METHOD FOR NONLINEAR EIGENPROBLEMS: HYBRIDIZING CONTOUR INTEGRAL AND NEWTON'S METHOD

YOONKYUNG EUNNIE LEE †

2015.05.17
Final Report for 18.335

Abstract. We propose a solution method for nonlinear eigenproblems (NEPs) by hybridizing two known solution methods for NEPs. The first method is traditional Newton iteration, which has excellent quadratic convergence provided that a good initial guess exists. The second method is a different approach using a complex contour integral. This method, referred to as Beyn's method in this work, computes the inverse of the original function and searches for all poles inside a given contour. For analytic functions, this is equivalent to searching for singular points of the original problem. The benefit of Beyn's method is that no initial guess is required, but computing a the matrix inverse for every point on the quadrature is numerically expensive. Our hybrid method combines the strengths of the two methods, which is demonstrated through a short example in this report. We further aim to generalize the solution method towards eigenvalue problems in science and engineering.

Key words. Nonlinear Eigenvalue Problems, (NEPs) Contour Integral (CI) Method, Newton Method, Nonlinear Inverse Iteration

1. Introduction. A nonlinear eigenproblem (NEP) is a generalization of an ordinary eigenproblem to equations that depend nonlinearly on the eigenvalue. We will use this definition and focus on eigenproblems where nonlinearties arise only from the eigenvalue, and not from the eigenvector.

1.1. Motivation. NEPs arise in diverse applications in science and technology, or any other field where a matrix or tensor represents the behavior of a system. [1, 2]. For physical systems characterized with an eigenfrequency ω, solving an eigenproblem is equivalent to finding the condition at which the system is singular and therefore can produce a divergent output in response to a finite input. This is closely related to the concept of resonance. Nonlinearity is extremely common when solving for the eigenfrequencies, because a system that contains energy dissipation, gain, memory, or feedback is nonlinear in ω. The nonlinearity can also arise from the boundary conditions or from the use of special basis functions, even when the original system is linear. Solutions for NEPs is a very active research field, filled with real-world applications and pressing needs from the scientific community.

1.2. General Framework. Given an analytic, matrix-valued function $A(\omega)$ defined on some some subdomain $\mathcal{D}$ of the complex plane, we consider the solution of:

$$A(\omega)\mathbf{v} = 0, \qquad \mathbf{v} \neq 0 \tag{1.1}$$

where $\omega \in \mathcal{D}$ is the nonlinear eigenvalue, and $\mathbf{v} \in \mathbb{C}^n \backslash \{0\}$ is the corresponding eigenvector. Any pair $(\omega, \mathbf{v})$ satisfying 1.1 is called an eigenpair of A, and the set of all eigenvalues is commonly known as the spectrum of A. For regular NEPs with non-empty set of ω, The mapping $\omega \mapsto A(\omega)^{-1}$ is called the resolvent of A and is well defined for all ω in the resolvent set.

An eigenvalue problem is essentially a root-finding problem because the goal is to search for all of the singular points of an operator. The characteristic equation for an NEP can be written as:

$$\det A(\omega) = 0. \tag{1.2}$$

1.3. Solution Method for Linear Eigenproblems . A linear eigenvalue problem is a special case of an NEP with the choice of

$$A(\omega) = \omega I - B, \tag{1.3}$$

in 1.1, where $B \in \mathbb{C}^{n \times n}$ is a constant matrix. It is well known that B has at most n distinct eigenvalues that are the roots of the characteristic polynomial

$$p_B(\omega) = \det(\omega I - B) \tag{1.4}$$

1.5. Overview. In this report, we review existing solution methods for NEPs and suggest to combine CI-based Beyn's method and Newton's iteration to achieve faster convergence to all eigenvalues inside a given contour. We discuss the basic aspects of available solution methods for NEPs in section 2, and review the algorithms for the two methods of interest. In section 3 we suggest and demonstrate the hybrid method. In section 4 we perform error analysis of an example test case using MATLAB's `polyeig` function. Lastly we summarize our results and conclude with a discussion on what future steps to take in order to make this approach useful for research.

2. Solutions of NEPs by Newton's Method . We begin by summarizing the existing numerical algorithms for solving NEPs. Since solving an NEP is equivalent to finding all roots of the characteristic equation 1.2, it is natural to begin by algorithms based on Newton's root-finding method. Newton's method finds the root of $f(\omega) = 0$ by the following iteration:

$$\omega \to \omega - \frac{f(\omega)}{f'(\omega)}. \tag{2.1}$$

We use 2.1 to iterate for the eigenvalue search performed in this report. Below we introduce algorithms that are useful when information about the eigenvector is also wanted.

