Received: 14 March 2024 | Revised: 9 July 2024 | Accepted: 1 August 2024

DOI: 10.1002/nla.2584

RESEARCH ARTICLE

WILEY

Dynamically accelerating the power iteration with momentum

Christian Austin[1] | **Sara Pollock**[1] | **Yunrong Zhu**[2]

[1]Department of Mathematics, University of Florida, Gainesville, Florida, USA

[2]Department of Mathematics & Statistics, Idaho State University, Pocatello, Idaho, USA

Correspondence
Sara Pollock, Department of Mathematics, University of Florida, Gainesville, FL 32611-8105, USA.
Email: s.pollock@ufl.edu

Funding information
National Science Foundation, Grant/Award Numbers: DMS-2045059, DMS-1929284

Abstract

In this article, we propose, analyze and demonstrate a dynamic momentum method to accelerate power and inverse power iterations with minimal computational overhead. The method can be applied to real diagonalizable matrices, is provably convergent with acceleration in the symmetric case, and does not require a priori spectral knowledge. We review and extend background results on previously developed static momentum accelerations for the power iteration through the connection between the momentum accelerated iteration and the standard power iteration applied to an augmented matrix. We show that the augmented matrix is defective for the optimal parameter choice. We then present our dynamic method which updates the momentum parameter at each iteration based on the Rayleigh quotient and two previous residuals. We present convergence and stability theory for the method by considering a power-like method consisting of multiplying an initial vector by a sequence of augmented matrices. We demonstrate the developed method on a number of benchmark problems, and see that it outperforms both the power iteration and often the static momentum acceleration with optimal parameter choice. Finally, we present and demonstrate an explicit extension of the algorithm to inverse power iterations.

KEYWORDS

acceleration of convergence, dynamic parameter selection, eigenvector computation, extrapolation, momentum method, power method

1 | INTRODUCTION

In recent years, there is a resurgence of interest in the power method, given its simplicity and ease of implementation. This method to find the dominant eigenmode of a matrix can be applied in a variety of machine learning algorithms, such as PCA, clustering, and low-rank matrix approximations (see References 1 and the references cited therein), PageRank,[2–6] and stability analysis of partial differential equations.[7]

There are a number of generalizations of the power method for large and often sparse systems that can be used to compute extreme eigenvectors or blocks of eigenvectors, relying on matrix-vector multiplications rather than manipulating matrix entries. Among these are the Arnoldi iteration and its variants[3,8,9]; and for symmetric problems, the popular locally optimal block preconditioned conjugate gradient (LOBPCG),[10,11] and the related but more general inverse-free preconditioned Krylov subspace methods.[12,13] These methods all use the idea of iteratively projecting the problem onto a

Krylov subspace of relatively small dimension where dense methods are used to solve a small eigenvalue problem. Additional methods close to this class include the Davidson[14] and Jacobi–Davidson[15] methods from computational chemistry which use a similar idea, but introduce a preconditioner by which the vectors of the projection subspace are no longer equivalent to a Krylov basis.

An alternate and complementary approach to accelerating eigenvector convergence in the power method is based on extrapolation. The idea is to recombine the latest update with previous information to form the next iterate in an approximation sequence. One of the best known methods in this class is Aitken's acceleration[8,16(chapter 9)], with extensions to vector and ϵ extrapolation methods including,[2,6,17,18] to name a few. Recently, several new methods for accelerating the power method with extrapolation have been developed, including, Reference 19 in which the power method is recast as a non-stationary Richardson method; and Reference 20 which damps the largest subdominant eigenmodes to accelerate convergence, and which introduces the idea of computing a dynamic extrapolation parameter based on a ratio of residuals. A similar technique was used in Reference 21 to accelerate the Arnoldi iteration.

In Reference 22, a power method with an added momentum-type extrapolation term was introduced, based on the well known heavy ball method of Reference 23. It was shown that this momentum term accelerates the convergence of the power iteration for positive semidefinite matrices, and the optimal momentum parameter for the acceleration is given by $\beta = \lambda_2^2/4$ where λ_2 is the second largest magnitude eigenvalue of the matrix. A method to add a beneficial momentum term without explicit knowledge of λ_2 was proposed in Reference 22 as the Best Heavy Ball method, which relies on multiple matrix-vector multiplications per iteration throughout the algorithm.

To improve upon this method, a delayed momentum power method (DMPower) was proposed in a more recent paper.[1] The method involves a two-phase approach. The first is a premomentum phase consisting of standard power iterations with inexact deflation, at a cost of three matrix-vector multiplies per iteration, to estimate both λ_1 and λ_2. The second phase runs the method of Reference 22 with fixed momentum parameter β computed with the approximation to λ_2 from the first phase. An analysis is included of how many preliminary iterations are required to obtain a reliable approximation to λ_2, based on a priori spectral knowledge.

In this article, we introduce a dynamic momentum method designed to accelerate the power iteration with minimal additional cost per iteration. In the method proposed herein, the momentum parameter is updated at each iteration based on the Rayleigh quotient and two previous residuals. Like the standard power iteration, this method requires only a single matrix-vector multiplication per iteration. As we will see in Section 4, the introduced dynamic method outperforms not only the power iteration, but also the static momentum method. We additionally show in Section 5 that the method is beneficial when applied to a shifted inverse iteration.

We will consider matrix $A \in \mathbb{R}^{n\times n}$ with eigenvalues $\lambda_1, \dots, \lambda_n$ with $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$. The results trivially generalize to the case where $\lambda_1 = \lambda_2 = \cdots = \lambda_r$ and $|\lambda_r| > |\lambda_{r+1}| \geq \dots \geq |\lambda_n|$. As in Reference 20, our proposed method dynamically updates parameters based on the detected convergence rate computed by the ratio of the last two residuals.

To fix notation, we can write the power iteration as

$$u_{k+1} = Ax_k, \quad x_{k+1} = h_{k+1}^{-1}u_{k+1}, \quad h_{k+1} = \|u_{k+1}\|. \tag{1}$$

The momentum method for the power iteration introduced in Reference 22, takes the form

$$u_{k+1} = Ax_k - \beta h_k^{-1}x_{k-1}, \quad x_{k+1} = h_{k+1}^{-1}u_{k+1}, \quad h_{k+1} = \|u_{k+1}\|, \tag{2}$$

where $\beta > 0$ is the momentum parameter. As shown in Reference 22 and summarized in Section 2, an optimal choice of β is $\lambda_2^2/4$, where it is assumed that $|\lambda_2| < |\lambda_1|$. Our proposed dynamic method based on iteration (2) takes the form

$$u_{k+1} = Ax_k - \beta_k h_k^{-1}x_{k-1}, \quad x_{k+1} = h_{k+1}^{-1}u_{k+1}, \quad h_{k+1} = \|u_{k+1}\|. \tag{3}$$

This method, described in Section 3, assigns the parameter β_k with minimal additional computation (and no additional matrix-vector multiplies), producing a dynamically updated version of (2).

The remainder of this article is structured as follows. Sections 1.1 and 1.2 state the basic assumptions and reference algorithms. In Section 2, we summarize convergence results for the "static" momentum method of Reference 22 through the lens of the power iteration applied to an augmented matrix. While this approach was outlined in Reference 22, our analysis goes a step further, showing that the augmented matrix is defective under the optimal parameter choice. In Section 3 we present the main contributions of this article: our dynamic momentum Algorithm 3, and an analysis of

its convergence and stability. Numerical results for the method are presented in Section 4. In Section 5, we present and discuss Algorithm 5 to accelerate the shifted inverse iteration with momentum.

1.1 | Preliminaries

Our standard assumption throughout the article is the following.

Assumption 1. Suppose $A \in \mathbb{R}^{n\times n}$ is diagonalizable and the n eigenvalues of A satisfy $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$.

Under Assumption 1, let $\{\phi_l\}_{l=1}^n$ be a set of eigenvectors of A so that each (λ_l, ϕ_l) is an eigenpair of A.

In order to analyze the momentum method for A, which we will see is equivalent to a power iteration on an augmented matrix, we will need to make a more general assumption on the augmented matrix.

Assumption 2. Suppose $A \in \mathbb{R}^{n\times n}$ and the n eigenvalues of A satisfy $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$.

The key difference in Assumption 2 is the matrix is not necessarily diagonalizable. In this case we will still refer to the eigenvectors as $\phi_1, \ldots, \phi_n$, but will specify which if any are in fact generalized eigenvectors corresponding to a defective eigenspace.

Throughout the article, $\|\cdot\|$ is the Euclidean or l_2 norm, induced by the l_2 inner-product denoted by $(\cdot,\cdot)$.

1.2 | Reference algorithms

Next we state the power iteration (1) and the momentum iteration (2) in algorithmic form. The algorithm for the momentum iteration will require a single preliminary power iteration, and the algorithm for the dynamic momentum method to be introduced in Section 3 will require two preliminary power iterations.

The algorithm for the power iteration with momentum assumes knowledge of λ_2 to assign the parameter $\beta = \lambda_2^2/4$ and implements the iteration (2).

Algorithm 1. Power iteration

Choose v_0, set $h_0 = \|v_0\|$ and $x_0 = h_0^{-1}v_0$
Set $v_1 = Av_0$
for $k \geq 0$ **do**
  Set $h_{k+1} = \|v_{k+1}\|$ and $x_{k+1} = h_{k+1}^{-1}v_{k+1}$
  Set $v_{k+2} = Ax_{k+1}$
  Set $\nu_{k+1} = (v_{k+2}, x_{k+1})$ and $d_{k+1} = \|v_{k+2} - \nu_{k+1}x_{k+1}\|$
  STOP if $\|d_{k+1}\| < \texttt{tol}$
end for

Algorithm 2. Power iteration with momentum

Set $\beta = \lambda_2^2/4$
Do a single iteration of Algorithm 1 ▷ $k = 0$
for $k \geq 1$ **do** ▷ $k \geq 1$
  Set $u_{k+1} = v_{k+1} - (\beta/h_k)x_{k-1}$
  Set $h_{k+1} = \|u_{k+1}\|$ and $x_{k+1} = h_{k+1}^{-1}u_{k+1}$
  Set $v_{k+2} = Ax_{k+1}$, $\nu_{k+1} = (v_{k+2}, x_{k+1})$ and $d_{k+1} = \|v_{k+2} - \nu_{k+1}x_{k+1}\|$
  STOP if $\|d_{k+1}\| < \texttt{tol}$
end for

2 | BACKGROUND: THE STATIC MOMENTUM METHOD

In this section we will review some results on Algorithm 2, the power iteration with momentum. To this end, we will also review some standard supporting results on the power iteration, Algorithm 1, in both diagonalizable and defective scenarios. These results will be useful to understand each step of the dynamic momentum method.

2.1 | Iteration (2) as a power iteration with an augmented matrix

As shown in Reference 22, the iteration (2) is equivalent to the first n rows of the standard power iteration (1) applied to the augmented matrix

$$A_\beta = \begin{pmatrix} A & -\beta I \\ I & 0 \end{pmatrix}. \tag{4}$$

To see this, consider the power iteration on A_β starting with x_0 in the first component (meaning the first n rows) and y_0 in the second, then writing

$$\begin{pmatrix} u_k \\ z_k \end{pmatrix} = A_\beta \begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} = \begin{pmatrix} Ax_{k-1} - \beta y_{k-1} \\ x_{k-1} \end{pmatrix}. \tag{5}$$

Normalizing each component by a scalar h_k (to be discussed below) with $x_k = h_k^{-1}u_k$ and $y_k = h_k^{-1}z_k = h_k^{-1}x_{k-1}$ yields the iteration

$$\begin{pmatrix} u_{k+1} \\ z_{k+1} \end{pmatrix} = A_\beta \begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} Ax_k - \beta y_k \\ x_k \end{pmatrix} = \begin{pmatrix} Ax_k - \beta h_k^{-1}x_{k-1} \\ x_k \end{pmatrix}. \tag{6}$$

The first component in (6) agrees with (2) if we choose $h_k = \|u_k\|$. Although this is actually a semi-norm over the tuple (u_k, z_k), it is the most convenient choice for the sake of computing the Rayleigh quotient corresponding to the first component at each iteration.

Hence the equivalence between iteration (2) and the power iteration given by (1) as applied to the augmented matrix (4) holds, up to the chosen normalization factor.

Algorithm 2 explicitly performs this iteration starting with $y_0 = 0$ and $\beta = \lambda_2^2/4$, which we discuss further below.

The convergence of iteration (2) for general $\beta \in [0, \lambda_1^2/4)$, $\beta \neq \lambda_i^2/4$, $i = 2, \dots, n$, can be quantified in terms of the convergence of the standard power iteration Algorithm 1. Under Assumption 1, this can be summarized as in Reference 8 (chapter 7) by

$$\operatorname{dist}(\operatorname{span}\{x_k\}, \operatorname{span}\{\phi_1\}) = \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right), \text{ and } |\lambda_1 - \nu_k| = \mathcal{O}\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right), \tag{7}$$

which follows by standard arguments from the expansion of initial iterate u_0 as a linear combination of the n eigenvectors of A, namely $u_0 = \sum_{l=1}^n a_l \phi_l$, by which

$$A^k u_0 = a_1 \lambda_1^k \left(\phi_1 + \sum_{l=2}^n \frac{a_l}{a_1}\left(\frac{\lambda_l}{\lambda_1}\right)^k \phi_l\right). \tag{8}$$

In the case that Assumption 2 holds and A is not diagonalizable, that is, *defective*, the power iteration still converges to the dominant eigenpair. This is the case for A_β when $\beta = \lambda^2/4$ for any subdominant eigenvalue λ of A, as we will see in Proposition 1. For a general defective matrix A, if the eigenspace of λ_1 does not have a full set of eigenvectors then the convergence is slow (like $1/k$, where k is the iteration count), as shown for instance in Reference 16 (chapter 9). If, on the other hand, Assumption 2 holds, A is defective, and the eigenspace for λ_J with $J \geq 2$ lacks a full set of eigenvectors, then

the convergence of Algorithm 1 still agrees with (7), but only asymptotically. In particular, from Reference 16 (chapter 9), if for $|\lambda_J| < |\lambda_1|$ we have $\lambda_J = \lambda_{J+1}$ and the corresponding eigenspace has geometric multiplicity 1, then letting ϕ_{J+1} be a generalized eigenvector with $A\phi_{J+1} = \lambda_J\phi_{J+1} + \phi_J$, in place of (8) we have

$$A^k u_0 = a_1\lambda_1^k\left(\phi_1 + \frac{a_{J+1}}{a_1}\left(\frac{k\lambda_J^{k-1}}{\lambda_1^k}\right)\phi_J + \sum_{l=2}^{n}\frac{a_l}{a_1}\left(\frac{\lambda_l}{\lambda_1}\right)^k\phi_l\right). \tag{9}$$

Noting that $(k\lambda_J^{k-1}/\lambda_1^k)/((k-1)\lambda_J^{k-2}/\lambda_1^{k-1}) \to \lambda_J/\lambda_1$ as $k \to \infty$, we have the same asymptotic convergence rate as in the non-defective case. This is important for the analysis of Algorithm 2 since as shown in the next proposition, the augmented matrix A_β is defective whenever $\beta = \lambda^2/4$ for any eigenvalue λ of A.

