathbf{u}}_\ell(t_{k-1}), \tilde{\mathbf{u}}_\ell^\star(t_{k-1})]$
6: $\quad [\eta_0(t_k), \nu_0(t_k), \tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k)] =$ `Estimate & Solve`$(\mathcal{T}_0(t_k), \rho(t_k), \omega, \tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k))$
7: $\quad j = 0$
8: $\quad \rho(t_k) = \eta_\ell(t_k)$
9: $\quad$ **while** $\eta_j(t_k) > \max(\delta\nu_j(t), \varepsilon)$ **do**
10: $\quad\quad j = j + 1$
11: $\quad\quad \mathcal{M}_j(t_k) =$ `Mark`$(\eta_j(t_k), \theta)$
12: $\quad\quad \mathcal{T}_j(t_k) =$ `Refine`$(\mathcal{T}_{j-1}(t_k), \mathcal{M}_j(t_k))$
13: $\quad\quad [\tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k)] = [P_{j,j-1}\tilde{\mathbf{u}}_{j-1}(t_k), P_{j,j-1}\tilde{\mathbf{u}}_{j-1}^\star(t_k)]$
14: $\quad\quad [\eta_j(t_k), \nu_j(t_k), \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k)] =$ `Estimate & Solve`$(\mathcal{T}_j(t_k), \rho(t_k), \omega, \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k))$
15: $\quad$ **end while**
16: $\quad \ell = \ell + j$
17: $\quad t_{k+1} = t_k + \tau,\ k = k + 1$
18: **end while**

**Output:** $\tilde{\lambda}(1), \tilde{\mathbf{u}}(1), \tilde{\mathbf{u}}^\star(1)$

**Algorithm 2**

**Input:** $t_0 = 0, \tau, \beta, \mathcal{T}_0(t_0), \rho, \varepsilon, \omega, \gamma, \tilde{\mathbf{u}}_0(t_0), \tilde{\mathbf{u}}_0^\star(t_0)$

1: $\ell = 1,\ k = 0$
2: **while** $t_k < 1$ **do**
3: $\quad \mathcal{T}_0(t_k) = \mathcal{T}_{\ell-1}(t_{k-1})$
4: $\quad \rho(t_k) = \rho$
5: $\quad [\tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k)] = [\tilde{\mathbf{u}}_{\ell-1}(t_{k-1}), \tilde{\mathbf{u}}_{\ell-1}^\star(t_{k-1})]$
6: $\quad [\eta_0(t_k), \nu_0(t_k), \tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k)] = \texttt{Estimate \& Solve}(\mathcal{T}_0(t_k), \rho(t_k), \omega, \tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k))$
7: $\quad \rho(t_k) = \eta_\ell(t_k)$
8: $\quad j = 0$
9: $\quad$ **while** $\eta_j(t_k) > \varepsilon$ **do**
10: $\quad\quad$ **if** $j > \gamma$ **then**
11: $\quad\quad\quad k = k - 1$
12: $\quad\quad\quad \tau = q\tau$
13: $\quad\quad\quad j = 0$
14: $\quad\quad\quad$ **break**
15: $\quad\quad$ **end if**
16: $\quad\quad j = j + 1$
17: $\quad\quad \mathcal{M}_j(t_k) = \texttt{Mark}(\eta_j(t_k), \theta)$
18: $\quad\quad \mathcal{T}_j(t_k) = \texttt{Refine}(\mathcal{T}_{j-1}(t_k), \mathcal{M}_j(t_k))$
19: $\quad\quad [\tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k)] = [P_{j,j-1}\tilde{\mathbf{u}}_{j-1}(t_k), P_{j,j-1}\tilde{\mathbf{u}}_{j-1}^\star(t_k)]$
20: $\quad\quad [\eta_j(t_k), \nu_j(t_k), \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k)] = \texttt{Estimate \& Solve}(\mathcal{T}_j(t_k), \rho(t_k), \omega, \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k))$
21: $\quad$ **end while**
22: $\quad \ell = \ell + j$
23: $\quad$ **if** $j < \gamma$ **then**
24: $\quad\quad \tau = q^{-1}\tau$
25: $\quad$ **end if**
26: $\quad t_{k+1} = \min(t_k + \tau, 1),\ k = k + 1$
27: **end while**

**Output:** $\tilde{\lambda}_\ell(1), \tilde{\mathbf{u}}_\ell(1), \tilde{\mathbf{u}}^\star(1)$

approximation is computed to assure that the discretization error $\eta_j(t_k)$ is smaller than the fixed, desired accuracy $\varepsilon$ (line 9). No dependence on the homotopy error is considered here. Additionally, for each homotopy parameter only a fixed number of refinement steps $\gamma$ inside the adaptive loop is allowed (see line 10). If the adaptive loop needs more refinement steps than $\gamma$ (line 10), then it means that the eigenvalue problems considered for parameters $t_k$ and $t_k + \tau$ differ too much and that the step-size $\tau$ should be decreased. In that case, to ensure good approximations in the eigenvalue continuation, the algorithm rejects the current homotopy step (lines 11–13), sets up a new $\tau = q\tau$ (line 12), for some $0 < q < 1$, and starts the adaptation loop for the new homotopy parameter $t_k + \tau$. If the number of refinements is smaller than $\gamma$, then the algorithm attempts to increase the step-size to $q^{-1}\tau$ (line 24). Otherwise $\tau$ is preserved in the next homotopy step. At this point the previously introduced distinction between global and local refinement indices $\ell$ and $j$ is used to carry out the rejection step, while keeping the right mesh hierarchy. Meshes obtained for the rejected homotopy parameter will not be considered in the final sequence of meshes.

Note, that here the initial mesh for the new homotopy parameter is taken as the last but one mesh obtained for the previous homotopy step (line 3). If the step-sizes were chosen optimally and the consecutive problems do not differ too much, then the previous mesh should be a good starting mesh for the next step. In this way

**Algorithm 3**

**Input:** $t_0 = 0, \tau, q, \mathcal{T}_0(t_0), \varepsilon, \omega, \delta, \gamma, \tilde{\mathbf{u}}_0(t_0), \tilde{\mathbf{u}}_0^\star(t_0)$

$\ell = 1, k = 0$
**while** $t_k \leq 1$ & $t_{k-1} < 1$ **do**
  $\mathcal{T}_0(t_k) = \mathcal{T}_{\ell-1}(t_{k-1})$
  $\rho(t_k) = \rho$
  $[\tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k)] = [\tilde{\mathbf{u}}_{\ell-1}(t_{k-1}), \tilde{\mathbf{u}}_{\ell-1}^\star(t_{k-1})]$
  $[\eta_0(t_k), \nu_0(t_k), \tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k)] = \texttt{Estimate \& Solve}(\mathcal{T}_0(t_k), \rho(t_k), \omega, \tilde{\mathbf{u}}_0(t_k), \tilde{\mathbf{u}}_0^\star(t_k))$
  $\rho(t_k) = \eta_\ell(t_k)$
  $j = 0$
  **while** $\eta_j(t_k) > \max(\delta\nu_j(t_k), \varepsilon)$ **do**
    **if** $j > \gamma$ **then**
      $k = k - 1$
      $\tau = q\tau$
      $j = 0$
      **break**
    **end if**
    $j = j + 1$
    $\mathcal{M}_j(t_k) = \texttt{Mark}(\eta_j(t_k), \theta)$
    $\mathcal{T}_j(t_k) = \texttt{Refine}(\mathcal{T}_{j-1}(t_k), \mathcal{M}_j(t_k))$
    $[\tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k)] = [P_{j,j-1}\tilde{\mathbf{u}}_{j-1}(t_k), P_{j,j-1}\tilde{\mathbf{u}}_{j-1}^\star(t_k)]$
    $[\eta_j(t_k), \nu_j(t_k), \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k)] = \texttt{Estimate \& Solve}(\mathcal{T}_j(t_k), \rho(t_k), \omega, \tilde{\mathbf{u}}_j(t_k), \tilde{\mathbf{u}}_j^\star(t_k))$
  **end while**
  $\ell = \ell + j$
  **if** $j < \gamma$ **then**
    $\tau = q^{-1}\tau$
  **end if**
  $t_{k+1} = \min(t_k + \tau, 1), k = k + 1$
**end while**

**Output:** $\tilde{\lambda}_\ell(1), \tilde{\mathbf{u}}_\ell(1), \tilde{\mathbf{u}}^\star(1)$

the continuation of meshes is also guaranteed. At the beginning it is reasonable to allow $\tau$ to be large and let the algorithm to adapt its step-size by itself. It is obvious, however, that if the total error is dominated by the homotopy error $\nu_\ell(t_k)$, then driving the discretization error $\eta_\ell(t_k)$ in each homotopy step below $\varepsilon$ may lead to large computational effort.