2.1. Algorithms. For an analytic matrix-valued function $A(\omega)$, we can write:

$$\frac{f(\omega)}{f'(\omega)} = \frac{\det(A(\omega))}{\operatorname{tr}\left(\operatorname{adj}(A(\omega))\dot{A}(\omega)\right)} = \frac{1}{\operatorname{tr}\left(A^{-1}(\omega)\dot{A}(\omega)\right)},$$

where $\mathrm{d}A(\omega)/\mathrm{d}\omega$ is simply written as $\dot{A}(\omega)$. This basic method is referred to as *nonlinear inverse iteration*, [4] and the algorithm is stated below.

Algorithm 1 Newton's method: *nonlinear inverse iteration*

Let e be a normalization vector. Start with an initial guess $(\omega_0, \mathbf{v}_0)$ such that $e^H\mathbf{v}_0 = 1$
for j=1, 2, ... until convergence **do**
  $\mathbf{x}_{j+1} = A(\omega_j)^{-1}\dot{A}(\omega_j)\mathbf{v}_j$
  $\omega_{j+1} = \omega_j - e^H\mathbf{v}_j/e^H\mathbf{x}_{j+1}$
  normalize $\mathbf{v}_{j+1} = \mathbf{x}_{j+1}/e^H\mathbf{x}_{j+1}$
end for

2.1.1. Nonlinear Inverse Iteration. Algorithm 1 is implemented in `NewtInv.m`.

2.1.2. Residual Inverse Iteration. In another variant of Newton's method, the gradient information $\dot{A}(\omega)$ is not required by storing $A(\omega_{j+1})$ and updating the residual instead. This is called *residual inverse iteration*. [5]

$A(\omega_j)^{-1}$ in the above algorithm can be further substituted with $A(\omega_0)^{-1}$ without destroying the convergence. Refer to [5] for a detailed discussion. The residual inverse iteration is not yet implemented in this report.

Algorithm 2 Newton's method: *residual inverse iteration*

Let e be a normalization vector. Start with an initial guess $(\omega_0, \mathbf{v}_0)$ such that $e^H \mathbf{v}_0 = 1$
for j=1, 2, ... until convergence **do**
  solve $e^H A(\omega_j)^{-1} A(\omega_{j+1})\mathbf{v}_j = 0$ for ω_{j+1}
  $\mathbf{v}_{j+1} = \mathbf{v}_j - A(\omega_j)^{-1} A(\omega_{j+1})\mathbf{v}_j$
end for

2.2. Quadratic Convergence. Algorithms 1 and 2 both have asymptotically quadratic convergence towards simple eigenvalues, meaning that the sequence converges with order $q = 2$ to L where

$$\lim_{k\to\infty} \frac{|x_{k+1} - L|}{|x_k - L|^2} = \mu \tag{2.2}$$

where there exists a number $\mu > 0$.

2.3. Operation Count. Each step of nonlinear inverse iteration of algorithm 1 requires a solution of a linear system for $A(\omega)^{-1}$, which asymptotically requires $\mathcal{O}\{m^3\}$ flops.

2.4. Other Methods. Solution methods for NEPs is an active research field, and contributions to the topic is not necessarily limited to Newton-based or CI-based methods. A good review can be found in [6].

2.4.1. Subspace Iteration with Newton-based Methods. A crucial point in iterative projection methods for general NEPs is to prohibit repeated convergence to the same eigenvalue. For this purpose, it can be advantageous to retain the previous approximations when memory is not the biggest concern. A Rayleigh-Ritz procedure for NEPs includes the nonlinear Arnoldi and [7] and nonlinear Jacobi-davidson [8, 9] method.

2.4.2. Block Algorithms for NEPs. Another interesting approach is to create block versions of aforementioned algorithms, which prevents re-convergence to the same eigenvalue by computing all eigenvalues in a cluster simultaneously, which is similar to subspace iteration for linear eigenvalue problems. Kressner proposed a block Newton method for nonlinear inverse iteration in 2009 [10].

3. Solutions of NEPs by Contour Integral Methods. On the other hand, a new class of solution methods for NEPs based on CIs have been developed in the 2000's. It is based on the observation that if we assume all eigenvalues to be distinct, the zeros of an analytic matrix function $A(\omega)$ are equal to the poles of the resolvent $A^{-1}(\omega)$.