2.2 | Spectrum of the augmented matrix

By the equivalence between the first component of the power iteration on A_β and Algorithm 2 as shown in (5)–(6), the convergence rate of the momentum accelerated method of iteration (2) depends on ratio of the two largest magnitude eigenvalues of A_β. In order to understand the convergence properties of Algorithm 2 and later our dynamic version of this method, the following proposition describes the spectral decomposition of A_β in terms of the eigenvalues and eigenvectors of A.

Proposition 1. *Suppose A satisfies Assumption1. Then the $2n$ (counting multiplicity) eigenvalues of A_β are given by*

$$\mu_{\lambda_\pm} = \frac{1}{2}\left(\lambda \pm \sqrt{\lambda^2 - 4\beta}\right), \quad \lambda \in \{\lambda_1, \dots, \lambda_n\}. \tag{10}$$

In the case that $\lambda^2 - 4\beta \neq 0$, the eigenvectors of A_β corresponding to each eigenvalue $\mu = \mu_{\lambda_\pm}$ are given by

$$\psi_{\lambda_\pm} = \begin{pmatrix}(\mu_{\lambda_\pm})\phi\\ \phi\end{pmatrix}, \tag{11}$$

where ϕ is the eigenvector of A corresponding to eigenvalue λ.

In the case that $\beta = \lambda^2/4 > 0$, the matrix A_β is not diagonalizable. Moreover, if λ is an eigenvalue of multiplicity m of A, then the eigenvalue $\mu_\lambda = \lambda/2$ of A_β has algebraic multiplicity $2m$ and geometric multiplicity m.

We restrict our attention to $\beta > 0$ as iteration (2) reduces to (1) if $\beta = 0$. Before the proof of Proposition 1, we include a corollary that follows immediately from its conclusions.

Corollary 1. *If A satisfies Assumption 1 and $\beta \in (0, \lambda_1^2/4)$, then A_β as given by (4) satisfies Assumption 2.*

Together, for symmetric matrices, Proposition 1 and Corollary 1 show that as the power iteration applied to the augmented matrix A_β converges, the first component of the eigenvector converges to the dominant eigenvector of A for any $\beta \in (0, \lambda_1^2/4)$. If $\beta = \lambda^2/4$ for any nonzero $\lambda = \lambda_2, \dots, \lambda_n$, then the matrix A_β is defective, but courtesy of (9), the power iteration will converge asymptotically at the same rate as in the diagonalizable case as given by (8), applied to the eigenvalues of A_β.

Proof. The eigenvectors of A_β are related to the eigenvectors of A by noting that if ϕ is an eigenvector of A with eigenvalue λ then solving

$$A_\beta\begin{pmatrix}\mu\phi\\ \phi\end{pmatrix} = \mu\begin{pmatrix}\mu\phi\\ \phi\end{pmatrix}, \quad \text{which reduces to} \quad \begin{pmatrix}(\mu\lambda - \beta)\phi\\ \mu\phi\end{pmatrix} = \mu\begin{pmatrix}\mu\phi\\ \phi\end{pmatrix},$$

for $\mu \in \mathbb{C}$, yields the quadratic equation $\mu^2 - \lambda\mu + \beta = 0$. If $\beta \neq \lambda^2/4$, the $2n$ eigenvalues of A_β are given by (10), and the corresponding eigenvectors are given by (11).

On the other hand, if $\beta = \lambda^2/4$ where λ is an eigenvalue of A with algebraic multiplicity 1, then the quadratic equation $\mu^2 - \lambda\mu + \beta = 0$ has a repeated root $\mu = \lambda/2$. To find the eigenvector(s) associated with μ, we can express the equation for null-vectors of $A_\beta - \mu I$ as

$$\begin{pmatrix} A - \frac{\lambda}{2}I & -\frac{\lambda^2}{4}I \\ I & -\frac{\lambda}{2}I \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

From the second component of the equation, $u = \frac{\lambda}{2}v$. Applying this to the first component yields $(A - \frac{\lambda}{2}I)\frac{\lambda}{2}v - \frac{\lambda^2}{4}v = 0$, or $Av = \lambda v$. This implies that v must be an eigenvector of A corresponding to eigenvalue λ. Therefore, the eigenspace for A_β corresponding to the repeated eigenvalue $\mu = \lambda/2$ has dimension 1.

More generally, if λ is an eigenvalue of algebraic and geometric multiplicity $m > 1$, then the argument above can be applied to each eigenpair $(\lambda, \hat{\phi}_i)$, $i = 1, \dots, m$, where $\{\hat{\phi}\}_{i=1}^m$ is some basis for the eigenspace corresponding to λ. Then for $\beta = \lambda^2/4$, A_β has an eigenvalue $\mu = \lambda/2$ with algebraic multiplicity $2m$ but with geometric multiplicity m. ■

From (10) of Proposition 1 we have three cases for each pair of eigenvalues of A_β corresponding to a real eigenvalue of A, determined by the sign of the discriminant in (10). Define μ_λ as the larger magnitude eigenvalue of A_β corresponding to eigenvalue λ of A, and $\widehat{\mu}_\lambda$ as the smaller magnitude corresponding eigenvalue, in the case that $\mu_{\lambda_\pm}$ are real. If $\mu_{\lambda_\pm}$ are complex, define μ_λ as having the positive imaginary component. Then

$$(\lambda/2)^2 \geq \beta : \mu_\lambda = \frac{1}{2}\left(\lambda + \text{sign}(\lambda)\sqrt{\lambda^2 - 4\beta}\right), \tag{12}$$

$$(\lambda/2)^2 = \beta : \mu_\lambda = \frac{1}{2}\lambda, \tag{13}$$

$$(\lambda/2)^2 \leq \beta : \mu_\lambda = \sqrt{\beta}e^{i\theta}, \text{ with } \theta = \arctan\left(\sqrt{\frac{4\beta}{\lambda^2} - 1}\right), \tag{14}$$

where (13) agrees with both (12) and (14) at $\beta = (\lambda/2)^2$, and is separately enumerated only for emphasis. In (14), it is understood that $\theta = \pi/2$ when $\lambda = 0$. Based on (14), we see $\beta \geq \lambda_1^2/4$ causes all real eigenvalues of A_β to have equal magnitude $\sqrt{\beta}$.

If A has complex eigenvalues, the complete set of eigenvalues can still be given by $\frac{1}{2}\left(\lambda \pm \sqrt{\lambda^2 - 4\beta}\right)$, applied to each eigenvalue λ of A, however the quantity in the square root may be complex.

We can now summarize the convergence properties of the standard power iteration (1) applied to the augmented matrix A_β given by (4), hence iteration (2), for symmetric matrices A as follows. An alternate approach based on Chebyshev polynomials shown for positive semidefinite matrices can be found in Reference 22.

Corollary 2. *For $0 < \beta < \lambda_1^2/4$, the power iteration (1) implemented in Algorithm 1 applied to the augmented matrix A_β of (4) for symmetric matrix A converges at the rate*

$$\frac{|\mu_{\lambda_2}|}{|\mu_{\lambda_1}|} = \begin{cases} \frac{2\sqrt{\beta}}{|\lambda_1| + \sqrt{\lambda_1^2 - 4\beta}}, & \lambda_2^2/4 < \beta < \lambda_1^2/4 \\ \frac{|\lambda_2| + \sqrt{\lambda_2^2 - 4\beta}}{|\lambda_1| + \sqrt{\lambda_1^2 - 4\beta}}, & 0 \leq \beta < \lambda_2^2/4, \end{cases} \tag{15}$$

and asymptotically at the rate

$$\frac{|\mu_{\lambda_2}|}{|\mu_{\lambda_1}|} \to \frac{|\lambda_2|}{|\lambda_1| + \sqrt{\lambda_1^2 - 4\beta}} = \frac{r}{1 + \sqrt{1 - r^2}}, \text{ with } r = |\lambda_2/\lambda_1|, \text{ for } \beta = \lambda_2^2/4. \tag{16}$$

The choice of β that optimizes the asymptotic convergence rate is $\beta = \lambda_2^2/4$, for which the power iteration applied to A_β and the power iteration with momentum Algorithm 2 applied to A converge asymptotically at the rate given by (16).

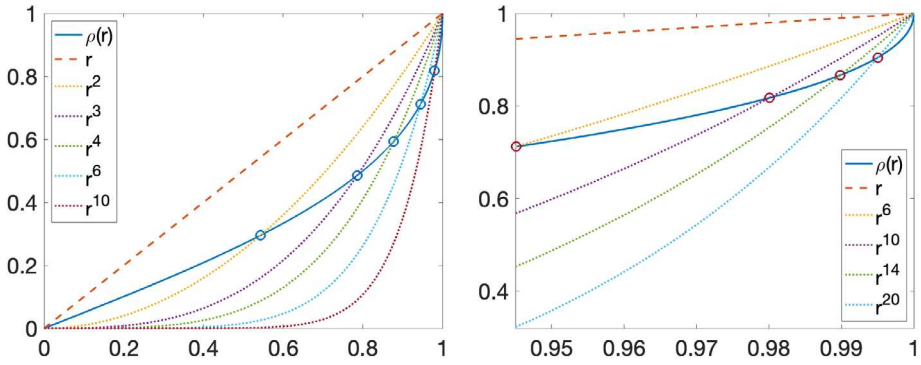

FIGURE 1 A comparison of $\rho(r)$ vs. r^p for $\rho(r) = r/(1+\sqrt{1-r^2})$, the rate given in (16). Left: $\rho(r)$ compared with r^p, for $p = 1, 2, 3, 4, 6, 10$. Right: a detail plot of $\rho(r)$ compared with r^p, for $p = 6, 10, 14, 20$. The crossings between $\rho(r)$ and r^p are marked in each plot.

As visualized in Figure 1, the rate given by (16) is less than r for $r \in (0,1)$, less than r^3 for $r \in (0.786, 1)$, less than r^4 for $r \in (0.878, 1)$, and less than r^6 for $r \in (0.945, 1)$, etc. Hence the smaller the spectral gap in A, namely the closer $r = |\lambda_2/\lambda_1|$ is to 1, the more beneficial it is to apply the acceleration.

Remark 1. Corollary 2 shows that the the power iteration applied to the augmented matrix A_β of (4) converges to the dominant eigenpair $(\mu_\lambda, \psi_\lambda)$ of A_β at a faster rate then the power iteration applied to matrix A converges to its dominant eigenpair (λ, ϕ). Proposition (1) shows that the dominant eigenvector of A is the first component (the first n entries) of the dominant eigenvector of A_β for symmetric A. As the momentum method (2) generates the first component of the power iteration for A_β (using a different normalization factor), this method approximates the dominant eigenvector of A, and converges at the rate described in Corollary 2. The dominant eigenvalue λ of A can then be recovered by taking a Rayleigh quotient with the approximate eigenvector. In practice the augmented matrix A_β is never formed; it is used here as a tool in the analysis of iteration (2).

Proof. The main technicality in the proof of (15) is verifying that $|\widehat{\mu}_{\lambda_1}| < |\mu_{\lambda_2}|$. Then from standard theory, for example, Reference 8, the (asymptotic) rate of convergence to the eigenvector ψ_1 corresponding to μ_{λ_1} is given by $|\mu_{\lambda_2}/\mu_{\lambda_1}|$.

Without loss of generality, suppose $\lambda_1 > 0$. Then for any $\beta \in (0, \lambda_1^2/4)$, we have

$$\widehat{\mu}_{\lambda_1} = \frac{1}{2}\left(\lambda_1 - \sqrt{\lambda_1^2 - 4\beta}\right).$$

By (12)–(14), we have $|\mu_{\lambda_2}| \geq \sqrt{\beta}$. Hence to see that $|\widehat{\mu}_{\lambda_1}| < |\mu_{\lambda_2}|$, it suffices to show that $|\widehat{\mu}_{\lambda_1}| \leq \sqrt{\beta}$. This is true since

$$\sqrt{\beta} - \widehat{\mu}_{\lambda_1} = \sqrt{\beta} - \frac{\lambda_1}{2} + \sqrt{\frac{\lambda_1^2}{4} - \beta} = \sqrt{\frac{\lambda_1}{2} - \sqrt{\beta}}\left(\sqrt{\frac{\lambda_1}{2} + \sqrt{\beta}} - \sqrt{\frac{\lambda_1}{2} - \sqrt{\beta}}\right) \geq 0.$$

The result (15) then follows directly from (12)–(14).