6.3. **Algorithm 3.** The third algorithm combines both ideas of controlling the homotopy error and using adaptive step-size control. In this way the homotopy method accepts only the approximations which are of a desired accuracy and whose computational cost is reasonable. Simultaneously, adaptation in space, homotopy and for the iterative solver is applied. During the mesh adaptation the discretization error $\eta_j(t_k)$ is adapted to be smaller than the homotopy error $\nu_j(t_k)$. Also at each iteration step of the algebraic eigensolver, the approximation error $\mu_j(t_k)$ is adjusted, to avoid computing a solution that is too accurate in comparison to the discretization error $\eta_j(t_k)$. The adaptation of the homotopy parameter $t$ is based on the maximal number of refinement levels $\gamma$. Currently, no analysis of the optimal choice of $\gamma$ is known, that will lead to the minimal number of refinement steps.

In summary, for Algorithm 1 fixed step-sizes in $t$ are considered together with adaptivity in the mesh size assuring that the complete discretization error $\eta$ is below the homotopy error $\nu$ for each homotopy parameter $t$. In Algorithm 2,

adaptivity in both the homotopy parameter $t$ and the mesh is achieved. Here, however, unlike in Algorithm 1 the discretization error is driven below the fixed tolerance $\varepsilon$, which is the same for each homotopy parameter and adaptation level. Algorithm 3 then combines the techniques of Algorithms 1 and 2. The homotopy error $\nu$ drives the mesh adaptivity and the homotopy step-sizes are adapted with respect to the parameter $\gamma$.

## 7. Numerical Experiments

This section presents some numerical results obtained with the three adaptive homotopy Algorithms 1–3 presented in Section 6. As a model problem we consider

$$-\Delta u + \beta \cdot \nabla u = \lambda u \quad \text{in} \quad \Omega \qquad \text{and} \qquad u = 0 \quad \text{on} \quad \partial\Omega$$

with $\Omega$ being either the unit square or the L-shaped domain. In order to calculate the eigenvalue errors we also compute some reference values. The reference values were obtained by Aitken extrapolation on uniform meshes [Ait26].

Common to all experiments is that for ARPACK [LSY98] the number $k$ of Arnoldi vectors equals 3 and the maximal number `MXITER` of Arnoldi restarts is set to 1 [LSY98]. The experiments were run on a AMD Phenom II X6 2,8 GHz processor with 8GB RAM using the programming environment MATLAB R2010a [MAT10].

The homotopy starts with the simple symmetric eigenvalue problem with known smallest eigenvalue $\lambda(t_0) = 2\pi^2$ for the unit square and known approximation $\lambda(t_0) \approx 9.6397238440219$ [TB06] for the L-shaped domain and then uses the homotopy to bring in the convection part. All experiments determine the eigenvalue with the smallest real part, since it is known to be simple and well-separated for any value of convection parameter $\beta$ [Eva00], thus there are no bifurcation points and the algorithms are following analytic eigenpaths.

To recall the motivation of the homotopy method, it is important to note that for general non-selfadjoint problems, there is no guarantee that we achieve convergence to an eigenvalue of interest if standard methods are used. Experiments show that with a small number of Arnoldi vectors (i.e., a low dimensional Krylov subspace,) and a random starting vector ARPACK does not find any good approximation to an eigenvalue for $t = 1$ even for very coarse meshes. Thus, stable adaptive mesh refinement is not possible with a low cost variation of the Arnoldi method as shown for selfadjoint problems in [MM10]. On the other hand the numerical experiments show that, starting from the symmetric problem and following the eigenpath lead to sufficiently accurate approximations of the original non-selfadjoint problem. This shows that we can view our algorithms as means to provide a starting vector for the non-selfadjoint problem which is sufficiently close to the eigenvector of interest. Therefore, most of the computational work is expected to occur in the last homotopy step $t = 1$ which is confirmed by the numerical experiments.

**Example 1.** For this example let $\Omega$ be the (convex) unit square $\Omega = (0,1) \times (0,1)$. We choose the convection parameter $\beta = (20,0)^T$, the starting point of the homotopy $t_0 = 0$, the marking parameter $\theta = 0.3$, the balancing parameter of the discretization and approximation error estimators $\omega = 0.1$, the step-size update parameter $q = 1/2$, the number of refinement steps $\gamma = 2$, the overall accuracy $\varepsilon = 10^{-1}$, the initial tolerance for the iterative solver $\rho = 1$ and the balancing parameter of the homotopy and discretization error estimators $\delta = 0.1$. A reference value for the eigenvalue with the smallest real part is given by

$$\lambda \approx 119.7392.$$

In general one can observe that all three algorithms lead to a finite sequence of homotopy steps and to an approximation of the eigenvalue of interest at the last

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.0 | 18.7972 | 267.9989 | 0.0025677 | 286.7986 |
| 0.1 | 21.9037 | 250.3131 | 0.0003188 | 272.2171 |
| 0.2 | 17.6390 | 224.2302 | 0.0042579 | 241.8735 |
| 0.3 | 14.7243 | 204.8199 | 0.0066615 | 219.5508 |
| 0.4 | 12.0933 | 185.7716 | 0.0054502 | 197.8704 |
| 0.5 | 10.1746 | 167.8197 | 0.0560768 | 178.0503 |
| 0.6 | 7.8788 | 142.9867 | 0.0189887 | 150.8845 |
| 0.7 | 11.0907 | 121.0055 | 0.0577501 | 132.1540 |
| 0.8 | 8.4339 | 85.4466 | 0.0206147 | 93.9012 |
| 0.9 | 3.4934 | 44.0072 | 0.0025632 | 47.5031 |
| 1.0 | 0.0854 | 0.0000 | 0.0008344 | 0.0862 |

TABLE 7.1. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 1 for Example 1.

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.0 | 20.31171 | 0.83037 | 65 | 0.04 |
| 0.1 | 21.19837 | 0.82296 | 65 | 0.05 |
| 0.2 | 23.76193 | 0.80155 | 114 | 0.09 |
| 0.3 | 28.68327 | 0.76045 | 222 | 0.13 |
| 0.4 | 35.57882 | 0.70286 | 436 | 0.17 |
| 0.5 | 44.58901 | 0.62762 | 838 | 0.24 |
| 0.6 | 55.71845 | 0.53467 | 1607 | 0.35 |
| 0.7 | 68.87482 | 0.42479 | 1607 | 0.41 |
| 0.8 | 83.83805 | 0.29983 | 3075 | 0.66 |
| 0.9 | 100.83461 | 0.15788 | 10370 | 1.86 |
| 1.0 | 119.74434 | 0.00004 | 587509 | 127.34 |

TABLE 7.2. The eigenvalue approximation $\tilde{\lambda}_\ell(t)$, the relative eigenvalue error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 1 applied to Example 1.