3.1. Brief History. CI-based method for generalized EPs was suggested by Sakurai and Sugiura in 2003, [11] and was extended for nonlinear eigenproblems using block Hankel matrices by Asakura *et al.* in 2009. [12] The eigenspace is first constructed using the contour integral, and a subspace iteration is used to solve the GEP. (Hankel: [12], Rayleigh-Ritz: [13])

In this report we follow the formulation of Beyn[14], which shares the same fundamental characteristics with [12, 15].

3.2. Theoretical Framework. The resolvent $A(\omega)^{-1}$ is a finitely meromorphic function, meaning that there exist $\kappa \in \mathbb{N}$ and $S_j \in \mathbb{C}^{n,n}$ for $j \geq -\kappa$ such that $S_{-\kappa} \neq 0$, and

$$A^{-1}(\omega) = \sum_{j=-\kappa}^{\infty} S_j(\omega - \omega_0)^j, \quad \omega \in \mathcal{U} \setminus \{\omega_0\} \tag{3.1}$$

for some neighborhood $\mathcal{U}$ of ω_0. Let us consider a Laurent series expansion of $A(\omega)^{-1}$. According to Keldysh theorem, [14, 16] the principal part of this Laurent series can be expressed in terms of the (generalized) left and right eigenvectors associated with ω:

$$A(\omega)^{-1} = \sum_{k} \frac{1}{\omega - \omega_k} \mathbf{v}_k \mathbf{w}_k^H + R(\omega). \tag{3.2}$$

Here k is the number of eigenvalues inside the contour Γ, ω_k are the complex eigenvalues, $\mathbf{v}_k, \mathbf{w}_k$ are the corresponding left and right eigenvectors, and $R(\omega)$ is the residual that is holomorphic. Using the residue theorem, we can write:

$$A_p = \frac{1}{2\pi i} \int_{\Gamma} \omega^p A^{-1}(\omega) \hat{M} \mathrm{d}\omega \ \in \mathbb{C}^{n \times l}, \tag{3.3}$$

where $\hat{M} \in \mathbb{C}^{n,l}$ is chosen as a random matrix that preserves the rank k with $k \leq l \leq n$. We have reduced the NEP to a k-dimensional linear EP. The choice of $p = 0, 1$ is sufficient when $k < n$, that is, if the number of eigenvalues inside the contour is less than the problem dimension. If not, a more general formalism using $p > 1$ should be used. [14]

3.3. Quadrature Evaluation. The contour integrals in eq. 3.3 are calculated approximately the by trapezoidal sum of a smooth analytical contour. This leads to an exponential decline of the quadrature error with the number of quadrature nodes.

We implement the contour integral using a circular parametrization $\varphi(t) = g_0 + \exp(it)$, discretized using $t_k = 2\pi k/N$, where $k = 0, 1, \ldots, (N-1)$.

$$A_{0,N} = \frac{1}{2\pi i} \int_0^{2\pi} A^{-1}(\varphi(t)) \hat{M} \varphi'(t) \mathrm{d}t \approx \frac{1}{iN} \sum_{k=0}^{N-1} A(\varphi(t_k))^{-1} \hat{M} \varphi'(t_k) \tag{3.4}$$

$$A_{1,N} = \frac{1}{2\pi i} \int_0^{2\pi} A^{-1}(\varphi(t)) \hat{M} \varphi(t) \varphi'(t) \mathrm{d}t \approx \frac{1}{iN} \sum_{k=0}^{N-1} A(\varphi(t_k))^{-1} \hat{M} \varphi(t_k) \varphi'(t_k) \tag{3.5}$$

3.4. Algorithm for SVD and Eigenvalues. Using the theoretical framework layed out above, the next step is to perform the SVD and solve the linearized eigenproblem. This is summarized in algorithm 3.

This algorithm is implemented in `Beyn1.m`.

3.5. Operation Counts and Convergence. Using a CI-based algorithm guarantees that all distinct eigenvalues inside the contour will be located with a choice of a sufficiently large N. From the trapezoidal sum of eq. 3.3, the main numerical

Algorithm 3 Beyn's algorithm for a few eigenvalues

Require: Choose an index $l \leq n$ and a random matrix $\hat{M} \in \mathbb{C}^{n \times l}$.
Compute $A_{0,N}$ and $A_{1,N}$ from 3.4, 3.5.
Compute the SVD $A_{0,N} = V \Sigma W^H$
where $V \in \mathbb{C}^{n \times l}$, $W \in \mathbb{C}^{l \times l}$, $V^H V = W^H W = I_l$, $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_l)$.
Perform a rank test to find k
by $\sigma_k \geq \mathrm{tol}_{\mathrm{rank}} \geq \sigma_{k+1} \approx \ldots \approx \sigma_l \approx 0$.
If $k = l$ then increase l and go to step 1.
Compute $B = V_0^H A_{1,N} W_0 \Sigma_0^{-1} \in \mathbb{C}^{k,k}$.
Solve $B\mathbf{v} = \omega \mathbf{v}$.