Next we show the asymptotic optimality of $\beta = \lambda_2^2/4$. For this purpose, we consider the convergence rate (15) as a function of β (for $\beta \neq \lambda_2^2/4$) defined as:

$$h(\beta) = \begin{cases} \dfrac{2\sqrt{\beta}}{|\lambda_1| + \sqrt{\lambda_1^2 - 4\beta}}, & \lambda_2^2/4 < \beta < \lambda_1^2/4, \\ \dfrac{|\lambda_2| + \sqrt{\lambda_2^2 - 4\beta}}{|\lambda_1| + \sqrt{\lambda_1^2 - 4\beta}}, & 0 < \beta \leq \lambda_2^2/4. \end{cases}$$

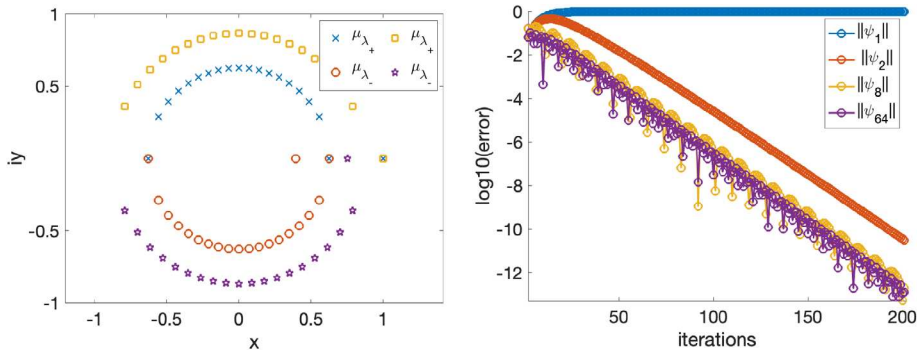

FIGURE 2 Left: The ratio of eigenvalues $\mu_{\lambda_+}/\mu_{\lambda_1}$ and $\mu_{\lambda_-}/\mu_{\lambda_1}$ of the augmented matrix A_β for $A = \text{diag}(10 : -1 : -9)$ with $\beta = 9^2/4$ (inner circle) and $\beta = 9.9^2/4$ (outer circle). Right: convergence of the eigenmodes ψ_1, ψ_2, ψ_8 and ψ_{64} of the augmented matrix A_β for $A = \text{diag}(100 : -1 : 1)$ and $\beta = 99^2/4$. All the subdominant modes converge at the same rate, but with increasing oscillation.

By direct calculation, we get $h'(\beta) > 0$ for $\beta \in (\lambda_2^2/4, \lambda_1^2/4)$, that is, the convergence rate is increasing with respect to β. For $\beta \in (0, \lambda_2^2/4)$, we have $h'(\beta) < 0$, so the convergence rate is decreasing on β. Hence by continuity, $h(\beta)$ achieves a minimum at $\beta = \lambda_2^2/4$. We note that $h(\beta)$ agrees with the convergence rate $|\mu_{\lambda_2}/\mu_{\lambda_1}|$ except when $\beta = \lambda_2^2/4$. When $\beta = \lambda_2^2/4$, the agreement is only asymptotic, that is $|\mu_{\lambda_2}/\mu_{\lambda_1}| \to h(\beta)$. ■

Two interesting observations follow from this analysis. First, as shown in Section 4, as well as in the numerical results of Reference 22, iteration (3) with a well-chosen dynamically assigned sequence of parameters β_k, for which in general $\beta_k \neq \lambda_2^2/4$, can converge faster than the iteration (2) with the optimal parameter $\beta = \lambda_2^2/4$. This can be explained by the above analysis which shows the optimal parameter is only asymptotically optimal. Our results of Sections 4 and 5 show that a close but inexact approximation to this parameter can give a better rate of convergence, at least in the preasymptotic regime.

Second, for $\beta \in [\lambda_2^2/4, \lambda_1^2/4)$, except for μ_{λ_1} and $\widehat{\mu}_{\lambda_1}$ all the remaining $2n - 2$ (complex) eigenvalues of A_β (corresponding to the eigenvalues $\lambda_2, \ldots, \lambda_n$ of A) have the same magnitude $\sqrt{\beta}$ according to (14). However, as the corresponding eigenvalues λ_j of A with $|\lambda_j| < |\lambda_2|$ decrease in magnitude, the argument θ in (14) increases. This causes oscillatory convergence at an increasing rate of oscillation for the subdominant modes. This is illustrated in Figure 2: the left plot shows the ratio of eigenvalues $\mu_{\lambda_\pm}/\mu_{\lambda_1}$ of A_β plotted on the complex plane for $\beta = 9^2/4$ (inner circle) and $\beta = 9.9^2/4$ (outer circle), where $A = \text{diag}(10 : -1 : -9)$. The right plot shows the magnitude of the 1st, 2nd, 8th, and 64th eigenmodes of the power iteration Algorithm 1 applied to the augmented matrix A_β for $A = \text{diag}(100 : -1 : 1)$ with $\beta = 99^2/4$. The plots agree with the above analysis: the modes all decay at the same rate, but the modes of A_β corresponding to the eigenmodes of A with smaller magnitude eigenvalues have larger imaginary parts, and their convergence is more oscillatory. The above analysis also shows that if $\beta \geq \lambda_1^2/4$, then all eigenvalues of A_β have the same magnitude. Therefore, if $\beta \geq \lambda_1^2/4$, the augmented matrix A_β does not satisfy Assumption 2, and neither the power iteration applied to A_β, nor iteration 2 applied to A, will converge.

3 | DYNAMIC MOMENTUM METHOD

We would like to use the acceleration of the momentum Algorithm 2, but without the a priori knowledge of λ_2. A method for determining an effective sequence of momentum parameters is presented in Reference 22 (algorithm 3), called the Best Heavy Ball method. This method however requires five matrix-vector multiplications per iteration, as compared to the single matrix-vector multiplication per iteration required by the standard power iteration Algorithm 1 or the momentum accelerated power iteration[22] presented here as Algorithm 2. This is improved upon in the DMPower algorithm of Reference 1 which uses inexact deflation[24(chapter 4)] in a preliminary iteration to approximate λ_2. However, the method is sensitive to the approximation of λ_2, and ensuring the approximation is good enough again requires a priori knowledge of the spectrum. Additionally, the preliminary iteration is more computationally expensive, requiring 3 matrix-vector multiplications per iteration.

Algorithm 3. Dynamic momentum

Do two iterations of Algorithm 1 ▷ $k = 0, 1$
Set $r_2 = \min\{d_2/d_1, 1\}$
for $k \geq 2$ **do** ▷ $k \geq 2$
  Set $\beta_k = \nu_k^2 r_k^2/4$
  Set $u_{k+1} = v_{k+1} - (\beta_k/h_k)x_{k-1}$
  Set $h_{k+1} = \|u_{k+1}\|$ and $x_{k+1} = h_{k+1}^{-1} u_{k+1}$
  Set $v_{k+2} = Ax_{k+1}$, $\nu_{k+1} = (v_{k+2}, x_{k+1})$ and $d_{k+1} = \|v_{k+2} - \nu_{k+1}x_{k+1}\|$
  Update $\rho_k = \min\{d_{k+1}/d_k, 1\}$ and $r_{k+1} = 2\rho_k/(1 + \rho_k^2)$
  STOP if $\|d_{k+1}\| <$ `tol`
end for

Our approach for approximating the momentum parameter $\beta = \lambda_2^2/4$ does not require any additional matrix-vector multiplication per iteration. We obtain an expression for r_{k+1}, as an approximation of $r = |\lambda_2/\lambda_1|$ from the detected residual convergence rate $\rho_k = d_{k+1}/d_k$ by inverting the optimal convergence rate (16) for r in terms of ρ. This is justified in Lemma 3. We then approximate λ_2 by r_{k+1} multiplied by the (computed) Rayleigh quotient approximation to λ_1, which yields the approximated momentum parameter β_k. The resulting dynamic momentum algorithm is presented below.

Lemma 3 and Remark 3 in the next section show that assigning r_{k+1} by $r_{k+1} = 2\rho_k/(1 + \rho_k^2)$, obtained by inverting the asymptotic convergence rate (16) of the optimal parameter β, gives a stable approximation to r and hence to β. In fact, the approximation becomes increasingly stable as r gets closer to unity.

The next remark describes the role of the subdominant eigenmodes in the residual.

Remark 2. The residual d_k as given in Algorithms 1–3 is given by $d_k = \|Ax_k - \nu_k x_k\|$ where the Rayleigh quotient ν_k is given by (Ax_k, x_k). Let $x_k = \sum_{l=1}^n \alpha_l^{(k)} \phi_l$ where $\{\phi_l\}_{l=1}^n$ is the eigenbasis of A. Then

$$d_k = \left\| \sum_{l=1}^n (\lambda_l \alpha_l^{(k)} \phi_l) - \sum_{l=1}^n (\nu_k \alpha_l^{(k)} \phi_l) \right\| = \left\| \sum_{l=1}^n (\lambda_l - \nu_k) \alpha_l^{(k)} \phi_l \right\|. \tag{17}$$

The detected convergence rate ρ_k is given by

$$\rho_k = \frac{d_{k+1}}{d_k} = \frac{\left\| \sum_{l=1}^n (\lambda_l - \nu_{k+1}) \alpha_l^{(k+1)} \phi_l \right\|}{\left\| \sum_{l=1}^n (\lambda_l - \nu_k) \alpha_l^{(k)} \phi_l \right\|}. \tag{18}$$

We will consider the preasymptotic regime to be that in which $\lambda_1 - \nu_k$ is not negligible in comparison to the coefficients $\alpha_l^{(k)}$, $l > 1$, which will be seen to decay. In the asymptotic regime, we have $\nu_k \approx \lambda_1$ hence (18) reduces for practical purposes to

$$\rho_k \approx \frac{\left\| \sum_{l=2}^n (\lambda_l - \nu_{k+1}) \alpha_l^{(k+1)} \phi_l \right\|}{\left\| \sum_{l=2}^n (\lambda_l - \nu_k) \alpha_l^{(k)} \phi_l \right\|}. \tag{19}$$

In the usual analysis of the power iteration, coefficients $\alpha_l^{(k+1)}$ decay like λ_l/λ_1 at each iteration as in (8), hence eventually (19) is dominated by the maximal such ratio λ_2/λ_1. In contrast, in the case of the augmented matrix A_{β_k}, for each of the eigenmodes with $\lambda^2/4 < \beta_k$, each of the corresponding eigenvalues has the same magnitude; and, as shown in (14) increasing imaginary parts as corresponding eigenvalues of A decrease. Hence it is not necessarily the case that the second eigenmode will dominate (19) through most of the iteration. The oscillation of the subdominant modes is the main reason we will see the sequence of convergence rates ρ_k fluctuate in the dynamic algorithm.

However, the stability of r_k with respect to ρ_k shown in Lemma 3 controls the oscillations in r_k with respect to ρ_k, and substantially damps them in the case that $r = |\lambda_2/\lambda_1|$ is close to unity. In this case there is a more

substantial relative gap between the convergence rate for the second eigenmode and the higher frequency modes, so long as some of the β_k satisfy $\beta_k < \lambda_2^2/4$, which is generally the case. Then the second eigenmode does (eventually) tend to dominate the residual. A further discussion of the coefficients $\alpha_l^{(k)}$ will be given in Remark 4, where it will be shown that $\alpha_l^{(k)}$ is controlled by the product of eigenvalues of the sequence of augmented matrices A_{β_k} corresponding to λ_l of A. The differences in convergence behavior between smaller and larger values of r are highlighted in Section 5.

The following subsection takes into account the nontrivial detail that the dynamic Algorithm 3 differs from a standard power method in that a different augmented matrix A_{β_k} is applied at each iteration.

3.1 | Convergence theory

In Section 2.1, we interpret the convergence of the momentum method with constant β as a power method applied to the augmented matrix A_β. However, this perspective no longer precisely holds for Algorithm 3 as the parameter β_k is subject to change at each step. Consequently, the corresponding augmented matrix A_{β_k} changes at each step as well. This presents a significant challenge in the analysis of the dynamic momentum algorithm.

For ease of presentation, we next define some notation to be used throughout the remainder of this section. Let

$$A^{(0)} = \begin{pmatrix} A & 0 \\ I & 0 \end{pmatrix}, \quad \text{and} \quad A^{(j)} = A_{\beta_j} = \begin{pmatrix} A & -\beta_j I \\ I & 0 \end{pmatrix}, \; j \geq 1,$$

where $A^{(0)}$ is the augmented matrix with $\beta = 0$. As in Section 1.1, let $\{\phi_l\}_{l=1}^n$ be an eigenbasis of A, with corresponding eigenvalues $\{\lambda_l\}_{l=1}^n$. For each eigenpair (λ_l, ϕ_l) of A, denote $(\mu_l^{(j)}, \psi_l^{(j)})$ the corresponding eigenpair of $A^{(j)}$ where $\mu_l^{(j)}$ is the eigenvalue with larger magnitude defined in (12)–(14). Then by (11)

$$\psi_l^{(j)} = \begin{pmatrix} \mu^{(j)}\phi_l \\ \phi_l \end{pmatrix},$$

for $j \geq 0$ with $\mu_l^{(0)} = \lambda_l$.

In the first technical lemma of this section we show the effect of applying a sequence of augmented matrices with changing parameter β_j to each eigenmode of A.

Lemma 1. *Let A satisfy Assumption1, let (λ, ϕ) be an eigenpair of A, and let $\mu^{(j)}$ be the corresponding eigenvalue of $A^{(j)}$, as in Proposition 1. Let $\delta_{ik} = \mu^{(i)} - \mu^{(k)}$. Define $\mathcal{P}^i(\mu)$ to be a product of i terms $\mu^{(k)}$, where $1 \leq k \leq j$, and $\mathcal{P}^i(\delta)$ to be a product of i terms δ_{kp}, where $0 \leq k, p \leq j$. Then*

$$\begin{aligned} A^{(j)} \cdots A^{(0)} \begin{pmatrix} \phi \\ 0 \end{pmatrix} = &\left(\prod_{i=1}^{j} \mu^{(i)} + \sum_{k=1}^{j-1} \delta_{k-1,k} \prod_{i=1, i\neq k}^{j} \mu^{(i)} + \sum_{i=2}^{j-1} \mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu) \right) \begin{pmatrix} \mu^{(j)}\phi \\ \phi \end{pmatrix} \\ &+ \left(\delta_{j-1,j} \prod_{i=1}^{j-1} \mu^{(i)} + \sum_{i=2}^{j} \mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu) \right) \begin{pmatrix} \mu^{(0)}\phi \\ \phi \end{pmatrix}. \end{aligned} \tag{20}$$

This lemma shows that applying the sequence of augmented matrices $A^{(j)} \cdots A^{(0)}$ to each eigenmode of A yields a perturbation to multiplying the eigenmode of $A^{(j)}$ associated with eigenmode ϕ of A by $\mu^{(1)}\mu^{(2)} \cdots \mu^{(j)}$. The higher-order in δ terms of (20) are given in a form that will be used in the next technical lemma. The $\mathcal{P}^i()$ notation is introduced to state the relevant result without keeping track of the specific factors in each product.