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.00 | 0.0725 | 183.1140 | 0.0000000 | 183.1865 |
| 0.25 | 0.0649 | 156.7655 | 0.0000002 | 156.8303 |
| 0.50 | 0.0740 | 136.5043 | 0.0000012 | 136.5783 |
| 0.75 | 0.0640 | 88.4754 | 0.0000598 | 88.5395 |
| 1.00 | 0.0783 | 0.0000 | 0.0004680 | 0.0788 |

TABLE 7.3. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 2 applied to Example 1.

step $t = 1$. Notice that for all algorithms, more or less, most of the computational work is done at the last step and therefore for the final problem. This can be seen in Tables 7.2, 7.4 and 7.6 when comparing the CPU time after the last step to the

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.00 | 19.74139 | 0.83513 | 18420 | 2.62 |
| 0.25 | 25.98903 | 0.78295 | 48506 | 20.51 |
| 0.50 | 44.73837 | 0.62637 | 124817 | 40.28 |
| 0.75 | 75.98888 | 0.36538 | 366519 | 112.36 |
| 1.00 | 119.74216 | 0.00002 | 641569 | 278.09 |

TABLE 7.4. The eigenvalue approximation $\tilde{\lambda}_\ell(t)$, the relative eigenvalue error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 2 applied to Example 1.

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.0000 | 18.7972 | 267.9987 | 0.0025668 | 286.7984 |
| 0.2500 | 21.9560 | 224.1103 | 0.0070254 | 246.0733 |
| 0.5000 | 12.7398 | 173.0761 | 0.1539409 | 185.9698 |
| 0.7500 | 6.2305 | 99.7848 | 0.0008341 | 106.0161 |
| 0.8750 | 5.1172 | 54.7893 | 0.0003906 | 59.9069 |
| 0.9375 | 1.8715 | 27.6650 | 0.0001211 | 29.5367 |
| 0.9688 | 1.1430 | 14.0956 | 0.0271601 | 15.2658 |
| 0.9844 | 0.6630 | 7.0425 | 0.0141278 | 7.7196 |
| 0.9922 | 0.2189 | 3.4744 | 0.0006248 | 3.6940 |
| 1.0000 | 0.0745 | 0.0000 | 0.0020618 | 0.0765 |

TABLE 7.5. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ concerning Algorithm 3 applied to Example 1.

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.0000 | 20.31171 | 0.83037 | 65 | 0.04 |
| 0.2500 | 25.86284 | 0.78401 | 112 | 0.25 |
| 0.5000 | 44.52525 | 0.62815 | 661 | 0.45 |
| 0.7500 | 75.97150 | 0.36553 | 3613 | 0.88 |
| 0.8750 | 96.37374 | 0.19514 | 6538 | 5.20 |
| 0.9375 | 107.66847 | 0.10081 | 21936 | 22.60 |
| 0.9688 | 113.63394 | 0.05099 | 40027 | 53.26 |
| 0.9844 | 116.67842 | 0.02556 | 71610 | 194.81 |
| 0.9922 | 118.19399 | 0.01290 | 226196 | 358.30 |
| 1.0000 | 119.76367 | 0.00020 | 685571 | 587.75 |

TABLE 7.6. The eigenvalue approximation $\tilde{\lambda}_\ell(t)$, the relative error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 3 for Example 1.

previous one. Note that here we only present the data for the best approximation of each homotopy step and not those for the intermediate approximations.

In Algorithm 1 the fixed homotopy step-size $\tau = 0.1$ is chosen. Table 7.1 and 7.2 for Algorithm 1 show that a small homotopy step-size leads to a sequence where the second last homotopy step $t = 0.9$ does involve a small discrete problem, i.e.,

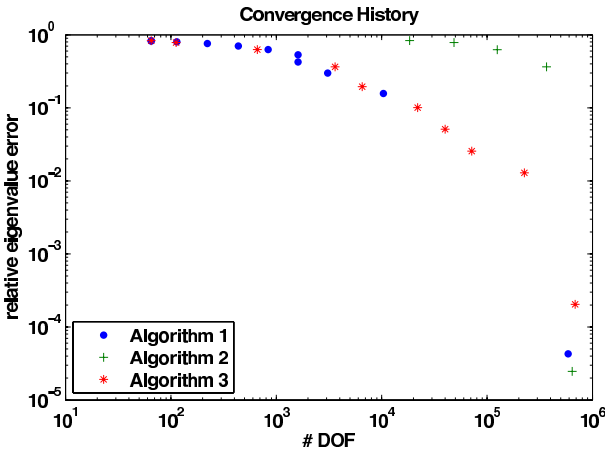


FIGURE 7.1. Convergence history of Algorithms 1, 2, and 3 with respect to #DOF for Example 1.

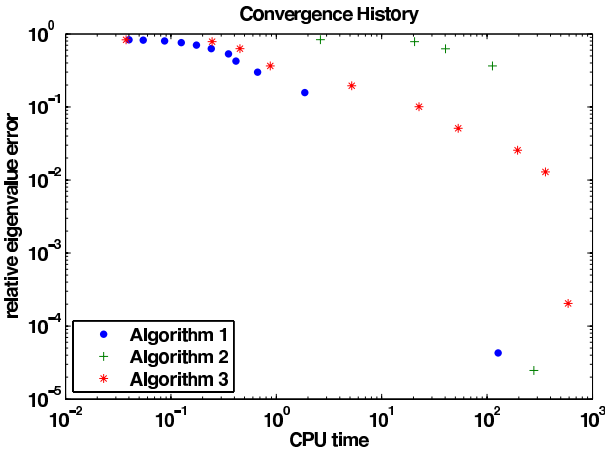


FIGURE 7.2. Convergence history of Algorithms 1, 2, and 3 with respect to CPU time for Example 1.

#DOF = 10370. Therefore, most of the refinement is done only in the last homotopy step $t = 1$, when the final accuracy is reached. Thus the computational overhead introduced by the homotopy is minor for the right choice of homotopy step-size $\tau$. Since the best choice for $\tau$ is not known, it is necessary, and in practice reasonable, to introduce some extra computational overhead by using adaptive step-size control. One may notice that the value obtained in the second last homotopy step has a large relative error and only the final approximation is good. As displayed in Figure 7.1 this effect leads to a nonlinear convergence rate and results in larger eigenvalue errors for $t < 1$ and accurate values only for $t = 1$.

Algorithm 2 introduces an adaptive homotopy step-size control. As initial step-size $\tau = 1$ is chosen. Tables 7.3 and 7.4 show that the first homotopy step is rejected and a smaller step-size $\tau$ is taken. In this example Algorithm 2 chooses less homotopy steps than the other two algorithms. Due to the fixed control of the discretization error by $\varepsilon$, the number of degrees of freedom (DOFs) is already high for the simple symmetric problem. This means that for $t < 1$ the error with

respect to the DOFs is much larger than for the other algorithms as displayed in Figure 7.1. On the other hand for the last step $t = 1$ the result is very accurate.

To overcome the drawback of a fixed step-size in Algorithm 1 and a fixed discretization error control in Algorithm 2, both techniques are combined in Algorithm 3. In Tables 7.5 and 7.6 we observe that the homotopy step-size is decreased very much towards the end of the homotopy process. This effect is due to the fact, that the algorithm increases the number of DOF strongly only for $t$ close to 1. This observation can be interpreted as that the algorithm computes a sufficiently accurate initial approximation to an eigenvector for $t = 1$. Note that most of the computational costs arise for $t$ close to 1, during the last three homotopy steps. Figure 7.1 shows that Algorithm 3 is a combination of the other two algorithms. The error for approximations with homotopy steps $t < 1$ is much smaller than for Algorithm 2 but similar to that of Algorithm 1. In contrast to Algorithm 1 the homotopy step-size is adapted, fewer homotopy steps needed and the steps are more concentrated towards $t = 1$.

In Figure 7.2 all three algorithms are compared with respect to computational time. Obviously, Algorithm 2 and 3 need more time than Algorithm 1, since they reject some steps during their automatic step-size control. For more complicated problems, going beyond this simple model example, it is expected that the adaptive step-size control will lead to faster computation than the method with a fixed step-size. The homotopy procedure in Algorithm 1 only introduces little computational overhead, with the possible drawback of a small (unknown) fixed step-size while Algorithm 2 does adapt the step-size automatically, but for the cost of larger computational overhead. In fact Table 7.4 shows that the overhead is less than $1/2$ of the overall CPU time, which is worthwhile. On the other hand Algorithm 3 needs even more computational time but combines the two advantages of Algorithm 1 and 2. The increase of the CPU time is due to the fact that Algorithm 3 rejects many steps during the homotopy process. Nevertheless, this moderate increase of the computational cost seems to be reasonable for more difficult situations, where without path following techniques no convergence to the desired eigenvalues can be guaranteed.