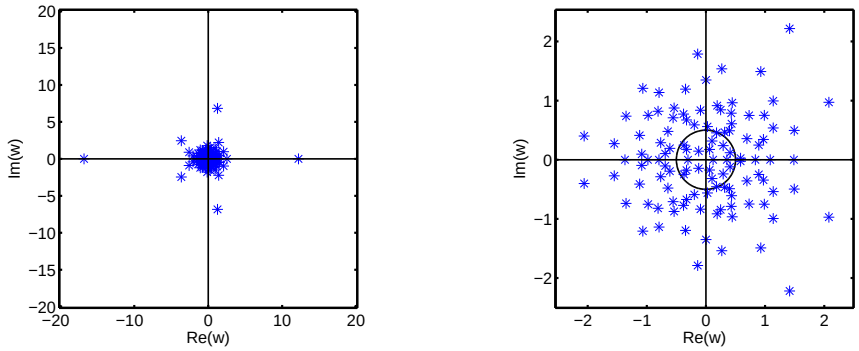

FIG. 1. *Spectrum of* $A(\omega)$. Left: spectrum includign all eigenvalues of A. Right: A closer view around the contour $g_0 + \rho e^{it}$ with $g_0 = 0$, $\rho = 0.5$, $N = 150$.

effort for N quadrature points arise from solving N LU-decompositions and Nk linear systems:

$$\mathcal{O}(Nm^3) + \mathcal{O}(Nkm^3) \approx \mathcal{O}(Nkm^3) \tag{3.6}$$

and therefore the factor N is a big disadvantage the finer the mesh gets. This convergence will be tested in section 4 in further detail.

4. Numerical Implementation and Error Analysis .

5. Strategy for Hybridization. We propose a solution method for NEPs by hybridizing Beyn's CI method and Newton's nonlinear inverse iteration method. First, Beyn's method of algorithm 3 is used to solve for a list of approximate eigenvalues. The biggest computational cost for Beyn's method is the LU factorization and matrix solution that all scale with the number of quadrature points N. ($\mathcal{O}(Nkm^3)$). Our hybrid method aims to remedy the strict condition for N by further refining the convergence by Newton's method of algorithm 1. Newton's method is computationally less expensive but depends on the quality of the initial guess provided. Therefore an optimum condition would exist, where the quality of the initial guess is good enough for Newton convergence while keeping N as small as possible. We aim to determine this optimum N for a numerical example introduced in the next section.

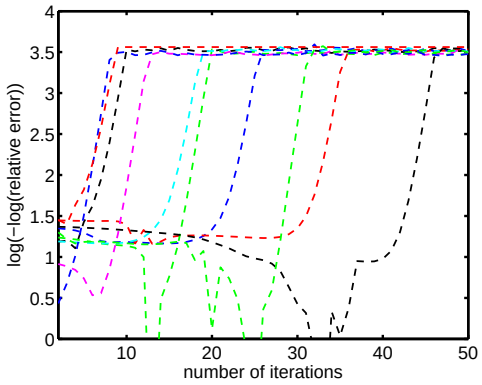

FIG. 2. *Convergence of Newton method with respect to the number of iterations.* Plots represent ten different iterations with random starting points. Linear slope of $log(-log(e_r))$ confirms the quadratic convergence of Newton's method.

5.1. Numerical Example: A Polynomial Eigenproblem. Let us consider the polynomial eigenvalue problem

$$A(\omega) = A_0 + A_1\omega + A_2\omega^2 + \ldots = \sum_{i=0}^{p} \omega^i A_i, \tag{5.1}$$

which can be easily solved using the MATLAB function `polyeig` for the exact solutions. We implement the Beyn-Newton method in MATLAB, and benchmark the performance using polynomial test cases. Unless otherwise noted, the experimentations presented in this report are obtained from the test case of a quadratic eigenproblem, where the coefficients A_0, A_1, A_2 are all generated using MATLAB's `rand(n)` function with $n = 100$. The spectrum of this test problem is illustrated in 5.1.