Proof. The proof relies on two repeated calculations. First, for any α, β

$$A_\beta \begin{pmatrix} \alpha\phi \\ 0 \end{pmatrix} = \begin{pmatrix} A & -\beta I \\ I & 0 \end{pmatrix} \begin{pmatrix} \alpha\phi \\ 0 \end{pmatrix} = \alpha \begin{pmatrix} \lambda\phi \\ \phi \end{pmatrix} = \alpha \begin{pmatrix} \mu^{(0)}\phi \\ \phi \end{pmatrix}, \tag{21}$$

where $\mu^{(0)} := \lambda$. Second,

$$\begin{aligned} A^{(k)}\begin{pmatrix}\mu^{(j)}\phi\\ \phi\end{pmatrix} &= A^{(k)}\begin{pmatrix}\mu^{(k)}\phi\\ \phi\end{pmatrix} + A^{(k)}\begin{pmatrix}\delta_{jk}\phi\\ 0\end{pmatrix} = \mu^{(k)}\begin{pmatrix}\mu^{(k)}\phi\\ \phi\end{pmatrix} + A^{(k)}\begin{pmatrix}\delta_{jk}\phi\\ 0\end{pmatrix} \\ &= \mu^{(k)}\begin{pmatrix}\mu^{(k)}\phi\\ \phi\end{pmatrix} + \delta_{jk}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}, \end{aligned} \tag{22}$$

where the last term in (22) is the result of (21).

Starting with (21), and proceeding to apply (22) we have

$$A^{(0)}\begin{pmatrix}\phi\\ 0\end{pmatrix} = \begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}, \tag{23}$$

$$A^{(1)}A^{(0)}\begin{pmatrix}\phi\\ 0\end{pmatrix} = \mu^{(1)}\begin{pmatrix}\mu^{(1)}\phi\\ \phi\end{pmatrix} + \delta_{01}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}, \tag{24}$$

$$\begin{aligned} A^{(2)}A^{(1)}A^{(0)}\begin{pmatrix}\phi\\ 0\end{pmatrix} &= \mu^{(2)}\mu^{(1)}\begin{pmatrix}\mu^{(2)}\phi\\ \phi\end{pmatrix} + \delta_{12}\mu^{(1)}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix} + \delta_{01}\mu^{(2)}\begin{pmatrix}\mu^{(2)}\phi\\ \phi\end{pmatrix} + \delta_{02}\delta_{01}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix} \\ &= \left(\mu^{(2)}\mu^{(1)} + \delta_{01}\mu^{(2)}\right)\begin{pmatrix}\mu^{(2)}\phi\\ \phi\end{pmatrix} + \left(\delta_{12}\mu^{(1)} + \mathcal{P}^2(\delta)\right)\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}. \end{aligned} \tag{25}$$

One more iteration reveals the form of the higher order terms.

$$\begin{aligned} A^{(3)}A^{(2)}A^{(1)}A^{(0)}\begin{pmatrix}\phi\\ 0\end{pmatrix} &= \left(\mu^{(2)}\mu^{(1)} + \delta_{01}\mu^{(2)}\right)\left(\mu^{(3)}\begin{pmatrix}\mu^{(3)}\phi\\ \phi\end{pmatrix} + \delta_{23}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}\right) \\ &\quad + \left(\delta_{12}\mu^{(1)} + \mathcal{P}^2(\delta)\right)\left(\mu^{(3)}\begin{pmatrix}\mu^{(3)}\phi\\ \phi\end{pmatrix} + \delta_{03}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}\right) \\ &= \left(\mu^{(3)}\mu^{(2)}\mu^{(1)} + \delta_{01}\mu^{(3)}\mu^{(2)} + \delta_{12}\mu^{(3)}\mu^{(1)} + \mathcal{P}^2(\delta)\mathcal{P}(\mu)\right)\begin{pmatrix}\mu^{(3)}\phi\\ \phi\end{pmatrix} \\ &\quad + \left(\delta_{23}\mu^{(2)}\mu^{(1)} + \mathcal{P}^2(\delta)\mathcal{P}(\mu) + \mathcal{P}^3(\delta)\right)\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}. \end{aligned} \tag{26}$$

Now we may proceed inductively. Suppose

$$\begin{aligned} \Phi^{(j)} := A^{(j)}\cdots A^{(0)}\begin{pmatrix}\phi\\ 0\end{pmatrix} &= \left(\prod_{i=1}^{j}\mu^{(i)} + \sum_{k=1}^{j-1}\delta_{k-1,k}\prod_{i=1,i\neq k}^{j}\mu^{(i)} + \sum_{i=2}^{j-1}\mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu)\right)\begin{pmatrix}\mu^{(j)}\phi\\ \phi\end{pmatrix} \\ &\quad + \left(\delta_{j-1,j}\prod_{i=1}^{j-1}\mu^{(i)} + \sum_{i=2}^{j}\mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu)\right)\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}. \end{aligned} \tag{27}$$

We will show

$$\begin{aligned} \Phi^{(j+1)} &= \left(\prod_{i=1}^{j+1}\mu^{(i)} + \sum_{k=1}^{j}\delta_{k-1,k}\prod_{i=1,i\neq k}^{j+1}\mu^{(i)} + \sum_{i=2}^{j}\mathcal{P}^i(\delta)\mathcal{P}^{j+1-i}(\mu)\right)\begin{pmatrix}\mu^{(j+1)}\phi\\ \phi\end{pmatrix} \\ &\quad + \left(\delta_{j,j+1}\prod_{i=1}^{j}\mu^{(i)} + \sum_{i=2}^{j+1}\mathcal{P}^i(\delta)\mathcal{P}^{j+1-i}\mu)\right)\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix}. \end{aligned} \tag{28}$$

The base step of the induction is satisfied by (26). For the inductive step, applying (22) to $A^{(j+1)}\Phi^{(j)}$ yields

$$\begin{aligned}A^{(j+1)}\Phi^{(j)} = \mu^{(j+1)}&\left\{\left(\prod_{i=1}^{j}\mu^{(i)} + \sum_{k=1}^{j-1}\delta_{k-1,k}\prod_{i=1,i\neq k}^{j}\mu^{(i)} + \sum_{i=2}^{j-1}\mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu)\right)\right.\\ &\left.+\left(\delta_{j-1,j}\prod_{i=1}^{j-1}\mu^{(i)} + \sum_{i=2}^{j}\mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu)\right)\right\}\begin{pmatrix}\mu^{(j+1)}\phi\\ \phi\end{pmatrix}\\ &+\left\{\delta_{j,j+1}\left(\prod_{i=1}^{j}\mu^{(i)} + \sum_{k=1}^{j-1}\delta_{k-1,k}\prod_{i=1,i\neq k}^{j}\mu^{(i)} + \sum_{i=2}^{j-1}\mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu)\right)\right.\\ &\left.+\delta_{0,j+1}\left(\delta_{j-1,j}\prod_{i=1}^{j-1}\mu^{(i)} + \sum_{i=2}^{j}\mathcal{P}^i(\delta)\mathcal{P}^{j-i}(\mu)\right)\right\}\begin{pmatrix}\mu^{(0)}\phi\\ \phi\end{pmatrix},\end{aligned}$$

which after multiplying though and combining the $\mathcal{P}(\cdot)$ terms of like order agrees with (28). ∎

This establishes the result (20). The next step in the argument is to generalize the first component of the initial vector used in Lemma 1 from a single eigenmode of A to a linear combination of eigenmodes of A, to arrive an an estimate analogous to (8).

Lemma 2. *Let A satisfy Assumption1. Let $\delta_{l,i,k} = \mu_l^{(i)} - \mu_l^{(k)}$. As in Lemma 1, define $\mathcal{P}^i(\mu_l)$ to be a product of i terms $\mu_l^{(k)}$, where $1 \le k \le j$, and $\mathcal{P}^i(\delta_l)$ to be a product of i terms $\delta_{l,k,p}$, where $0 \le k, p \le j$. Let $u_0 = \sum_{l=1}^{n} a_l\phi_l$, a linear combination of the eigenvectors of A. Then it holds that the product $A^{(j)}A^{(j-1)}\cdots A^{(0)}\begin{pmatrix}u_0\\0\end{pmatrix}$ satisfies*

$$\begin{aligned}A^{(j)}A^{(j-1)}\cdots A^{(0)}\begin{pmatrix}u_0\\0\end{pmatrix} &= a_1\left(\prod_{i=1}^{j}\mu_1^{(i)}\right)\left\{\psi_1^{(j)} + \sum_{l=2}^{n}\frac{a_l}{a_1}\left(\prod_{i=1}^{j}\frac{\mu_l^{(i)}}{\mu_1^{(i)}}\right)\psi_l^{(j)}\right\}\\ &+\sum_{i=1}^{j-1}a_1\mathcal{P}^{j-i}(\mu_1)\left\{\psi_1^{(j)}\mathcal{P}^i(\delta_1) + \sum_{l=2}^{n}\frac{a_l}{a_1}\mathcal{P}^i(\delta_l)\mathcal{P}^{j-i}\left(\frac{\mu_l}{\mu_1}\right)\psi_l^{(j)}\right\}\\ &+\sum_{i=1}^{j}a_1\mathcal{P}^{j-i}(\mu_1)\left\{\psi_1^{(0)}\mathcal{P}^i(\delta_1) + \sum_{l=2}^{n}\frac{a_l}{a_1}\mathcal{P}^i(\delta_l)\mathcal{P}^{j-i}\left(\frac{\mu_l}{\mu_1}\right)\psi_l^{(0)}\right\}.\end{aligned} \tag{29}$$

Supposing additionally that $\mu_1^{(i)} > \delta_{l,k,p}$ for any $i, k, p = 1, \ldots, j$, and $l \ge 2$, then as j increases, the product $A^{(j)}A^{(j-1)}\cdots A^{(0)}u_0$ aligns to a linear combination of $\psi_1^{(j)}$ and $\psi_1^{(0)}$.

The proof shows additional detail on the $\mathcal{O}(\delta)$ terms, as revealed in Lemma 1. This lemma shows that the product of the sequence of matrices $A^{(j)} \ldots A^{(0)}$ applied to a vector with a general first component, $u_0 \in \mathbb{R}^n$ and null second component $0 \in \mathbb{R}^n$ aligns with a vector whose first component is the dominant eigenvector ϕ_1 of A. It will be shown in Theorem 1 that the convergence is similar to the power method with $(\lambda_l/\lambda_1)^j$ as in (8) replaced by the product $(\mu_l^{(1)}\cdots\mu_l^{(j)})/(\mu_1^{(1)}\cdots\mu_1^{(j)})$. The appreciable difference in the convergence is from the contribution of the δ-scaled terms which are in the directions of the eigenvectors $\psi_l^{(j)}$ and $\psi_l^{(0)}$, with $l = 1, \ldots, n$. As we will see in Theorem 1 and Remark 4, these terms will not interfere with convergence or the asymptotically expected rate, due to the stability of the parameters β_i, as shown in Lemma 3.

Proof. First by applying linearity and (20) we have

$$\begin{aligned}A^{(j)}\cdots A^{(0)}\begin{pmatrix}u_0\\0\end{pmatrix} &= \sum_{l=1}^{n}a_lA^{(j)}\cdots A^{(0)}\begin{pmatrix}\phi_l\\0\end{pmatrix}\\ &= \sum_{l=1}^{n}a_l\left(\prod_{i=1}^{j}\mu_l^{(i)} + \sum_{k=1}^{j-1}\delta_{l,k-1,k}\prod_{i=1,i\neq k}^{j}\mu_l^{(i)} + \sum_{i=2}^{j-1}\mathcal{P}^i(\delta_l)\mathcal{P}^{j-i}(\mu_l)\right)\begin{pmatrix}\mu_l^{(j)}\phi_l\\ \phi_l\end{pmatrix}\\ &+\sum_{l=1}^{n}a_l\left(\delta_{l,j-1,j}\prod_{i=1}^{j-1}\mu_l^{(i)} + \sum_{i=2}^{j}\mathcal{P}^i(\delta_l)\mathcal{P}^{j-i}(\mu_l)\right)\begin{pmatrix}\mu_l^{(0)}\phi_l\\ \phi_l\end{pmatrix}.\end{aligned} \tag{30}$$

Now we will examine each term of (30).

We rewrite the first term in the right hand side of (30) as

$$\sum_{l=1}^{n} a_l \left(\prod_{i=1}^{j} \mu_l^{(i)} \right) \psi_l^{(j)} = a_1 \left(\prod_{i=1}^{j} \mu_1^{(i)} \right) \left\{ \psi_1^{(j)} + \sum_{l=2}^{n} \frac{a_l}{a_1} \left(\prod_{i=1}^{j} \frac{\mu_l^{(i)}}{\mu_1^{(i)}} \right) \psi_l^{(j)} \right\}. \tag{31}$$

This is similar to (8) and displays convergence to ψ_1 so long as the other terms do not interfere. The second term in the right hand side of (30) can be written as

$$\begin{aligned} &\sum_{l=1}^{n} a_l \sum_{k=1}^{j-1} \delta_{l,k-1,k} \left(\prod_{i=1, i\neq k}^{j} \mu_l^{(i)} \right) \psi_l^{(j)} \\ &= \sum_{k=1}^{j-1} a_1 \left(\prod_{i=1, i\neq k}^{j} \mu_1^{i} \right) \left\{ \delta_{1,k-1,k} \psi_1^{(j)} + \sum_{l=2}^{n} \frac{a_l}{a_1} \delta_{l,k-1,k} \left(\prod_{i=1, i\neq k}^{j} \frac{\mu_l^{(i)}}{\mu_1^{(i)}} \right) \psi_l^{(j)} \right\}, \end{aligned} \tag{32}$$

which is an $\mathcal{O}(\delta)$ term where the factors of $\mu_l^{(i)}/\mu_1^{(i)}$ multiplying the subdominant eigenmodes are one power lower than in the dominant term (31). The higher order terms multiplying the eigenvectors of $A^{(j)}$ are

$$\sum_{l=1}^{n} a_l \sum_{i=2}^{j-1} \mathcal{P}^i(\delta_l) \mathcal{P}^{j-i}(\mu_l) \psi_l^{(j)} = \sum_{i=2}^{j-1} a_1 \mathcal{P}^{j-i}(\mu_1) \left\{ \psi_1^{(j)} \mathcal{P}^i(\delta_1) + \sum_{l=2}^{n} \frac{a_l}{a_1} \mathcal{P}^i(\delta_l) \mathcal{P}^{j-i} \left(\frac{\mu_l}{\mu_1} \right) \psi_l^{(j)} \right\}. \tag{33}$$