The final approximate primal and dual eigenfunctions for Algorithms 1, 2, and 3, together with the corresponding meshes, are depicted in Figures 7.3, 7.4 and 7.5. The final meshes for all problems look quite similar. Notice that, due to the adaptive refinement procedure for triangles, the symmetry of the mesh cannot strictly be preserved. For the square domain, primal and dual solutions of the problem have almost independent supports living on the opposite boundaries of the domain due to the convection in $x$ direction. Therefore, all final meshes look quite "symmetric". This observation shows that, in general, it is necessary to adapt the mesh for both the primal and dual eigenfunctions. Note that the meshes are more refined towards the strong boundary layers of both the primal and the dual solution.

**Example 2.** As in the first example, let $\Omega$ be the (convex) unit square $\Omega = (0,1) \times (0,1)$ and the convection parameter $\beta = (20, 0)^T$. We choose the starting point of the homotopy $t_0 = 0$, the marking parameter $\theta = 0.3$, the balancing parameter of the discretization and approximation error estimators $\omega = 0.1$, the step-size update parameter $q = 1/3$, the number of refinement steps $\gamma = 2$, the overall accuracy $\varepsilon = 10^{-1}$, the initial tolerance for the iterative solver $\rho = 1$ and the balancing parameter of the homotopy and discretization error estimators $\delta = 0.1$. Note that the only difference to Example 1 is the choice of the homotopy update parameter $q$. Here we demonstrate how a different choice of $q$ influences the homotopy process for algorithms 2 and 3. The results are presented in Tables 7.7, 7.8, 7.9 and 7.10. Figures 7.6 and 7.7 compare the results obtained for Examples

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.0000 | 0.0725 | 183.1140 | 0.0000000 | 183.1865 |
| 0.1111 | 0.0912 | 168.2191 | 0.0000003 | 168.3103 |
| 0.2222 | 0.0942 | 159.3306 | 0.0000002 | 159.4248 |
| 0.3333 | 0.0989 | 152.3163 | 0.0000008 | 152.4151 |
| 0.4444 | 0.0960 | 143.1067 | 0.0000018 | 143.2027 |
| 0.5556 | 0.0911 | 129.1835 | 0.0000177 | 129.2746 |
| 0.6667 | 0.0801 | 108.7764 | 0.0001128 | 108.8567 |
| 0.7778 | 0.0683 | 80.8181 | 0.0000674 | 80.8864 |
| 0.8889 | 0.0957 | 45.0185 | 0.0054764 | 45.1197 |
| 1.0000 | 0.0754 | 0.0000 | 0.0000176 | 0.0755 |

TABLE 7.7. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 2 applied to Example 2.

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.0000 | 19.74139 | 0.83513 | 18420 | 2.49 |
| 0.1111 | 20.97550 | 0.82482 | 18790 | 16.60 |
| 0.2222 | 24.67750 | 0.79391 | 29056 | 25.07 |
| 0.3333 | 30.84926 | 0.74236 | 45356 | 30.59 |
| 0.4444 | 39.49122 | 0.67019 | 79339 | 40.55 |
| 0.5556 | 50.60291 | 0.57739 | 125471 | 57.97 |
| 0.6667 | 64.18232 | 0.46398 | 229212 | 94.00 |
| 0.7778 | 80.23295 | 0.32994 | 373527 | 163.85 |
| 0.8889 | 98.74642 | 0.17532 | 374404 | 223.33 |
| 1.0000 | 119.74011 | 0.00001 | 664996 | 347.61 |

TABLE 7.8. The eigenvalue approximation $\tilde{\lambda}_\ell(t)$, the relative eigenvalue error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 2 applied to Example 2.

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.0000 | 16.8811 | 263.0051 | 0.0021032 | 279.8883 |
| 0.3333 | 18.0718 | 206.6701 | 0.0269778 | 224.7690 |
| 0.6667 | 12.4096 | 131.4125 | 0.0267075 | 143.8488 |
| 0.7778 | 5.3468 | 93.0901 | 0.5150339 | 98.9520 |
| 0.8889 | 4.1960 | 49.5161 | 0.1155408 | 53.8276 |
| 0.9259 | 2.5740 | 33.2055 | 0.0664040 | 35.8459 |
| 0.9630 | 1.6085 | 16.6848 | 0.0016130 | 18.2949 |
| 0.9753 | 0.9158 | 11.0943 | 0.0048967 | 12.0149 |
| 0.9877 | 0.5368 | 5.5432 | 0.0030163 | 6.0829 |
| 0.9918 | 0.3062 | 3.6710 | 0.0001411 | 3.9773 |
| 1.0000 | 0.0585 | 0.0000 | 0.0001896 | 0.0586 |

TABLE 7.9. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 3 applied to Example 2.

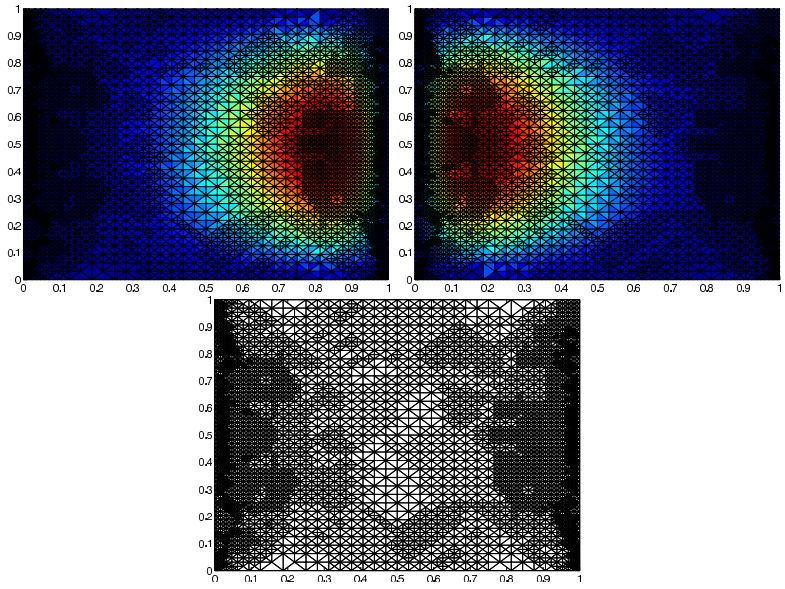

FIGURE 7.3. Primal (top left) and dual (top right) eigenfunction approximations for the final mesh (bottom) with 5130 nodes for Algorithm 1 applied to Example 1 with $\varepsilon = 2$.

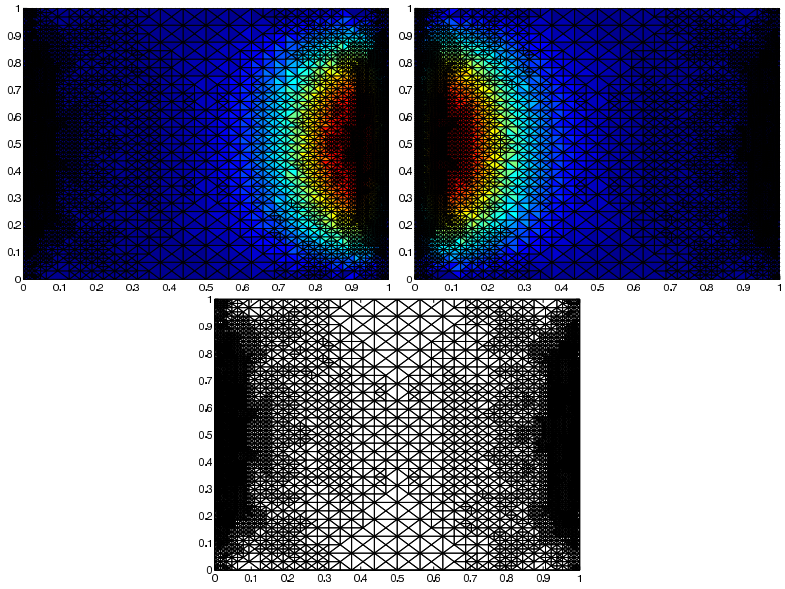

FIGURE 7.4. Primal (top left) and dual (top right) eigenfunction approximations for the final mesh (bottom) with 6225 nodes for Algorithm 2 applied to Example 1 with $\varepsilon = 10$.