All calculations were run under MATLAB 8.3 on a Intel Xeon processor with 3.5 GHz and 250GB RAM.

5.2. Benchmarking Newton's Method. The Newton step implementation is analyzed in this section. The quadratic convergence of Newton's method with respect to the number of iteration is verified in figure **??**. We plot the double log of the relative error in figure **??** for ten samples. The error tolerance is set to 10^{-15}, which determines the cut-off value for all log-log error plots. The starting points are selected a random point inside the unit circle. The relative error represents the normalized difference between adjacent Newton steps as below.

$$e_r = \frac{|\omega_{j+1} - \omega_j|}{|\omega_j|} \tag{5.2}$$

The linear slope of the double-log plot $\log(-\log(e_r))$ shows that the convergence of Newton method is indeed quadratic with respect to the number of iterations. The plot also shows how the quality of the initial guess influences the convergence. An interesting finding is that only a small number of steps, about 6 steps in this test case, are required after the quadratic convergence begins. Before the start of fast convergence, the update slowly moves around and has no significant contribution to convergence. Figure 5.2 illustrates this behavior by tracking the Newton step of a

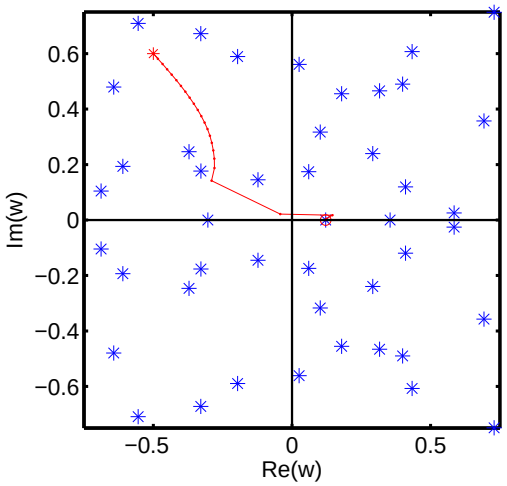

FIG. 3. *Trajectory of Newton iteration on the complex plane.* The rate of convergence is governed by the quality of the initial guess and the surrounding poles in the vicinity of the updated guess. The transition from stagnant region to quadratic convergence is clearly visible.

single eigenvalue on the complex plane. A good starting point would accelerate the initiation of the quadratic convergence and thereby eliminate the idle steps in the beginning.

5.3. Benchmarking Beyn's Method.

5.3.1. Convergence of Trapezoidal Sum. Exponential convergence is expected for the trapezoidal sum of holomorphic periodic integrands. We first check this by plotting the error of two largest eigenvalues with different magnitudes. The error is defined as:

$$e(\omega_k) = \min\left\{|\omega_k - \hat{\omega}| : \hat{\omega} \in \sigma_{\texttt{polyeig.}}\right\}, \tag{5.3}$$

where normalization is omitted for the known answers $\hat{\omega}_1 = -0.371 + 0.247i$ and $\hat{\omega}_2 = 0.179 + 0.455i$. The result is plotted in figure 5.3.1. Figure 5.3.1 confirms the exponential convergence of Beyn's method using a circular contour around 19 eigenvalues. When both eigenvalues and eigenvectors are concerned, the following residual would be an appropriate benchmark:

$$\text{res} = \frac{\|A(\hat{\omega})\hat{\mathbf{v}}\|_2}{\|A(|\hat{\omega}|)\|_F}. \tag{5.4}$$

5.4. Benchmarking Hybrid Beyn-Newton Method. In the hybrid Beyn-Newton method, the results from Beyn's method are used as the initial guess for Newton's iteration until all eigenvalues converge to a desired level ($e_r < 10^{-15}$). The quality of the initial guess is determined by the number of quadrature points on the Beyn contour. Six representative plots are shown in 5.4. In figure 5.4, the convergence of all 19 eigenvalues are plotted together. While most eigenvalues begin with an idle period in $N = 10$, quadratic convergence begins right away for 14 eigenvalues already at $N = 30$. As the number of sampling points increases, most eigenvalues converge before reaching the 8th iteration. Only a single pair remains after 5iterations when