Next, we look at the terms of (30) multiplying the eigenvectors $\psi_0^{(0)}$ of $A^{(0)}$. The lowest order term is $\mathcal{O}(\delta)$ and is given by

$$\sum_{l=1}^{n} a_l \delta_{l,j-1,j} \left(\prod_{i=1}^{j-1} \mu_l^{(i)} \right) = a_1 \left(\prod_{i=1}^{j-1} \mu_1^{(i)} \right) \left\{ \delta_{1,j-1,j} \psi_1^{(0)} + \frac{a_l}{a_1} \delta_{l,j-1,j} \left(\prod_{i=1}^{j-1} \frac{\mu_l^{(i)}}{\mu_1^{(i)}} \right) \psi_l^{(0)} \right\}. \tag{34}$$

Last we have the higher order terms

$$\sum_{l=1}^{n} a_l \left(\sum_{i=2}^{j} \mathcal{P}^i(\delta_l) \mathcal{P}^{j-i}(\mu_l) \right) \psi_l^{(0)} = \sum_{i=2}^{j} a_1 \mathcal{P}^{j-i}(\mu_1) \left\{ \psi_1^{(0)} \mathcal{P}^i(\delta_1) + \sum_{l=1}^{n} \frac{a_l}{a_1} \mathcal{P}^i(\delta_l) \mathcal{P}^{j-i} \left(\frac{\mu_l}{\mu_1} \right) \psi_l^{(0)} \right\}. \tag{35}$$

Sweeping the results of the more detailed (32) into (33), and likewise (34) into (35) yields the result (29). Finally, the alignment of the product (29) to a combination of $\psi_1^{(j)}$ and $\psi_1^{(0)}$ follows from noting each ratio $(\mu_l^{(i)}/\mu_1^{(i)}) < 1$ and applying the hypothesis $\mu_1^{(i)} > \delta_{l,k,p}$ for any $i, k, p = 1, \ldots, j$, and $l \geq 2$. ∎

The next lemma shows that if ρ_k is an ϵ perturbation of $\rho = |\mu_2/\mu_1|$, then r_{k+1} is an $\hat{\epsilon}$ perturbation of r, where $\hat{\epsilon} < 2\epsilon$ for $\rho \in (0, 1)$, and $\hat{\epsilon} \to 0$ as $\rho \to 1$. This means the smaller the spectral gap in A, the more stable the dynamic momentum method becomes.

Lemma 3. *Let $\rho \in (0, 1)$ and consider ϵ small enough so that $(2\rho\epsilon + \epsilon^2)/(1 + \rho^2) < 1$. Let $\rho_k = \rho + \epsilon$ and define $r_{k+1} = 2\rho_k/(1 + \rho_k^2)$, as in Algorithm 3. Then*

$$r_{k+1} = r + \hat{\epsilon} + \mathcal{O}(\epsilon^2) \text{ with } \hat{\epsilon} = \epsilon \frac{2(1 - \rho^2)}{(1 + \rho^2)^2}. \tag{36}$$

The condition $(2\rho\epsilon + \epsilon^2)/(1 + \rho^2) < 1$ is satisfied for $\rho \in (0, 1)$ by $\epsilon < 0.71$.

Proof. For $r = |\lambda_2/\lambda_1|$ the asymptotic convergence rate of iteration (2) is $\rho = r(1 + \sqrt{1 - r^2})^{-1}$, as given by (16), when $\beta = \lambda_2^2/4$. Inverting this expression for r in terms of ρ yields

$$r = \frac{2\rho}{1 + \rho^2}. \tag{37}$$

Suppose the detected convergence rate $\rho_k = d_{k+1}/d_k$ is an ϵ perturbation of ρ, meaning $\rho_k = \rho + \varepsilon$. Expanding $r_{k+1} = 2\rho_k/(1+\rho_k^2)$ in ε yields

$$\begin{aligned} r_{k+1} &= \frac{2\rho}{1+(\rho+\varepsilon)^2} + \frac{2\varepsilon}{1+(\rho+\varepsilon)^2} \\ &= \frac{2\rho}{1+\rho^2}\left(\frac{1}{1+\frac{2\rho\varepsilon+\varepsilon^2}{1+\rho^2}}\right) + \frac{2\varepsilon}{1+\rho^2}\left(\frac{1}{1+\frac{2\rho\varepsilon+\varepsilon^2}{1+\rho^2}}\right) \\ &= \frac{2\rho}{1+\rho^2}\left(1 - \frac{2\rho\varepsilon+\varepsilon^2}{1+\rho^2} + \mathcal{O}(\varepsilon^2)\right) + \frac{2\varepsilon}{1+\rho^2} + \mathcal{O}(\varepsilon^2). \end{aligned} \tag{38}$$

Applying (37) to (38) yields

$$r_{k+1} = r\left(1+\varepsilon\left(\frac{1}{\rho} - r\right)\right) + \mathcal{O}(\varepsilon^2) = r + \widehat{\varepsilon} + \mathcal{O}(\varepsilon^2), \text{ where } \widehat{\varepsilon} = r\varepsilon\left(\frac{1}{\rho} - r\right). \tag{39}$$

Applying (37) to (39) yields the result (36), by which $\widehat{\varepsilon} < 2\varepsilon$ for $\rho \in (0,1)$, and $\widehat{\varepsilon} < \varepsilon$ for $\rho \in (0.486, 1)$, or $r \in (0.786, 1)$. Moreover as r getting closer to unity, the approximation becomes more stable, with $\widehat{\varepsilon} < 0.161 \cdot \varepsilon$ for $r \in (0.99, 1)$ and $\widehat{\varepsilon} < 0.0468 \cdot \varepsilon$ for $r \in (0.999, 1)$. ∎

The stability of $\beta_k = (r_k \nu_k)^2/4$ in Algorithm 3 is inherited directly from the stability of r_k, once ν_k sufficiently converges to λ_1.

Remark 3. Another way to view how close β_k is to $\beta = \beta_{opt} = \lambda_2^2/4$ with respect to r_k and ρ_k viewed as perturbations of r and ρ is to consider ρ_k written as

$$\rho_k = \frac{r\sqrt{1+\varepsilon/r^2}}{1+\sqrt{1-r^2(1+\varepsilon/r^2)}},$$

for some ε with $-r^2 < \varepsilon < 1 - r^2$. Applying $r_{k+1} = 2\rho_k/(1+\rho_k^2)$ we then have $r_{k+1} = r\sqrt{1+\varepsilon/r^2}$, by which $\beta_{k+1} = r^2(1+\varepsilon/r^2)\nu_{k+1}^2/4$. For $\nu_{k+1} \approx \lambda_1$ this yields

$$\beta_{k+1} \approx \frac{\lambda_2^2}{4} + \varepsilon\frac{\lambda_1^2}{4},$$

which shows how perturbations r_k with respect to r result in perturbations to β_k with respect to β.

Now we can summarize the results of this section in a convergence theorem.

Theorem 1. *Let A satisfy Assumption 1. Let $\delta_{l,i,k} = \mu_l^{(i)} - \mu_l^{(k)}$, and let $u_0 = \sum_{l=1}^n a_j\phi_j$, a linear combination of the eigenvectors of A.*

If A is symmetric then $\beta_k < \lambda_1^2/4$ for all k. Then (29) holds and Algorithm 3 converges to the dominant eigenpair.

Here we proceed by assuming generically that none of the β_k take a value of exactly equal to $\lambda^2/4$ for any eigenvalue λ of A. This is a reasonable assumption due both to floating point arithmetic, and that as shown in Lemma 3, the β_k only converge to $\lambda_2^2/4$ as $r \to 1$, and we are always in the circumstance that $r < 1$.

Proof. By the definitions of ρ_k and r_k, we have $r_k^2 \leq 1$. Since ν_k is the Rayleigh quotient with approximate eigenvector x_{k+1} and symmetric A, it follows that $\beta_k < \lambda_1^2/4$.

We will start by developing bounds on the $\mu_l^{(i)}$ and the $\delta_{l,i,k}$, and in the process will verify the final hypothesis of Lemma 2 by verifying $|\delta_{l,i,k}| \leq \max\{|\mu_l^{(i)}|, |\mu_l^{(k)}|\}$. We will also see that $\delta_{l,i,k} \to 0$ as $\beta_i - \beta_k \to 0$ for each λ_l. Consider $\lambda = \lambda_l \neq 0$. There are three cases we need to consider. Without loss of generality, suppose $\beta_i \geq \beta_k$.

(i) If $\lambda^2/4 \geq \beta_i$ then $\mu_l^{(i)}$ and $\mu_l^{(k)}$ are given by (12), and $|\mu_l^{(i)}| \in [|\lambda|/2, |\lambda|]$ for $l = 2, \dots, n$. Since we have $\beta < \lambda_1^2/4$ we have $|\mu_1^{(i)}| \in (|\lambda_1|/2, |\lambda_1|]$. To bound $\delta_{l,i,k}$ we have

$$|\mu_l^{(i)} - \mu_l^{(k)}| = \frac{|\lambda|}{2}\left|\sqrt{1-4\beta_i/\lambda^2} - \sqrt{1-4\beta_k/\lambda^2}\right| < \frac{|\lambda|}{2} \leq |\mu_l^{(i)}| < |\mu_l^{(k)}|. \tag{40}$$

It is clear from continuity and (40) that $\delta_{l,i,k} \to 0$ as $\beta_i - \beta_k \to 0$. We can also expand to first order to see how, yielding $|\delta_{l,i,k}| = |(\beta_i - \beta_k) + \dots|$.

(ii) If $\lambda^2/4 \leq \beta_i, \beta_k < \lambda_1^2/4$ then $\mu_l^{(i)}$ and $\mu_l^{(k)}$ are given by (14), and we have $|\mu_l^{(i)}| = \sqrt{\beta_i}$. In this case $\delta_{l,i,k}$ satisfies

$$|\mu_l^{(i)} - \mu_l^{(k)}| = \left|\sqrt{\beta_i}\sqrt{1-\lambda^2/(4\beta_i)} - \sqrt{\beta_k}\sqrt{1-\lambda^2/(4\beta_k)}\right| \leq \sqrt{\beta_i} = |\mu_l^{(i)}|. \tag{41}$$

From continuity and (41) it is clear that $\delta_{l,i,k} \to 0$ as $\beta_k - \beta_i \to 0$. Expanding (41) to first order to see how, yields $|\mu_l^{(i)} - \mu_l^{(k)}| = |(\sqrt{\beta_i} - \sqrt{\beta_k})(1 - \lambda^2/8) + \dots|$.

(iii) If $\beta_k \leq \lambda^2/4 \leq \beta_i$, then we have

$$\mu_l^{(i)} - \mu_l^{(k)} = \frac{\lambda}{2} + \frac{1}{2}\sqrt{\lambda^2 - 4\beta_i} - \left(\frac{\lambda}{2} + \frac{1}{2}\sqrt{\lambda^2 - 4\beta_k}\right) = \frac{i}{2}\sqrt{4\beta_i - \lambda^2} - \frac{1}{2}\sqrt{\lambda^2 - 4\beta_k},$$

by which $|\mu_l^{(i)} - \mu_l^{(k)}| = \sqrt{\beta_i - \beta_k} \leq \sqrt{\beta_i} = |\mu_l^{(i)}|$.

Combining with the above results, we have $|\delta_{l,k,p}| \leq \max\{|\mu_l^{(k)}|, |\mu_l^{(p)}|\}$, for any $l, k, p = 1, \dots, j$.

We now have by Lemma 2 that as j increases, the product $A^{(j)}A^{(j-1)} \cdots A^{(0)}\begin{pmatrix} u_0 \\ 0 \end{pmatrix}$ aligns with a linear combination of $\psi_1^{(j)}$ and $\psi_1^{(0)}$. As in Section 2.1, we now analyze the convergence of Algorithm 3 by the convergence of

$$\begin{pmatrix} x_{j+1} \\ y_{j+1} \end{pmatrix} = \frac{1}{h_{j+1}}\begin{pmatrix} u_{j+1} \\ z_{j+1} \end{pmatrix} = \frac{1}{h_{j+1}}\left(\prod_{i=0}^{j} h_i^{-1}\right)A^{(j)}A^{(j-1)} \cdots A^{(0)}\begin{pmatrix} u_0 \\ 0 \end{pmatrix}, \tag{42}$$

where $h_j = \|u_j\|$. Applying (29) to (42), we have

$$\begin{aligned}\begin{pmatrix} u_{j+1} \\ z_{j+1} \end{pmatrix} = {} & \frac{a_1}{h_0}\left(\prod_{i=1}^{j}\frac{\mu_1^{(i)}}{h_i}\right)\left\{\psi_1^{(j)} + \sum_{l=2}^{n}\frac{a_l}{a_1}\left(\prod_{i=1}^{j}\frac{\mu_l^{(i)}}{\mu_1^{(i)}}\right)\psi_l^{(j)}\right\} \\ & + \left(\prod_{i=0}^{j}\frac{1}{h_i}\right)\sum_{i=1}^{j-1} a_1 \mathcal{P}^{j-i}(\mu_1)\left\{\psi_1^{(j)}\mathcal{P}^i(\delta_1) + \sum_{l=2}^{n}\frac{a_l}{a_1}\mathcal{P}^i(\delta_l)\mathcal{P}^{j-i}\left(\frac{\mu_l}{\mu_1}\right)\psi_l^{(j)}\right\} \\ & + \left(\prod_{i=0}^{j}\frac{1}{h_i}\right)\sum_{i=1}^{j} a_1 \mathcal{P}^{j-i}(\mu_1)\left\{\psi_1^{(0)}\mathcal{P}^i(\delta_1) + \sum_{l=2}^{n}\frac{a_l}{a_1}\mathcal{P}^i(\delta_l)\mathcal{P}^{j-i}\left(\frac{\mu_l}{\mu_1}\right)\psi_l^{(0)}\right\}.\end{aligned} \tag{43}$$

Distributing through the normalization factors in (43) yields

$$\begin{aligned}\begin{pmatrix} u_{j+1} \\ z_{j+1} \end{pmatrix} = {} & \frac{a_1}{h_0}\left(\prod_{i=1}^{j}\frac{\mu_1^{(i)}}{h_i}\right)\left\{\psi_1^{(j)} + \sum_{l=2}^{n}\frac{a_l}{a_1}\left(\prod_{i=1}^{j}\frac{\mu_l^{(i)}}{\mu_1^{(i)}}\right)\psi_l^{(j)}\right\} \\ & + \sum_{i=1}^{j-1}\frac{a_1}{h_0}\left(\frac{\mathcal{P}^{j-i}(\mu_1)}{\prod_{k=1}^{j-i}h_k}\right)\left\{\psi_1^{(j)}\mathcal{P}^i(\delta_1) + \sum_{l=2}^{n}\frac{a_l}{a_1}\left(\frac{\mathcal{P}^i(\delta_l)}{\prod_{k=j-i+1}^{j}h_k}\right)\mathcal{P}^{j-i}\left(\frac{\mu_l}{\mu_1}\right)\psi_l^{(j)}\right\} \\ & + \sum_{i=1}^{j}\frac{a_1}{h_0}\left(\frac{\mathcal{P}^{j-i}(\mu_1)}{\prod_{k=1}^{j-i}h_k}\right)\left\{\psi_1^{(0)}\mathcal{P}^i(\delta_1) + \sum_{l=2}^{n}\frac{a_l}{a_1}\left(\frac{\mathcal{P}^i(\delta_l)}{\prod_{k=j-i+1}^{j}h_k}\right)\mathcal{P}^{j-i}\left(\frac{\mu_l}{\mu_1}\right)\psi_l^{(0)}\right\}.\end{aligned} \tag{44}$$

By the arguments above, $\begin{pmatrix} u_{j+1} \\ z_{j+1} \end{pmatrix}$ aligns with a linear combination of $\psi_1^{(j)}$ and $\psi_1^{(0)}$, both of which have first components in the direction of ϕ_1. This further shows that the Rayleigh quotient $\nu_k = (Ax_k, x_k) \to \lambda_1$, by which the residual (17) converges to zero. ∎

We conclude this section with a heuristic discussion of the coefficients of each eigenmode that appear in the residual, as per Remark 2.