1 and 2. In general we do not observe significant differences, which confirms that the presented algorithms seem to be rather robust with respect to the adaptivity of the homotopy. Comparing the results with those of the Example 1 shows that the choice $q = 1/3$ leads to similar relative eigenvalue errors for $t < 1$ but smaller relative eigenvalue error at the end for homotopy step $t = 1$. It is remarkable that Table 7.8 indicates that Algorithm 2 generates a sequence with almost fixed

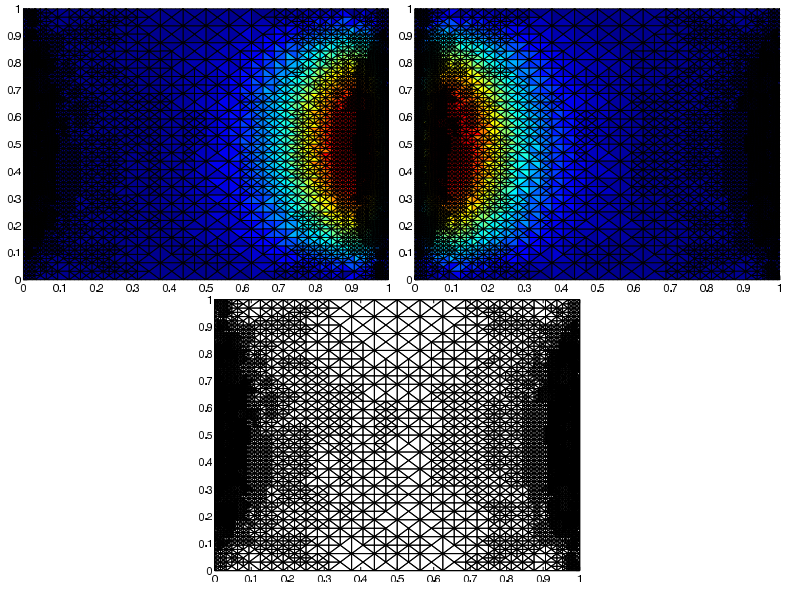

FIGURE 7.5. Primal (top left) and dual (top right) eigenfunction approximations for the final mesh (bottom) with 6663 nodes for Algorithm 3 applied to Example 1 with $\varepsilon = 10$.

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.0000 | 20.23079 | 0.83104 | 67 | 0.05 |
| 0.3333 | 30.72139 | 0.74343 | 211 | 0.29 |
| 0.6667 | 64.20818 | 0.46377 | 1283 | 0.46 |
| 0.7778 | 80.23738 | 0.32990 | 4610 | 1.90 |
| 0.8889 | 98.94516 | 0.17366 | 8390 | 2.47 |
| 0.9259 | 105.61009 | 0.11800 | 15539 | 20.97 |
| 0.9630 | 112.53208 | 0.06019 | 27839 | 38.42 |
| 0.9753 | 114.91115 | 0.04032 | 50910 | 92.94 |
| 0.9877 | 117.31628 | 0.02023 | 90675 | 148.05 |
| 0.9918 | 118.11498 | 0.01356 | 162166 | 340.33 |
| 1.0000 | 119.74169 | 0.00002 | 874628 | 510.75 |

TABLE 7.10. The approximation $\tilde{\lambda}_\ell(t)$, the relative error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 3 applied Example 2.

homotopy step-size $\tau = 0.1$. For Algorithm 2 the choice of $q = 1/3$ leads to 10 homotopy steps compared to 5 steps in Example 1. Although this is an increase by a factor of two, the overall computational costs increase only slightly. This can be explained by the fact that in each homotopy step there are fewer refinements and overall fewer rejections of homotopy steps than in Example 1. For Algorithm 3 the choice of $q = 1/3$ leads to one additional homotopy step but the computational costs moderately decrease. All these examples show that a proper choice of the parameter $q$ is important for the overall performance of the algorithms.

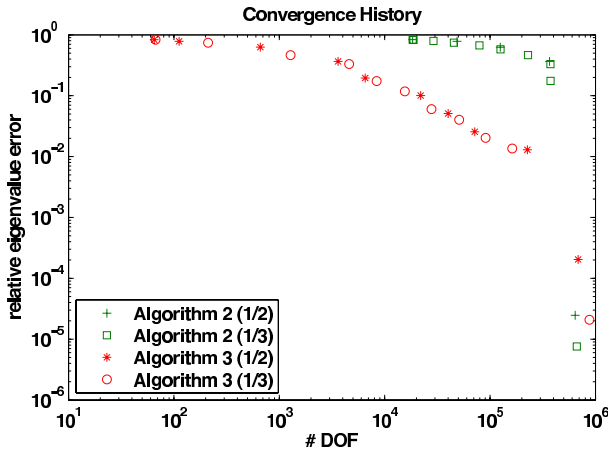


FIGURE 7.6. Comparison of the convergence history of Algorithms 2, and 3 with respect to #DOF for Example 1 and Example 2.

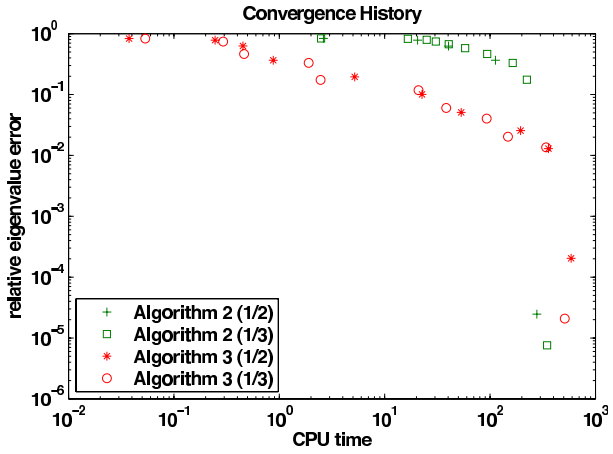


FIGURE 7.7. Comparison of the convergence history of Algorithms 2, and 3 with respect to CPU time for Example 1 and Example 2.

**Example 3.** For this example let $\Omega$ be the (non-convex) L-shaped domain $\Omega = (-1,1)\times(-1,1)\backslash([0,1]\times[-1,1])$. We choose the convection parameter $\beta = (10,0)^T$, the starting point of the homotopy $t_0 = 0$, the marking parameter $\theta = 0.3$, the balancing parameter of the discretization and approximation error estimators $\omega = 0.1$, the step-size update parameter $q = 1/2$, the number of refinement steps $\gamma = 2$, the overall accuracy $\varepsilon = 10^{-1}$, the initial tolerance for the iterative solver $\rho = 1$ and the balancing parameter of the homotopy and discretization error estimators $\delta = 0.1$. A reference value for the eigenvalue with smallest real part is given by

$$\lambda \approx 34.6397.$$

Again for Algorithm 1 a fixed step-size $\tau = 0.1$ is chosen. The results look similar to those of the Examples 1 and 2. The eigenvalue errors for the homotopy steps $t < 1$ are rather large and only the values for $t = 1$ are accurate. Table 7.12 shows that most of the CPU time is used on the last level.

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.0 | 7.1409 | 90.0380 | 0.0020414 | 97.1809 |
| 0.1 | 7.6368 | 83.3621 | 0.0197150 | 91.0186 |
| 0.2 | 5.1146 | 71.0670 | 0.0044748 | 76.1861 |
| 0.3 | 6.3955 | 67.5948 | 0.0474799 | 74.0378 |
| 0.4 | 4.7441 | 58.7391 | 0.0509343 | 63.5341 |
| 0.5 | 3.5712 | 50.2084 | 0.0339932 | 53.8136 |
| 0.6 | 2.5295 | 42.2268 | 0.1135079 | 44.8698 |
| 0.7 | 3.2350 | 33.5816 | 0.0020547 | 36.8187 |
| 0.8 | 2.3482 | 23.5356 | 0.0127627 | 25.8966 |
| 0.9 | 0.9418 | 11.9678 | 0.0041016 | 12.9137 |
| 1.0 | 0.0721 | 0.0000 | 0.0068876 | 0.0790 |

TABLE 7.11. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 1 applied to Example 3.