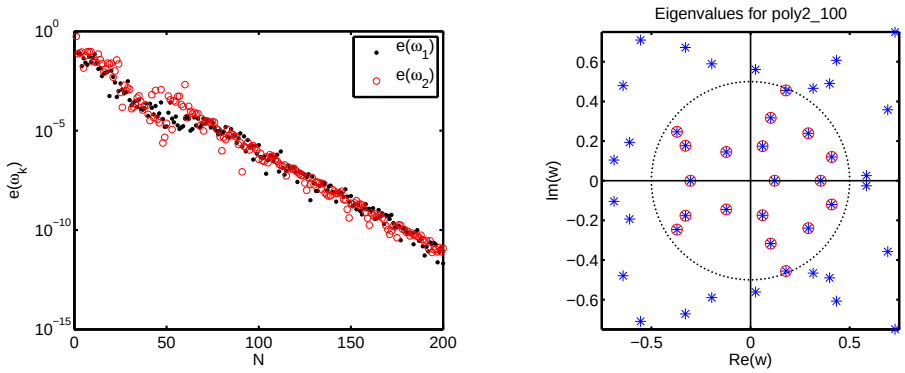

FIG. 4. *Benchmarking Test for Beyn Algorithm.* Left: normalized differences $e(\omega_k)$ for ω_1 and ω_2. Right: plot of the exact eigenvalues from 100×100 quadratic eigenproblem using `polyeig` (blue stars) and the result of Beyn's method (red circles) with $N = 200$, $\hat{M} = rand(100, 25)$

$N = 70$, and $N = 80$ allows all eigenvalues to rapidly converge as soon as the Newton step begins.

The purpose of combining the two methods is two-folds. First, compared to Beyn's method alone, the hybrid method requires less computation to arrive at the same level of convergence. Second, compared to Newton's method alone, the use of an 'educated initial guess' guarantees that no eigenvalue will be missed.

An additional Newton step produces $\mathcal{O}(n_{step} \times m^3)$ work, while an additional contour point N produces $\mathcal{O}(Nkm^3)$ work. If we only consider the amount of flops, $N = 30$ is superior to $N = 50$ or $N = 70$ due to the large computation volume incurred by an increment in N.

5.5. Discussion and Plans.

5.5.1. Convergence of Beyn's Method affected by Eigenvalues on the Contour. Care must be taken to ensure that the contour point does not coincide with an eigenvalue. Figure 5.5.1 shows how the convergence of Beyn's method is slowed down by the existence of an eigenvalue very close to the contour.

5.5.2. Independence of convergence from the sampling matrix $\hat{M}$. We also note that the dimension of the sampling matrix $\hat{M}$ did not produce a clear difference in the eigenvalue convergence. The most important factor for $M \in \mathbb{R}^{n \times l}$ is that it preserves the rank of the original problem, and is not accidentally orthogonal to an existing mode. The use of a random matrix is thus suited for this purpose.

5.5.3. Plan: SLEPC implementation. For the versatile treatment of eigenproblems arising from discretized PDEs, the codes currently written in MATLAB are being translated into C/C++ using Slepc. Another functionality needed is a rigorous benchmarking method that can automatically determine when to switch to Newton iteration, without knowing how much more iteration would occur.

6. Conclusion. We propose a hybrid solution method for NEPs by using a contour integral method with nonlinear inverse iteration based on Newton's method. We implement and verify that the contour integral method is capable of constructing a successful initial guess for Newton's method. For the test case of 19 eigenvalues inside a circular contour around the origin of the complex plane, the quadrature of N=30 already forced 14 eigenvalues to converge after 8 newton iterations.