Remark 4. By Theorem 1, the Rayleigh quotient ν_k converges to λ_1. As in Remark 2, we consider the asymptotic regime where $\nu_k \approx \lambda_1$, so that the ratio between consecutive residuals ρ_k is well approximated by

$$\rho_j \approx \frac{\left\| \sum_{l=2}^{n} (\lambda_l - \nu_{j+1}) \alpha_l^{(j+1)} \phi_l \right\|}{\left\| \sum_{l=2}^{n} (\lambda_l - \nu_j) \alpha_l^{(j)} \phi_l \right\|}. \tag{45}$$

From (45), and the definition of the eigenvectors of the augmented matrix in (11), the coefficients $\alpha_l^{(j+1)}$, $l \geq 2$ are given by

$$\alpha_l^{(j+1)} = \frac{a_l}{\prod_{i=0}^{j+1} h_i} \left\{ \mu_l^{(j)} \left(\prod_{i=1}^{j} \mu_l^{(i)} \right) + \sum_{i=1}^{j-1} (\mu_l^{(j)} + 1) \mathcal{P}^{j-i}(\mu_l) \mathcal{P}^{i}(\delta_l) + \mathcal{P}^{j}(\delta_l) \right\}. \tag{46}$$

We next make the argument that the first term inside the brackets in (46) dominates the others.

From Theorem 1, each $\delta_{l,i,k}$ satisfies $|\delta_{l,i,k}| \leq \max\{\mu_l^{(i)}, \mu_l^{(k)}\}$. Referring to the proof of Lemma 1, each of the factors of $\delta_{l,i,k}$ have either the form $\delta_{l,p-1,p}$ or $\delta_{l,0,p}$, where p ranges from 1 to $j+1$. As per the discussion in Theorem 1, the terms of the form $\delta_{l,p-1,p}$ go to zero as the $\beta_k \to \beta = \lambda_2^2/4$. By Lemma 3, considering the detected convergence rate ρ_k as a perturbation of the theoretically optimal rate ρ, the computed approximation r_{k+1} to $r = |\lambda_2/\lambda_1|$ is restricted to a tighter interval about r when r is closer to one. By this argument, and Remark 3, β_{k+1} is restricted to a small interval around β (smaller as r getting closer to 1). So as j increases, terms with of the form $\delta_{l,j-1,j}$ become negligible. By these arguments, each of the terms under the sum of (46) should be of equal order or less than the first term, and as j increases, additional terms under the sum should be essentially negligible.

By inspecting the proof of Lemma 1, the final term in (46) can be seen to be $\delta_{l,0,1}\delta_{l,0,2} \cdots \delta_{l,0,j}$. By the same arguments above, this term should also be of equal order or less than the first, although $\delta_{l,0,j}$ is not in general expected to become negligible as j increases. In conclusion, the coefficients $\alpha_l^{(j)}$ are dominated by the products of the eigenvalues $\mu_l^{(i)}$, $i = 1, \ldots, j$.

4 | NUMERICAL RESULTS

In this section, we include four suites of tests comparing the introduced dynamic momentum method Algorithm 3 with the power method Algorithm 1 and the static momentum method with optimal $\beta = \lambda_2^2/4$ as in Algorithm 2. We include additional comparisons in the first three test suites with the delayed momentum power method (DMPOW)[1(algorithm 1)]. In the last test suite we include comparisons with Algorithm 2 with the parameter β replaced by small perturbations above and below the optimal value.

In our implementation of DMPOW we do not assume any spectral knowledge, and we consider 20, 100, and 500 preliminary power iterations with deflation in the preliminary stage to determine an approximation of λ_2. As each of the preliminary iterations contains 3 matrix-vector multiplications, that number where it is reported exceeds the number of total iterations for DMPOW as it includes both stages of the algorithm. The other methods tested each require one matrix-vector multiply per iteration. We found we were able to improve the performance of DMPOW by choosing w_0, which is the the initial approximation to the second eigenvector, to be orthogonal to u_0 (denoted q_0 in Reference 1). We used this technique in DMPOW for all reported results.

All tests were performed in Matlab R2023b running on an Apple MacBook Air with 24GB of memory, 8 core CPU with 8 core GPU. Throughout this section, each iteration was run to a maximum of 2000 iterations or a residual tolerance of 10^{-12}. We include tests started from the fixed initial iterate $u_0 = \begin{pmatrix}1 & 1 & \cdots & 1\end{pmatrix}^T$ so that the results can be reproduced, as well as tests starting from random initial guesses via `u0 = (rand(n,1)-0.5);`. To run Algorithm 2 which requires knowledge of λ_2, we recovered the first two eigenvalues using `eigs(A,2)`. We emphasize that we did this for comparison purposes only, and that our interest is in developing effective methods that do not require any a priori information of the spectrum.

4.1 | Test suite 1

Our first test suite consists of three symmetric positive definite (SPD) benchmark problems. All three matrices have similar values of $r \approx 0.999$. This first is a diagonal matrix included for its transparency. The second matrix `Kuu` is used as a benchmark in Reference 21. The third, `Muu` features $\lambda_2 = \lambda_1$, so we demonstrate replacing λ_2 with λ_3 in Algorithm 2. Our dynamic Algorithm 3 works as expected without modification.

Matrix 1: $A = \text{diag}(1000 : -1 : 1)$. This matrix is a standard benchmark with $r = 0.999$.
Matrix 2: $A =$ `Kuu` from Reference 25, with $n = 7102$. This matrix has leading eigenvalues $\lambda_1 = 54.0821$ and $\lambda_2 = 53.9817$, with $r = 0.9981$.
Matrix 3: $A =$ `Muu` from Reference 25, with $n = 7102$. This matrix has leading eigenvalues $\lambda_1 = 10^{-3} \times 0.8399$, $\lambda_2 = 10^{-3} \times 0.8398$ and $\lambda_3 = 10^{-3} \times 0.8391$. Using `eigs`, λ_1 and λ_2 agreed to 10^{-14}, and Algorithm 2 did not converge using $\beta = \lambda_2^2/4$. The results shown use $\beta = \lambda_3^2/4$, as λ_3 is the second largest eigenvalue for this matrix. Taking in this case $r = \lambda_3/\lambda_1$ yields $r = 0.9992$.

Figure 3 shows iteration count vs. the residual norm using Algorithm 1, DMPOW with 20, 100 and 500 preliminary iterations, Algorithm 2, and Algorithm 3. Each iteration was started with the initial $u_0 = \begin{pmatrix}1 & 1 & \cdots & 1\end{pmatrix}^T$. The preliminary iterations of DMPOW were started with $w_0 = \begin{pmatrix}-1 & 1 & -1 & 1 \cdots\end{pmatrix}^T$, so that w_0 is orthogonal to u_0. In the first two cases, we see the dynamic method Algorithm 3 converges at approximately the same asymptotic rate as Algorithm 2, though in the second case the latter has an extended preasymptotic regime. The three DMPOW instances work essentially as they should for Matrix 1 and Matrix 2, where the approximation of λ_2 from the deflation method, hence the approximation of $\beta_{opt} = \lambda_2^2/4$ improves as the preliminary iterations are increased. For Matrix 1 DMPOW with 500 preliminary iterations does appear to achieve the optimal convergence rate. In Matrix 3 on the right, only the dynamic method Algorithm 3 achieves a steady optimal convergence rate. Algorithm 2 initially stalls then achieves a good but suboptimal rate. DMPOW with 500 preliminary iterations achieves an apparently optimal but oscillatory convergence rate, with sub-optimal rates with 100 and 20 preliminary iterations. The oscillatory behavior of DMPOW suggests that the approximation to λ_2 is greater than λ_2, hence all subdominant modes are oscillatory, via (12).

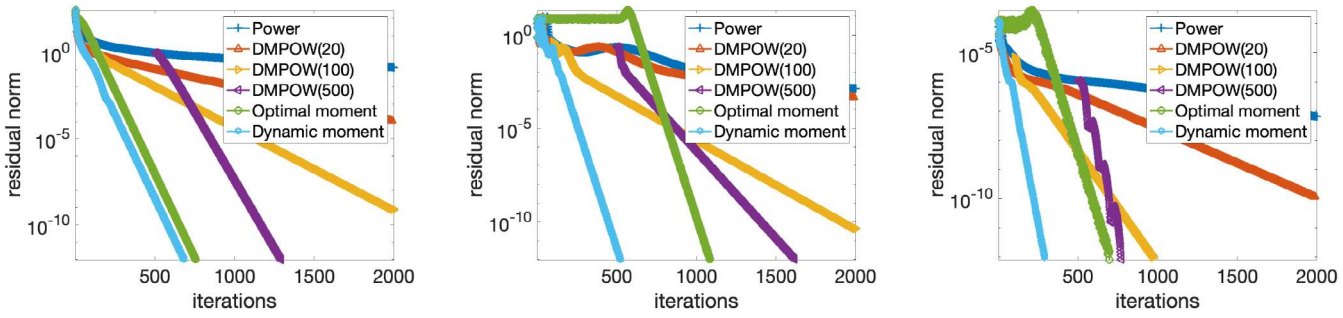

FIGURE 3 Convergence of the residual by iteration count for the three matrices in test suite 1, using Algorithm 1, DMPOW with 20, 100 and 500 preliminary iterations, Algorithm 2, and Algorithm 3. Left: Matrix 1, diag(1000 : −1 : 1); center: Matrix 2, `Kuu`; right: Matrix 3, `Muu`.

TABLE 1 Minimum and maximum number of matrix-vector multiplies to residual convergence for 100 runs of the power method (Algorithm 1), DMPOW run with 20, 100, and 500 preliminary iterations, the power iteration with optimal momentum (Algorithm 2) and the power iteration with dynamic momentum (Algorithm 3), applied to Matrix 4–Matrix 7.

	Matrix 4		**Matrix 5**		**Matrix 6**		**Matrix 7**	
Method	min	max	min	max	min	max	min	max
Power	247	359	1088	1583	2000	2000	2000	2000
DMPOW(20)	125	746	197	655	2040	2040	2040	2040
DMPOW(100)	340	376	415	1735	1281	2200	1318	2200
DMPOW(500)	1503	1503	1575	1626	1767	3000	1970	3000
$\beta = \lambda_2^2/4$	71	**86**	152	179	**241**	**288**	550	640
Dynamic β	**66**	96	**133**	**175**	255	652	**470**	**612**

Note: Each run used a randomly generated initial vector.

4.2 | Test suite 2

The second test suite consists of four matrices. The first three are symmetric indefinite and the fourth is SPD with increasing gaps between the smaller eigenvalues.

Matrix 4: $A =$ `ash292` from Reference 25, with $n = 292$. This matrix has leading eigenvalues $\lambda_1 = 9.1522$ and $\lambda_2 = 8.3769$, with $r = 0.9153$. It is symmetric indefinite.

Matrix 5: $A =$ `bcspwr06` from Reference 25, with $n = 1454$. This matrix has leading eigenvalues $\lambda_1 = 5.6195$ and $\lambda_2 = 5.5147$, with $r = 0.9814$. It is symmetric indefinite.

Matrix 6: $A = \text{diag}($`linspace`$(-99,100,200))$. This matrix has $n = 200$, $\lambda_1 = 100$, $\lambda_{2,3} = \pm 99$, and $r = 0.99$. It is included to test the sensitivity to positive and negative leading subdominant eigenvalues.

Matrix 7: $A = \text{diag}(10-$`logspace`$(0,1,200))$. This matrix has $n = 200$, $\lambda_1 = 9$, $\lambda_2 = 8.9884$, and $r = 0.999$. It is included to test the sensitivity to increasing gaps between smaller eigenvalues.

Results of the experiments with the second set of matrices is shown in Table 1. We see that the dynamic Algorithm 3 shows more sensitivity to initial vector than does the static algorithm with optimal parameter 2 in the indefinite cases, and particularly for the highly indefinite Matrix 6. From the results for Matrix 7, we see that increasing the spacing between the smaller eigenvalues does not cause increased sensitivity to u_0. We can also see that Algorithms 2 and 3 significantly outperform the others on all tests in this suite.