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.0 | 9.87965 | 0.71479 | 150 | 0.05 |
| 0.1 | 10.11007 | 0.70814 | 150 | 0.07 |
| 0.2 | 10.74190 | 0.68990 | 292 | 0.11 |
| 0.3 | 11.94127 | 0.65527 | 292 | 0.13 |
| 0.4 | 13.64386 | 0.60612 | 488 | 0.18 |
| 0.5 | 15.87295 | 0.54177 | 835 | 0.25 |
| 0.6 | 18.63379 | 0.46207 | 1546 | 0.38 |
| 0.7 | 21.85930 | 0.36895 | 1546 | 0.47 |
| 0.8 | 25.62643 | 0.26020 | 2769 | 0.69 |
| 0.9 | 29.89331 | 0.13702 | 9117 | 1.51 |
| 1.0 | 34.63932 | 0.00001 | 154994 | 79.15 |

TABLE 7.12. The eigenvalue approximation $\tilde{\lambda}_\ell(t)$, the relative eigenvalue error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 1 applied to Example 3.

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.00 | 0.0688 | 64.7314 | 0.0000002 | 64.8002 |
| 0.25 | 0.0669 | 52.1595 | 0.0000032 | 52.2264 |
| 0.50 | 0.0864 | 41.3648 | 0.0000454 | 41.4512 |
| 0.75 | 0.0612 | 24.9728 | 0.0000235 | 25.0340 |
| 1.00 | 0.0654 | 0.0000 | 0.0002845 | 0.0657 |

TABLE 7.13. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 2 applied to Example 3.

Algorithm 2 starts with a step-size $\tau = 1$ which is reduced by the adaptive procedure to $\tau = 0.25$ and afterwards not changed any more. Therefore, Algorithm 2 needs in total only 5 homotopy steps and not 11 as Algorithm 1. Since the discretization error estimator at each homotopy step is forced to be smaller than the

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.00 | 9.64199 | 0.72165 | 18602 | 2.26 |
| 0.25 | 11.20316 | 0.67658 | 28573 | 15.85 |
| 0.50 | 15.88943 | 0.54129 | 39141 | 20.94 |
| 0.75 | 23.70187 | 0.31576 | 99976 | 37.13 |
| 1.00 | 34.62952 | 0.00029 | 168258 | 74.28 |

TABLE 7.14. The eigenvalue approximation $\tilde{\lambda}_\ell(t)$, the relative eigenvalue error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 2 applied to Example 3.

| t | $\eta_\ell(t)$ | $\nu_\ell(t)$ | $\mu_\ell(t)$ | error estimator |
|---|---|---|---|---|
| 0.0000 | 7.1752 | 90.1329 | 0.0023012 | 97.3104 |
| 0.5000 | 4.3043 | 50.7307 | 0.0108970 | 55.0459 |
| 0.7500 | 2.7624 | 29.2814 | 0.1024431 | 32.1463 |
| 0.8750 | 1.1924 | 14.9202 | 0.0143657 | 16.1270 |
| 0.9375 | 0.4360 | 7.5295 | 0.0208416 | 7.9863 |
| 1.0000 | 0.0932 | 0.0000 | 0.0000282 | 0.0932 |

TABLE 7.15. The discretization $\eta_\ell(t)$, the homotopy $\nu_\ell(t)$, and the iteration $\mu_\ell(t)$ error estimator for all homotopy steps $t$ in Algorithm 3 applied to Example 3.

| t | $\tilde{\lambda}_\ell(t)$ | $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$ | #DOF | CPU time |
|---|---|---|---|---|
| 0.0000 | 9.88054 | 0.71476 | 148 | 0.11 |
| 0.5000 | 15.87104 | 0.54183 | 698 | 0.34 |
| 0.7500 | 23.66888 | 0.31671 | 2156 | 0.98 |
| 0.8750 | 28.75123 | 0.16999 | 6912 | 3.10 |
| 0.9375 | 31.60501 | 0.08761 | 22058 | 11.88 |
| 1.0000 | 34.63909 | 0.00002 | 124469 | 32.37 |

TABLE 7.16. The approximation $\tilde{\lambda}_\ell(t)$, the relative error $\frac{\lvert\lambda_\ell(1)-\tilde{\lambda}_\ell(t)\rvert}{\lvert\lambda_\ell(1)\rvert}$, the number of degrees of freedom (#DOF), and the CPU time for all homotopy steps $t$ in Algorithm 3 applied to Example 3.

fixed tolerance $\varepsilon$, the number of degrees of freedom is large already for the first homotopy step. Here, in contrast to the previous examples, the approximation for the last step $t = 1$ is less accurate than for the other two algorithms.

The results for Algorithm 3 show the nature of both other algorithms. The step-size is chosen adaptively without loss of accuracy compared to the eigenvalue error of Algorithm 1. Moreover, it needs only one more homotopy step than Algorithm 2 and the meshes for the step $t < 1$ are much coarser than those of Algorithm 2. Again most of the time is spent to compute the final approximation on the last and second last level. It is also interesting to see that the second last approximation of the eigenvalue obtained in Algorithm 3 is much better than the corresponding one for Algorithm 2, despite using four times fewer DOFs.

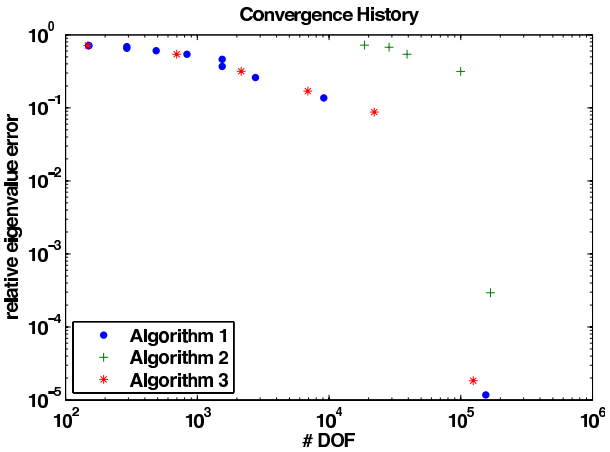


FIGURE 7.8. Convergence history of Algorithms 1, 2, and 3 with respect to #DOF for Example 3.

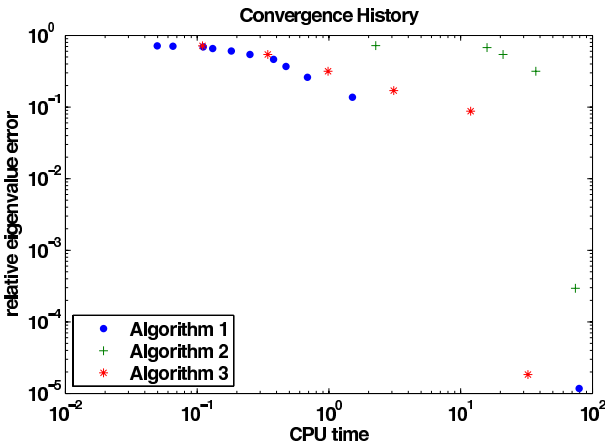


FIGURE 7.9. Convergence history of Algorithms 1, 2, and 3 with respect to CPU time for Example 3.

It is remarkable that for this more complicated example the fastest algorithm, with respect to computational time, is Algorithm 3, see Figure 7.9. This experiment, therefore, strongly underlines the advantages of adaptivity in all three directions, namely the homotopy, the discretization and the approximation.