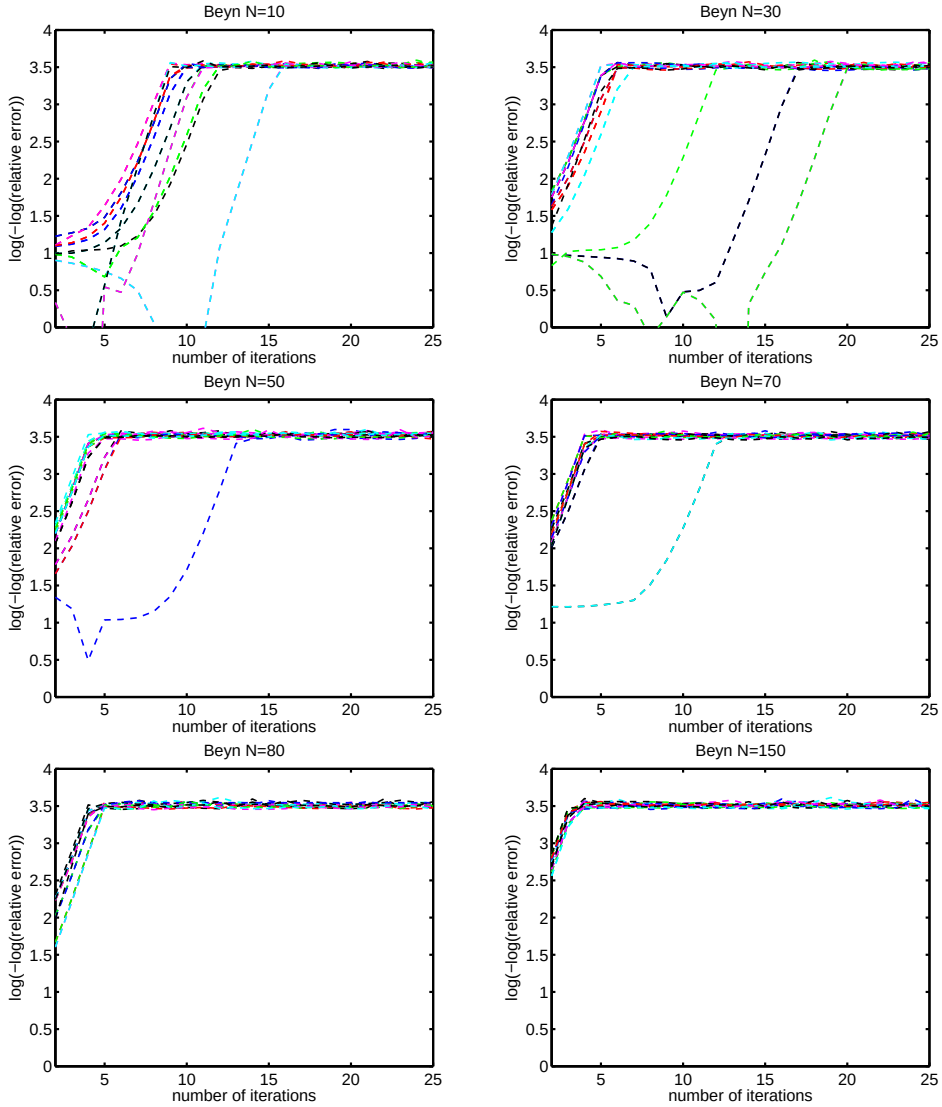

FIG. 5. *Benchmarking Test for Hybrid Beyn-Newton Method.* Accelrated Quadratic Convergence of Hybrid Beyn-Newton Method. Initial guess is given by the output of Beyn's contour integral using $N = 10, 30, 50, 70, 80, 150$. All eigenvalues converge before 5 iterations after the $N > 80$.

The hybridization scheme between the two methods depend on the dimensionality and the nonlinearity involved in each problem. We aim to test more test cases of practical importance using the proposed Beyn-Newton method in the near future.