4.3 | Test suite 3

For the third test suite, we generated 100 symmetric matrices with unit diagonal, and quasi-randomly generated normally distributed off-diagonals with mean zero and standard deviation one, via `v = ones(n,1); v1 = randn(n-1,1); A = diag(v,0) + diag(v1,1) + diag(v1,-1);`. For each matrix, we checked the ratio $r = |\lambda_2/\lambda_1|$. Over the 100 matrices, the values of r ranged from 0.7944 to 0.9996, with mean value 0.9491 and standard deviation 0.0455. Each run was started with the initial iterate $u_0 = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}^T$.

Table 2 shows the results. While Algorithm 2 with optimal fixed β has the lowest minimal number of iterations over 100 runs, dynamic Algorithm 3 has the lowest mean and maximum iteration count. For these two methods the iteration count is the same as the reported number of matrix-vector multiplies. On the other hand, DMPOW with 20, 100, and 500 preliminary iterations each had at least one run that did not terminate after 2000 total iterations (preliminary included), and all three of the DMPOW methods had a substantially higher minimum number of matrix-vector multiplies than either the optimal or dynamic methods.

TABLE 2 The number of matrix-vector multiplies and terminal residual for 100 runs of the power method (Algorithm 1), DMPOW run with 20, 100 and 500 preliminary iterations, the power iteration with optimal momentum (Algorithm 2) and the power iteration with dynamic momentum (Algorithm 3).

	Matrix-vector multiplies				Terminal residual	
Method	mean	std. dev.	min	max	min	max
Power	905.42	658.728	96	2000	8.74e-13	4.89e-04
DMPOW(20)	439.15	528.614	101	2040	6.31e-13	4.43e-04
DMPOW(100)	498.3	295.655	302	2200	2.15e-13	7.95e-12
DMPOW(500)	1600.55	196.335	1502	3000	1.78e-13	4.96e-06
$\beta = \lambda_2^2/4$	162.22	160.065	**40**	1183	6.02e-13	9.99e-13
Dynamic β	**150.15**	**133.842**	63	**949**	5.66e-13	1.00e-12

Note: Each run used pseudo-randomly generated tridiagonal matrix `v = ones(n,1); v1 = randn(n-1,1); A = diag(v,0) + diag(v1,1) + diag(v1,-1);`, and the same initial vector $u_0 = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}^T$.

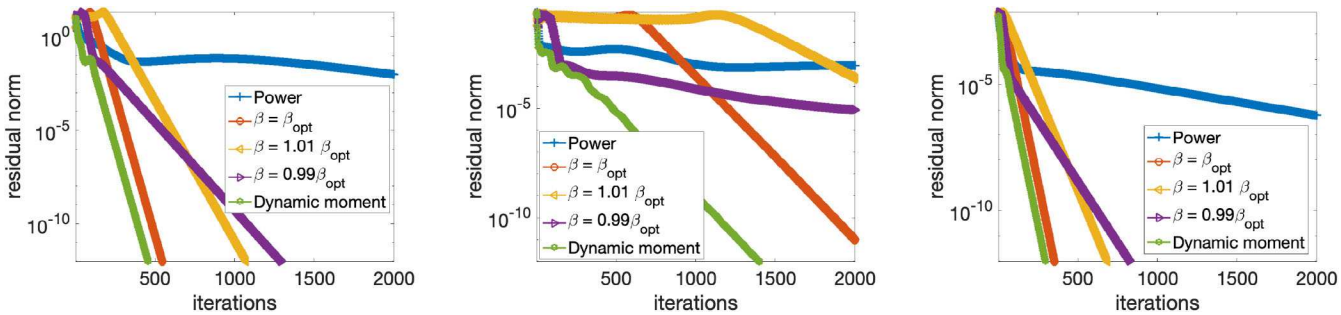

FIGURE 4 Convergence of the residual by iteration count for the three matrices in test suite 4, using Algorithm 1, Algorithm 2 with $\beta = \beta_{opt} = \lambda_2^2/4$, with $\beta = \min\{1.01 \times \beta_{opt}, (3\lambda_1^2 + \lambda_2^2)/16\}$, $\beta = 0.99 \times \beta_{opt}$, and and Algorithm 3. Left: Matrix 8, `Si5H12`; center: Matrix 9, `ss1`; right: Matrix 10, `thermomech_TC`.

4.4 | Test suite 4

In this fourth suite of tests, we consider three problems of varying structure and scale, and which have eigenvalues of varying magnitudes. The dynamic momentum Algorithm 3 is tested against the power method 1, the static momentum Algorithm 2 with optimal parameter $\beta = \lambda_2^2/4$, and perturbations thereof, $\beta_- = 0.99 \times \beta_{opt}$, and $\beta_+ = \min\{1.01 \times \beta_{opt}, (3\lambda_1^2 + \lambda_2^2)/16\}$. The parameters β_+ and β_- are within 1% of β_{opt}, but do not exceed $\lambda_1^2/4$, which as per Section 2 would prevent convergence.

Matrix 8: $A =$ `Si5H12` from Reference 25, with $n = 19{,}896$. This matrix has leading eigenvalues $\lambda_1 = 58.5609$ and $\lambda_2 = 58.4205$, with $r = 0.998$. It is symmetric indefinite.

Matrix 9: $A =$ `ss1` from Reference 25, with $n = 205{,}282$. This matrix has leading eigenvalues $\lambda_1 = 1.3735$ and $\lambda_2 = 1.3733$, with $r = 0.9998$. It is nonsymmetric.

Matrix 10: $A =$ `thermomech_TC`, with $n = 102{,}158$. This matrix has leading eigenvalues $\lambda_1 = 0.03055$ and $\lambda_2 = 0.03047$, with $r = 0.9975$. It is SPD.

Convergence of the residual in each case is shown in Figure 4. Each of the tests was started from the initial vector $u_0 = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}^T$. The results show the dynamic method 3 is not sensitive to the scaling of the eigenvalues which vary in each of the examples. The results also show a better rate of convergence with β_+ than with β_-, but at the cost of a potentially extended preasymptotic regime. For `ss1` (Matrix 9) shown in the center plot, the dynamic method shows some initial oscillations but does not suffer for the extended asymptotic regime that β_{opt} and β_+ experience.

5 | DYNAMIC MOMENTUM METHOD FOR INVERSE ITERATION

As an immediate extension of Algorithm 3, this section explores the application of the dynamic momentum method to accelerate the shifted inverse power iteration. The shifted inverse iteration is a well-known and powerful technique in the numerical solution of eigenvalue problems. A review of the method including its history, theory and implementation can be found in Reference 26. By appropriately choosing shifting parameters, the inverse iteration with shift can be used to identify any targeted eigenpair. When a good approximation of the targeted eigenvalue is available, the method is remarkably efficient. As the inverse iteration with shift σ is equivalent to applying the power iteration on the matrix $(A-\sigma I)^{-1}$ (with the same eigenvectors as those of A), the analysis carried out in Section 3.1 is directly applicable to the Algorithm 5 below.

Each step of the inverse iteration involves solving a linear system. For a fixed shift, one can perform a factorization of the shifted matrix before the iterative loop to save some computational cost. Unlike updating the shift to attain faster convergence, applying a momentum acceleration does not require a re-factorization of the matrix. As shown below, the momentum accelerated algorithm substantially reduces the number of iterations to convergence, particularly for sub-optimal shifts. Presumably, the use of a sub-optimal shift indicates the user does not have a good approximation of the target eigenvalue, by which the user is unlikely to have a good approximation of the second eigenvalue of the shifted system. Hence the automatic assignment of the extrapolation parameter β_k is essential for this method to be practical. Fortunately, as seen below, the proposed method with dynamic β_k is comparable to or outperforms the optimal parameter in each case tested. Numerical experiments below illustrate the improved efficiency, particularly with the dynamic strategy.

In our implementation of DMPOW in this section we terminated the preliminary iterations in the deflation stage when the target (second) eigenvalue achieved a given relative tolerance, that is in the notation of Reference 1, $|(\mu_j - \mu_{j-1})/\mu_j| < 10^{-n}$. We show results using $n = 1, 2, 4$. We implemented DMPOW as a shifted inverse iteration as follows, referring to the implementation given in Reference 1 (algorithm 1): We replaced the multiplication by A in line 2 with a multiplication by $(A-\sigma I)^{-1}$, implemented as a solve of the LU-factored system, and the multiplication by $A-P$ in line 6 with a multiplication by $(A-\sigma I)^{-1}$, again implemented as a solve of the factored system, and a multiplication by P. We replaced the multiplication by A in line 8 to compute the Rayleigh quotient corresponding to the second eigenvalue as a multiplication by $A-\sigma I$. We did this rather than multiplying by $(A-\sigma I)^{-1}$ to reduce the number of system solves per iteration from three to two, with little to no effect on the total iteration count.

Just as Algorithm 3 requires two preliminary power iterations, the dynamic momentum strategy for the inverse iteration requires two preliminary inverse iterations. For tests in this section, we ran iterations to a residual tolerance of 10^{-15} or a maximum of 2000 iteration. In this section we also numerically verify the stability of the extrapolation parameter β_k as shown in Lemma 3 and Remark 3.

The dynamically accelerated version of Algorithm 4 follows.

In Table 3 we show results for accelerating the inverse iteration used to recover the largest eigenvalue of the matrix $A = \text{diag}(1000 : -1 : 1)$. We test shifts $\sigma = \{999.75, \ldots, 1064\}$, chosen with increasing distance from the target eigenvalue $\lambda_1 = 1000$ to see how much a suboptimal shift can be made up for with the extrapolation.

We see the dynamic momentum method and the "optimal" fixed momentum parameter give the best performance, with the dynamic method converging in fewer iterations as the shift increases away from the target eigenvalue. The performance of the DMPOW iteration is intermediate between the base inverse iteration and the dynamical momentum method.

Algorithm 4. Inverse power iteration

Choose v_0 and shift σ, set $h_0 = \|v_0\|$ and $x_0 = h_0^{-1} v_0$
Compute $(A-\sigma I) = LU$ ▷ Compute LU factors of $A-\sigma I$
Solve $Ly = x_0$ and $Uv_1 = y$
for $k \geq 0$ **do**
  Set $h_{k+1} = \|v_{k+1}\|$ and $x_{k+1} = h_{k+1}^{-1} v_{k+1}$
  Solve $Ly = x_{k+1}$ and $Uv_{k+2} = y$
  Set $v_{k+1} = (v_{k+2}, x_{k+1})$ and $d_{k+1} = \|v_{k+2} - v_{k+1} x_{k+1}\|$
  STOP if $\|d_{k+1}\| <$ `tol`
end for

Algorithm 5. Dynamic momentum for inverse iteration

Do two iterations of Algorithm 4 ▷ $k = 0, 1$
Set $r_2 = \min\{d_2/d_1, 1\}$
for $k \geq 2$ **do** ▷ $k \geq 2$
  Set $\beta_k = v_k^2 r_k^2/4$
  Set $u_{k+1} = v_{k+1} - (\beta_k/h_k)x_{k-1}$
  Set $h_{k+1} = \|u_{k+1}\|$ and $x_{k+1} = h_{k+1}^{-1} u_{k+1}$
  Solve $Ly = x_{k+1}$ and $Uv_{k+2} = y$
  Set $v_{k+1} = (v_{k+2}, x_{k+1})$ and $d_{k+1} = \|v_{k+2} - v_{k+1}x_{k+1}\|$
  Update $\rho_k = \min\{d_{k+1}/d_k, 1\}$ and$r_{k+1} = 2\rho_k/(1 + \rho_k^2)$
  STOP if $\|d_{k+1}\| <$ `tol`
end for

TABLE 3 Number of system solves using the shifted inverse iteration to find the largest eigenvalue with fixed and dynamic momentum for the matrix $A = \text{diag}(n : -1 : 1)$, with $n = 1000$.

σ	$\beta = 0$	dynβ_k	β_{opt}	$\beta_{DM}(10^{-1})$	$\beta_{DM}(10^{-2})$	$\beta_{DM}(10^{-4})$
999.75	33	**21**	23 (4.44e-1)	25 (4.60e-1)	29 (4.45e-1)	32 (4.44e-1)
1000.25	23	**17**	**18** (1.60e-1)	21 (1.59e-1)	22 (1.60e-1)	25 (1.60e-1)
1000.5	32	**23**	**22** (1.11e-1)	26 (1.11e-1)	26 (1.11e-1)	31 (1.11e-1)
1001	49	33	**29** (6.25e-2)	32 (6.39e-2)	32 (6.39e-2)	40 (6.25e-2)
1004	142	55	**52** (1.00e-2)	56 (1.02e-2)	58 (1.03e-2)	80 (1.00e-2)
1016	478	**88**	95 (8.65e-4)	215 (7.43e-4)	134 (8.52e-4)	120 (8.73e-4)
1064	1691	**163**	175 (5.92e-5)	841 (4.55e-5)	525 (5.47e-5)	239 (5.93e-5)

Note: The optimal fixed extrapolation parameters $\beta_{opt} = 1/(4(\lambda_2 - \sigma)^2)$ are shown after the number of solves in the β_{opt} column, and the dynamically chosen parameter β_k is set as in Algorithm 5. The last three columns, $\beta_{DM}(10^{-n})$, $n = 1, 2, 4$ contain the number of system solves for DMPOW, where the preliminary deflation stage is terminated after the target eigenvalue reaches the relative tolerance of 10^{-n}. The parameter is shown after the number of solves. Each iteration is started from $v_0 = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}^T$, and run to a residual tolerance of 10^{-15}.

Compared with the base Algorithm 4, Algorithm 5 with dynamically chosen β_k not only reduces the number of iterations for each given shift, it also achieves a better iteration count with shifts more than twice as far away from the target eigenvalue. This shows Algorithm 5 reduces the sensitivity to the shift in the standard inverse iteration.

Figure 5 shows the extrapolation parameters β_k for three different shifts σ as shown in Table 3. The left plot shows $\{\beta_k\}$ for $\sigma = 1001$. Denoting the eigenvalues of $(A - \sigma I)^{-1}$ as $\{\tilde{\lambda}_i\}, i = 1, \ldots, n$, we have $\tilde{\lambda}_1 = -1$, and $\tilde{\lambda}_2 = -1/2$ so that $r = 0.5$. In this case β_k oscillates above and below β_{opt}. For $\beta_k < \beta_{opt}$, $\tilde{\lambda}_2$ is non-oscillatory by (12), but each of the eigenvalues with $\tilde{\lambda}^2/4 < \beta$ is oscillatory, and each decays at a slightly faster rate than $\tilde{\lambda}_2$ by (14). As per Lemma 3, the approximation of r by the detected convergence rate ρ_k is stable, but the oscillations in r_{k+1} are not necessarily damped with respect to the detected ρ_k. For $\beta_k > \beta_{opt}$, all subdominant modes are oscillatory and decay at the same rate by (14), and the stability of r_{k+1} with respect to ρ_k still holds.