Figures 7.10, 7.11 and 7.12 show adaptively refined meshes for Algorithms 1, 2 and 3 in Example 3. Note that due to the re-entrant corner the meshes show stronger refinement towards the origin. Since the solution for the selfadjoint problem is known to have a strong singularity at the origin, it is not clear whether this extra refinement results from the homotopy process or from the refinement on the last homotopy step $t = 1$. Indeed, looking at the approximated final primal and dual solutions does not suggest extra refinement, since they have function values close to zero at the origin, but this may be misleading. The fact that the convection acts only along the $x$ axis is clearly visible in the shape of the discrete primal and dual solutions. Note that the primal and dual solution are not mirror images as in the previous examples, but again show strong boundary layers on opposite boundary edges.

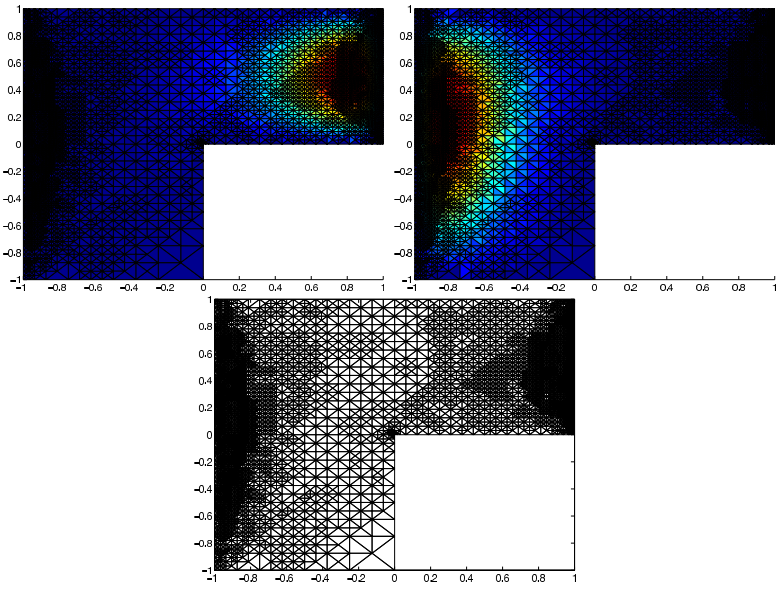

FIGURE 7.10. Primal (top left) and dual (top right) eigenfunction approximations for the final mesh (bottom) with 4926 nodes for Algorithm 1 for Example 3 with $\varepsilon = 3$.

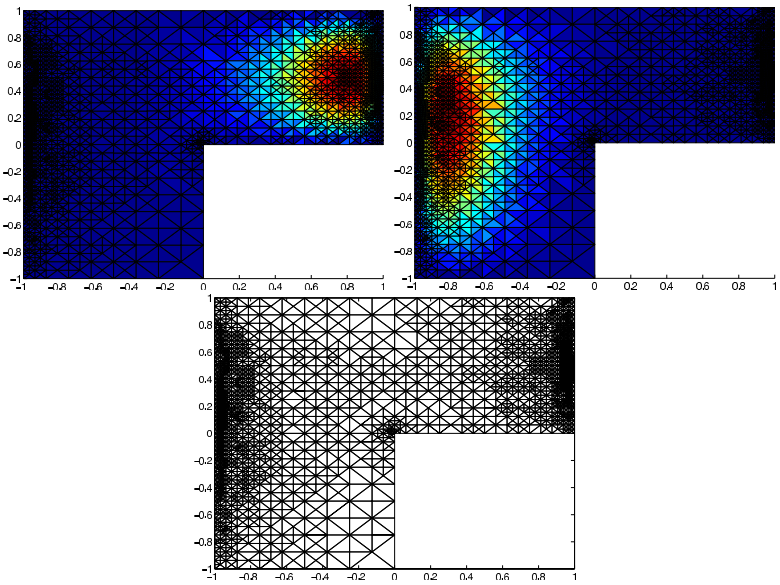

FIGURE 7.11. Primal (top left) and dual (top right) eigenfunction approximations for the final mesh (bottom) with 3694 nodes for Algorithm 2 for Example 3 with $\varepsilon = 3$.

## ACKNOWLEDGMENTS

We thank U. L. Hetmaniuk and R. B. Lehoucq for fruitful discussions and their helpful remarks about ARPACK which improved our work.

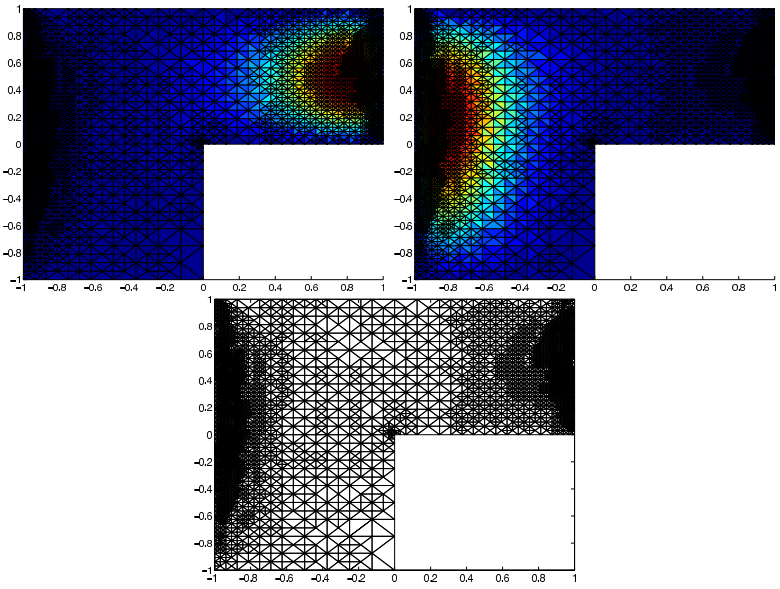

FIGURE 7.12. Primal (top left) and dual (top right) eigenfunction approximations for the final mesh (bottom) with 3745 nodes for Algorithm 3 for Example 3 with $\varepsilon = 3$.

## REFERENCES


[ACF99] J. Alberty, C. Carstensen, and S.A. Funken, *Remarks around 50 lines of Matlab: short finite element implementation*, Numer. Algorithms **20** (1999), 117–137.

[ADRPR01] A. Alonso, A. Dello Russo, C. Padra, and R. Rodríguez, *A posteriori error estimates and a local refinement strategy for a finite element method to solve structural-acoustic vibration problems*, Adv. Comput. Math. **15** (2001), no. 1-4, 25–59 (2002).

[Ait26] A. C. Aitken, *On Bernoulli's numerical solution of algebraic equations.*, Proceedings Royal Soc. Edinburgh **46** (1926), 289–305.

[AKPP08] M. E. Argentati, A. V. Knyazev, C. C. Paige, and I. Panayotov, *Bounds on changes in Ritz values for a perturbed invariant subspace of a Hermitian matrix*, SIAM J. Matrix Anal. Appl. **30** (2008), no. 2, 548–559.

[AO00] M. Ainsworth and J.T. Oden, *A posteriori error estimation in finite element analysis*, John Wiley & Sons, Inc., 2000.

[BDD+00] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the solution of algebraic eigenvalue problem. A practical guide*, SIAM, Philadelphia, 2000.

[BE03] M. Braack and A. Ern, *A posteriori control of modeling errors and discretization errors*, Multiscale Model. Simul. **1** (2003), no. 2, 221–238.

[BFGP99] D. Boffi, P. Fernandes, L. Gastaldi, and I. Perugia, *Computational models of electromagnetic resonators: analysis of edge element approximation*, SIAM J. Numer. Anal. **36** (1999), no. 4, 1264–1290.

[BO89] I. Babuška and J. E. Osborn, *Finite element-Galerkin approximation of the eigenvalues and eigenvectors of selfadjoint problems*, Math. Comp. **52** (1989), no. 186, 275–297.

[BR03] W. Bangerth and R. Rannacher, *Adaptive finite element methods for differential equations*, Birkhäuser, Basel, 2003.

[BS02] S.C. Brenner and L.R. Scott, *The mathematical theory of finite element methods*, second ed., Texts in Applied Mathematics, Springer-Verlag, 2002.