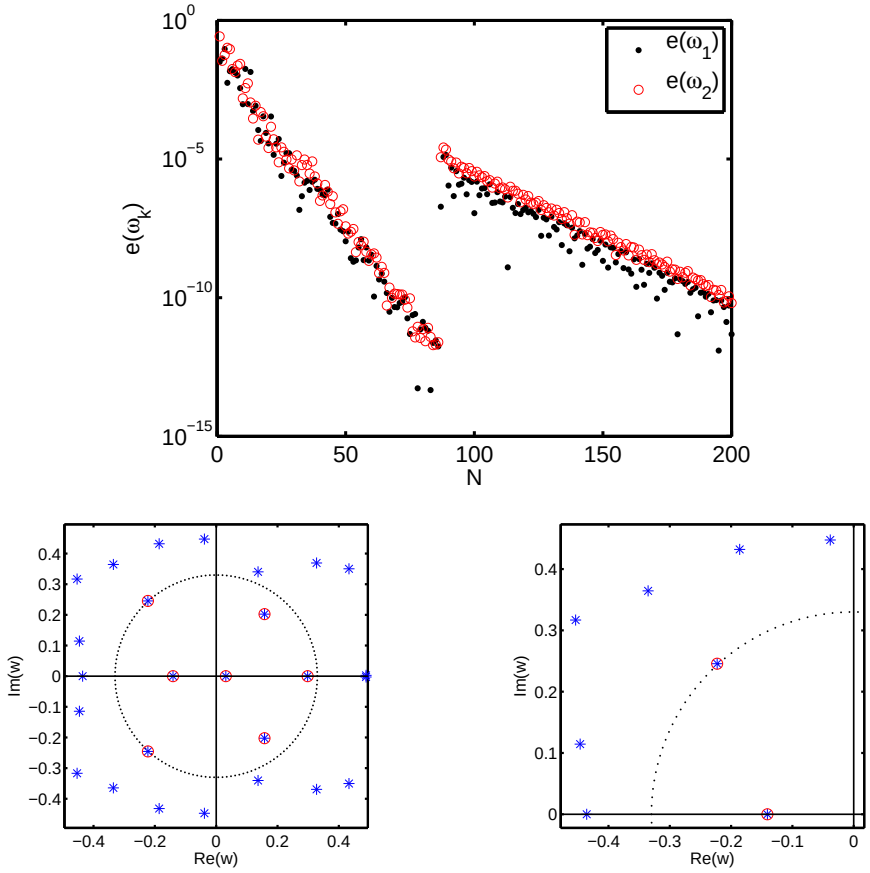

FIG. 6. *Benchmarking Test for Contour overlapping with an Eigenvalue.* Left: normalized differences $e(\omega_k)$ for ω_1 and ω_2. Right: plot of the exact eigenvalues from 60×60 quadratic eigenproblem using `polyeig` (blue stars) and the result of Beyn's method (red circles) with $N = 150$, $\hat{M} = rand(60, 25)$

References.

[1] Philippe Guillaume. Nonlinear eigenproblems. *SIAM journal on matrix analysis and applications*, 20(3):575–595, 1999.

[2] Timo Betcke, Nicholas J Higham, Volker Mehrmann, Christian Schröder, and Françoise Tisseur. Nlevp: A collection of nonlinear eigenvalue problems. *ACM Transactions on Mathematical Software (TOMS)*, 39(2):7, 2013.

[3] Vicente Hernandez, Jose E Roman, and Vicente Vidal. Slepc: A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Transactions on Mathematical Software (TOMS)*, 31(3):351–362, 2005.

[4] P. M. Anselone and L. B. Rall. The solution of characteristic value-vector problems by Newton's method. *Numerische Mathematik*, 11:38–45, 1968.

[5] A Neumaier. Residual inverse iteration for the nonlinear eigenvalue problem. *SIAM journal on numerical analysis*, 22(5):914–923, 1985.

[6] Cedric Effenberger. *Robust solution methods for nonlinear eigenvalue problems*. PhD thesis, cole polytechnique fdrale de Lausanne, 2013.

[7] Heinrich Voss. An arnoldi method for nonlinear eigenvalue problems. *BIT numerical mathematics*, 44(2):387–401, 2004.

[8] Volker Mehrmann and Heinrich Voss. Nonlinear eigenvalue problems: A challenge for modern eigenvalue methods. *GAMM-Mitteilungen*, 27(2):121–152, 2004.

[9] H Voss. A jacobi–davidson method for nonlinear and nonsymmetric eigenproblems. *Computers & Structures*, 85(17):1284–1292, 2007.

[10] Daniel Kressner. A block newton method for nonlinear eigenvalue problems. *Numerische Mathematik*, 114(2):355–372, 2009.

[11] Tetsuya Sakurai and Hiroshi Sugiura. A projection method for generalized eigenvalue problems using numerical integration. *J. Comput. Appl. Math.*, 159:119–128, 2003.

[12] Junko Asakura, Tetsuya Sakurai, Hiroto Tadano, Tsutomu Ikegami, and Kinji Kimura. A numerical method for nonlinear eigenvalue problems using contour integrals. *JSIAM Letters*, 1(0):52–55, 2009.

[13] Shinnosuke Yokota and Tetsuya Sakurai. A projection method for nonlinear eigenvalue problems using contour integrals. *JSIAM Letters*, 5(0):41–44, 2013.

[14] Wolf-Jrgen Beyn. An integral method for solving nonlinear eigenvalue problems. *Linear Algebra and its Applications*, 436(10):3839–3863, May 2012. ISSN 00243795. doi: 10.1016/j.laa.2011.03.030.

[15] Tetsuya Sakurai, Yasunori Futamura, and Hiroto Tadano. Efficient parameter estimation and implementation of a contour integral-based eigensolver. *Journal of Algorithms & Computational Technology*, 7(3):249–270, 2013.

[16] MV Keldysh. On the characteristic values and characteristic functions of certain classes of non-selfadjoint equations, dokl. *AN SSSR*, 77:11–14, 1951.