For the center plot in Figure 5, $\sigma = 1016$ so that $r \approx 0.94$; and in the right plot $\sigma = 1064$ so that $r \approx 0.98$.

In both of these cases, Lemma 3 shows that r_{k+1} is stable with respect to ρ_k, and the difference between r_k and r is damped in comparison to the difference between the detected ρ_k and ρ; and moreso in the plot on the right.

Notably for the center figure with $r \approx 0.94$, β_k converges to β_{opt} to within 10^{-4}, and for the right plot with $r \approx 0.98$, β_k converges to β_{opt} to within 10^{-5}.

This demonstrates how r_k approaches r as r approaches one, as described in Lemma 3.

In Table 4 we show the results of a similar experiment to recover the smallest eigenvalue of the matrix $A = \text{diag}(1000 : -1 : 1)$. We test shifts $\sigma = \{1.25, 0.75, 0.5, 0, -1, -4, -8, -16, -32\}$, a range of shifts with increasing distance from the target eigenvalue $\lambda_n = 1$. Our results are similar to the largest eigenvalue case of Table 3.

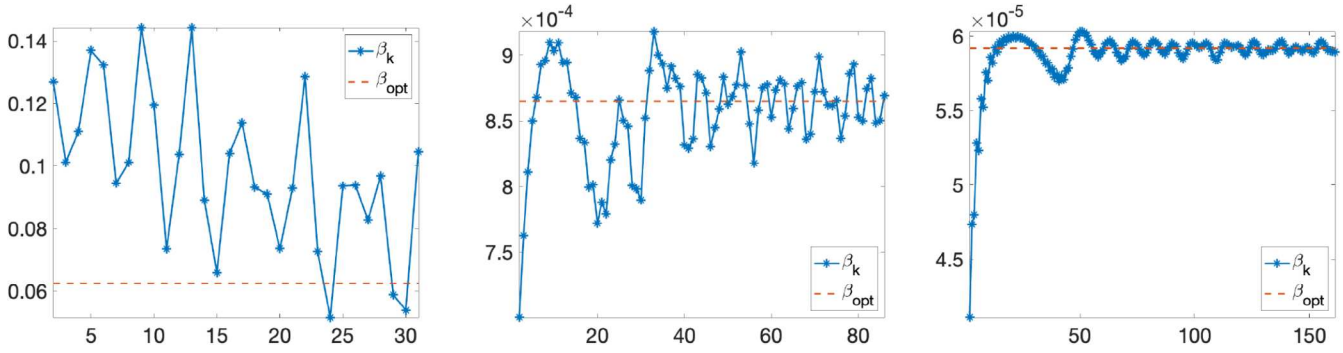

FIGURE 5 Behavior of β_k with respect to β_{opt} for representative examples from Table 3, illustrating that β_k stabilizes closer to β_{opt} in agreement with Lemma 3 as $r \to 1$. Left: $\sigma = 1001$, for which $r = 0.5$. Center: $\sigma = 1016$, for which $r \approx 0.94$. Right: $\sigma = 1064$ for which $r \approx 0.98$.

TABLE 4 Number of system solves using the shifted inverse iteration to find the smallest eigenvalue with fixed and dynamic momentum for the matrix $A = \text{diag}(n : -1 : 1)$, with $n = 1000$.

σ	$\beta = 0$	dynβ_k	β_{opt}	$\beta_{DM}(10^{-1})$	$\beta_{DM}(10^{-2})$	$\beta_{DM}(10^{-4})$
1.25	33	21	23 (4.44e-1)	25 (4.60e-1)	29 (4.45e-1)	32 (4.44e-1)
0.75	23	17	17 (1.60e-1)	21 (1.59e-1)	22 (1.60e-1)	25 (1.60e-1)
0	49	33	29 (6.25e-2)	32 (6.39e-2)	32 (6.39e-2)	40 (6.25e-2)
−1	81	46	39 (2.78e-2)	41 (2.89e-2)	42 (2.86e-2)	57 (2.78e-2)
−4	171	58	57 (6.94e-3)	59 (6.97e-3)	64 (7.10e-3)	89 (6.95e-3)
−8	286	70	74 (2.50e-3)	119 (2.32e-3)	78 (2.52e-3)	120 (2.50e-3)
−16	505	91	97 (7.72e-4)	229 (6.58e-4)	143 (7.57e-4)	124 (7.79e-4)
−32	922	123	130 (2.16e-4)	444 (1.73e-4)	288 (2.03e-4)	177 (2.17e-4)

Note: The optimal fixed extrapolation parameters $\beta_{opt} = 1/(4(\lambda_{n-1} - \sigma)^2)$ are shown after the number of solves in the β_{opt} column, and the dynamically chosen parameter β_k is set as in Algorithm 5. The last three columns, $\beta_{DM}(10^{-n})$, $n = 1, 2, 4$ contain the number of system solves for DMPOW, where the preliminary deflation stage is terminated after the target eigenvalue reaches the relative tolerance of 10^{-n}. The parameter is shown after the number of solves. Each iteration is started from $v_0 = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}^T$, and run to a residual tolerance of 10^{-15}.

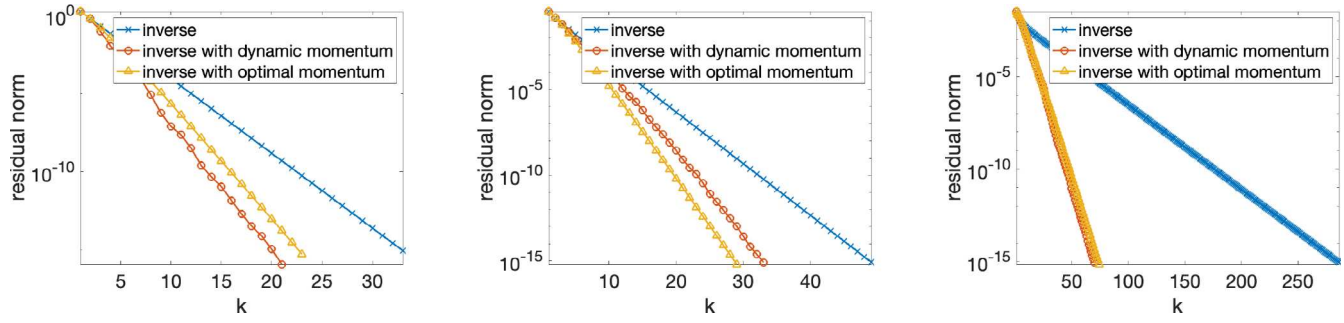

FIGURE 6 Residual convergence for representative examples from Table 4, illustrating the improvement in convergence for the optimal and dynamic momentum acceleration for a variety of shifts. Left: $\sigma = 1.25$, for which $r = 1/3$. Center: $\sigma = 0$, for which $r = 0.5$. Right: $\sigma = -8$ for which $r = 0.9$.

We see the ratio between the number of iterations in the dynamic method and the inverse iteration decreases as the the shift increases. For instance, as the shift ranges between 0 and -32, the corresponding ratio between the number of dynamic momentum iterations and inverse iterations without momentum decreases monotonically from 0.67 to 0.13. In each case tested, the dynamic method is either comparable to or better than the momentum method with optimal shift, and outperforms all of the DMPOW iterations. Figure 6 shows convergence plots for $\sigma = 1.25, \sigma = 0$ and $\sigma = -8$, providing a visualization of the improved convergence rates from the dynamic Algorithm 5.

These examples illustrate the gain in convergence from this practical and low-cost acceleration method.

6 | CONCLUSION

In this article, we introduced and analyzed a one-step extrapolation method to accelerate convergence of the power iteration for real, diagonalizable matrices, and proved convergence to the dominant eigenpair with acceleration in the symmetric case.

The method is based on the momentum method for the power iteration introduced in Reference 22, and requires a single matrix-vector multiply per iteration. Unlike the method of Reference 22 and other recent variants such as Reference 1, the presently introduced technique gives a dynamic update of the key extrapolation parameter at each iteration, and does not require any a priori knowledge of the spectrum.

We first reviewed some results on the analysis of a static method of the form introduced in Reference 22, by considering the power iteration applied to an augmented matrix. Our analysis goes beyond that shown in the original article, revealing that the augmented matrix is defective for the optimal parameter choice, which explains why slower convergence is expected in the preasymptotic regime. We then analyzed our dynamic method showing both stability of the dynamic extrapolation parameter and convergence of the method.

In the last two sections we numerically demonstrated the efficiency of the introduced dynamic Algorithm 3 as applied to power and inverse iterations. We demonstrated that Algorithm 3 often outperforms the original static method with the optimal parameter choice as given in Algorithm 2. We further showed Algorithm 3 performs favorably in comparison to the method of Reference 1, which generally accelerates the power iteration but does not exceed the performance of Algorithm 2. Finally, we showed that the introduced dynamic method is a useful tool to accelerate inverse power iterations, and can be used to converge in as few iterations as having a shift twice as close to the target eigenvalue, and without significant additional computational complexity. Future work will include the development of an analogous method applied to (preconditioned) Krylov subspace projection methods as in References 11–13,21 to efficiently recover multiple eigenpairs.

ACKNOWLEDGEMENTS

CA and SP are supported in part by NSF DMS-2045059 (CAREER). This material is based upon work supported by the NSF under DMS-1929284 while SP was in residence at the Institute for Computational and Experimental Research in Mathematics in Providence, RI, during the Numerical PDEs: Analysis, Algorithms and Data Challenges Program. SP would like to thank Prof. Nilima Nigam for many interesting discussions that led to the formulation of this work.

CONFLICT OF INTEREST STATEMENT

The authors have no conflicts of interest to declare.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Christian Austin https://orcid.org/0000-0001-5510-4024
Sara Pollock https://orcid.org/0000-0001-7896-350X

REFERENCES

1. Rabbani T, Jain A, Rajkumar A, Huang F. Practical and fast momentum-based power methods. In: Bruna J, Hesthaven J, Zdeborova L, editors. Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference. Volume 145. New York:PMLR; 2022. p. 721–56.

2. Brezinski C, Redivo-Zaglia M. The PageRank vector: Properties, computation, approximation, and acceleration. SIAM J Matrix Anal Appl. 2006;28:551–75.
3. Golub GH, Greif C. An Arnoldi-type algorithm for computing page rank. BIT Numer Math. 2006;46:759–71.
4. Haveliwala TH, Kamvar SD, Klein D, Manning CD, Golub GH. Computing PageRank Using Power Extrapolation. Stanford, CA: Stanford University; 2003. Technical report SCCM03-02.
5. Kamvar S. Numerical Algorithms for Personalized Search in Self-organizing Information Networks. Princeton: Princeton University Press; 2010.
6. Sidi A. Approximation of largest eigenpairs of matrices and applications to Pagerank computation.
7. Babuška I, Osborn J. Eigenvalue problems, Finite element methods (Part 1). In: Ciarlet PG, Lions JL, editors. Handbook of Numerical Analysis. Volume II. Amsterdam: North-Holland; 1991. p. 641–787.
8. Golub GH, Van Loan CF. Matrix Computations. 3rd ed. Baltimore, MD, USA: Johns Hopkins University Press; 1996.
9. Hu Q-Y, Wen C, Huang T-Z, Shen Z-L, Gu X-M. A variant of the Power-Arnoldi algorithm for computing PageRank. J Comput Appl Math. 2021;381:113034.
10. Duersch JA, Shao M, Yang C, Gu M. A robust and efficient implementation of LOBPCG. SIAM J Sci Comput. 2018;40(5):C655–76.
11. Knyazev AV. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. SIAM J Sci Comput. 2001;23(2):517–41.
12. Golub GH, Ye Q. An inverse free preconditioned Krylov subspace method for symmetric generalized eigenvalue problems. SIAM J Sci Comput. 2002;24(1):312–34.
13. Quillen P, Ye Q. A block inverse-free preconditioned Krylov subspace method for symmetric generalized eigenvalue problems. J Comput Appl Math. 2010;233(5):1298–313. Special Issue Dedicated to William B. Gragg on the Occasion of His 70th Birthday.
14. Davidson ER. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. J Comput Phys. 1975;17(1):87–94.
15. Hochstenbach M, Notay Y. The Jacobi-Davidson method. GAMM-Mitteilungen. 2006;29(2):368–82.
16. Wilkinson JH. The Algebraic Eigenvalue Problem. Oxford: Clarendon Press; 1965.
17. Brezinski C. Computation of the eigenelements of a matrix by the ε-algorithm. Linear Algebra Appl. 1975;11(1):7–20.
18. Sidi A. Vector extrapolation methods with applications to solution of large systems of equations and to PageRank computations. Comput Math Appl. 2008;56:1–24.
19. Bai Z-Z, Wu W-T, Muratova GV. The power method and beyond. Appl Numer Math. 2021;1;164:29–42.
20. Nigam N, Pollock S. A simple extrapolation method for clustered eigenvalues. Numer Algorithms. 2022;89:115–43.
21. Pollock S, Scott LR. Extrapolating the Arnoldi algorithm to improve eigenvector convergence. Int J Numer Anal Model. 2021;18(5):712–21.
22. Sa CD, He B, Mitliagkas I, Ré C, Xu P. Accelerated stochastic power iteration. Proc Mach Learn Res. 2019;84:58–67.
23. Polyak BT. Some methods of speeding up the convergence of iteration methods. USSR Comput Math Math Phys. 1964;45:1–17.
24. Saad Y. Numerical methods for large eigenvalue problems. Vol 66. Revised ed. Philadelphia, PS, USA: Society for Industrial and Applied Mathematics; 2011.
25. Davis TA, Hu Y. The University of Florida sparse matrix collection. ACM Trans Math Softw. 2011;38(1):1–25.
26. Ipsen ICF. Computing an eigenvector with inverse iteration. SIAM Rev. 1997;39(2):254–91.

How to cite this article: Austin C, Pollock S, Zhu Y. Dynamically accelerating the power iteration with momentum. Numer Linear Algebra Appl. 2024;e2584. https://doi.org/10.1002/nla.2584