[CG08] C. Cartensen and J. Gedicke, *An oscillation-free adaptive FEM for symmetric eigenvalue problems*, Preprint 489, DFG Research Center MATHEON, Straße des 17.Juni 136, D-10623 Berlin, 2008.

[Cha83] F. Chatelin, *Spectral approximation of linear operators*, Academic Press, New York, 1983.

[Dör96] W. Dörfler, *A convergent adaptive algorithm for Poisson's equation*, SIAM J. Numer. Anal. **33** (1996), 1106–1124.

[DPR03] R. G. Durán, C. Padra, and R. Rodriguez, *A posteriori error estimates for the finite element approximation of eigenvalue problems*, Math. Models Methods Appl. Sci. **13** (2003), 1219–1229.

[Eva00] L.C. Evans, *Partial differential equations*, American Mathematical Society, 2000.

[GC09] J. Gedicke and C. Carstensen, *A posteriori error estimators for non-symmetric eigenvalue problems*, Preprint 659, DFG Research Center MATHEON, Straße des 17.Juni 136, D-10623 Berlin, 2009.

[GG09] S. Giani and I. G. Graham, *A convergent adaptive method for elliptic eigenvalue problems*, SIAM J. Numer. Anal. **47** (2009), no. 2, 1067–1091.

[GMZ09] E.M. Garau, P. Morin, and C. Zuppa, *Convergence of adaptive finite element methods for eigenvalue problems*, Math. Models Methods Appl. Sci. **19** (2009), no. 5, 721–747.

[GO09] L. Grubišić and J.S. Ovall, *On estimators for eigenvalue/eigenvector approximations*, Math. Comp. **78** (2009), no. 266, 739–770.

[GV96] G.H. Golub and C.F. Van Loan, *Matrix computations*, third ed., The Johns Hopkins University Press, 1996.

[HL06] U. L. Hetmaniuk and R. B. Lehoucq, *Uniform accuracy of eigenpairs from a shift-invert lanczos method*, SIAM J. Matrix Anal. Appl. **28** (2006), 927–948.

[HNW93] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations I: Nonstiff problems*, 2nd ed., Springer, Berlin, Germany, 1993.

[HR01] V. Heuveline and R. Rannacher, *A posteriori error control for finite element approximations of elliptic eigenvalue problems*, Adv. Comp. Math. **15** (2001), 107–138.

[HZS+04] D. Heiserer, H. Zimmer, M. Schäfer, C. Holzheuer, and R. Kondziella, *Formoptimierung in der frühen Phase der Karosserieentwicklung*, Vdi-Berichte 1846, Würzburg, 2004.

[KA10] A.V. Knyazev and M.E. Argentati, *Rayleigh-Ritz majorization error bounds with applications to fem*, SIAM. J. Matrix Anal. & Appl. **31** (2010), no. 3, 1521–1537.

[Kat80] T. Kato, *Pertubation theory for linear operators*, Springer, 1980.

[Kat82] ———, *A short introduction to pertubation theory for linear operators*, Springer, 1982.

[Kny97] A.V. Knyazev, *New estimates for Ritz vectors*, Math. Comp. **66** (1997), no. 219, 985–995.

[KO06] A.V. Knyazev and J. E. Osborn, *New a priori FEM error estimates for eigenvalues*, SIAM J. Numer. Anal. **43** (2006), no. 6, 2647–2667.

[Lar00] M.G. Larson, *A posteriori and a priori error analysis for finite element approximations of self-adjoint elliptic eigenvalue problems*, SIAM J. Numer. Anal. **38** (2000), no. 2, 608–625 (electronic).

[LG95] S. H. Lui and G.H. Golub, *Homotopy method for the numerical solution of the eigenvalue problem of slef-adjoint partial differential operators*, Numerical Algorithms **10** (1995), 363–378.

[LKK97] S. H. Lui, H. B. Keller, and T. W. C. Kwok, *Homotopy method for the large sparse real nonsymmetric eigenvalue problem*, SIAM J. Matrix Anal. Appl. **18** (1997), 312–333.

[LSY98] R.B. Lehoucq, D.C. Sorensen, and C. Yang, *ARPACK users' guide: Solution of large-scale eigenvalue problems with implicitly restarted arnoldi methods*, SIAM, Philadelphia, PA. USA, 1998.

[LT03] S. Larsson and V. Thomée, *Partial differential equations with numerical methods*, Springer, Berlin, 2003.

[LZ92] T.Y. Li and Z. Zeng, *Homotopy-determinant algorithm for solving non-symmetric eigenvalue problems*, Math. Comp. **59** (1992), 483–502.

[LZ99] ———, *The homotopy continuation algorithm for the real nonsymmetric eigenproblem: Further development and implementation*, SIAM J. Sci. Comp. **20** (1999), 1627–1651.

[LZC92] T.Y. Li, Z. Zeng, and L. Cong, *Solving eigenvalue problems of real nonsymmetrric matrices with real homotopies*, SIAM J. Numer. Anal. **29** (1992), 229–248.

[MAT10] *MATLAB, Version 7.10.0.499 (R2010a)*, The MathWorks, inc., 24 Prime Park Way, Natick, MA 01760-1500, USA, 2010.

[MM10] V. Mehrmann and A. Miedlar, *Adaptive computation of smallest eigenvalues of elliptic partial differential equations*, Numer. Linear Algebra Appl. (2010), electronically DOI: 10.1002/nla.733.

[MSZ06] D. Mao, L. Shen, and A. Zhou, *Adaptive finite element algorithms for eigenvalue problems based on local averaging type a posteriori error estimates*, Adv. Comp. Math. **25** (2006), 135–160.

[Ney02] K. Neymeyr, *A posteriori error estimation for elliptic eigenproblems*, Numer. Linear Algebra Appl. **9** (2002), no. 4, 263–279.

[OB91] J.E. Osborn and I. Babuška, *Eigenvalue problems*, vol. 2, Handbook of Numerical Analysis, 1991.

[Par98] B. N. Parlett, *The symmetric eigenvalue problem*, SIAM, Philadelphia, PA, 1998.

[PTVF92] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C*, 2nd ed., Cambridge University Press, Cambridge, UK, 1992.

[RT83] P.A. Raviart and J.M. Thomas, *Introduction á l'analyse numérique des equations aux dérivées partielles*, Masson, Paris, 1983.

[Saa92] Y. Saad, *Numerical methods for large eigenvalue problems*, Manchester University Press, Manchester, UK, 1992.

[Sau10] S. Sauter, *hp-finite elements for elliptic eigenvalue problems: error estimates which are explicit with respect to $\lambda$, h, and p*, SIAM J. Numer. Anal. **48** (2010), no. 1, 95–108.

[SF73] G. Strang and G.J. Fix, *An analysis of the finite element method*, Prentice-Hall, Englewood Cliffs, 1973.

[SS90] G. W. Stewart and J. G. Sun, *Matrix perturbation theory*, Academic Press Inc., Boston, MA, 1990.

[TB06] L. N. Trefethen and T. Betcke, *Computed eigenmodes of planar regions*, Contemp. Math. **412** (2006), 297–314.

[Ver96] R. Verfürth, *A review of a posteriori error estimation and adaptive mesh-refinement techniques*, Wiley and Teubner, 1996.

[XZ01] J. Xu and A. Zhou, *A two-grid discretization scheme for eigenvalue problems*, Math. Comp. **70** (2001), no. 233, 17–25.

(C. Carstensen) Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany;
Department of Computational Science and Engineering, Yonsei University, 120–749 Seoul, Korea.
*E-mail address*: cc@mathematik.hu-berlin.de

(J. Gedicke) Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany
*E-mail address*: gedicke@mathematik.hu-berlin.de

(V. Mehrmann) Institut für Mathematik, TU Berlin, Str. des 17. Juni 136, 10623 Berlin, Germany.
*E-mail address*: mehrmann@math.tu-berlin.de

(A. Miedlar) Institut für Mathematik, TU Berlin, Str. des 17. Juni 136, 10623 Berlin, Germany.
*E-mail address*: miedlar@math.tu-berlin.de