Journal of Scientific Computing (2022) 93:63
https://doi.org/10.1007/s10915-022-02025-0

Triangularized Orthogonalization-Free Method for Solving Extreme Eigenvalue Problems

Weiguo Gao[1,2] · **Yingzhou Li**[1] · **Bichen Lu**[3]

Received: 30 November 2021 / Revised: 25 July 2022 / Accepted: 5 September 2022

Abstract
A novel orthogonalization-free method together with two specific algorithms is proposed to address extreme eigenvalue problems. On top of gradient-based algorithms, the proposed algorithms modify the multicolumn gradient such that earlier columns are decoupled from later ones. Locally, both algorithms converge linearly with convergence rates depending on eigengaps. Momentum acceleration, exact linesearch, and column locking are incorporated to accelerate algorithms and reduce their computational costs. We demonstrate the efficiency of both algorithms on random matrices with different spectrum distributions and matrices from computational chemistry.

Keywords Eigenvalue problem · Orthogonalization-free · Iterative eigensolver · Full configuration interaction

Mathematics Subject Classification (2020) 65CF52

1 Introduction

This paper proposes a novel triangularized orthogonalization-free method (TriOFM) for solving extreme eigenvalue problems. Given a symmetric matrix A, the extreme eigenvalue problem is defined as,

$$AU = U\Lambda, \tag{1}$$

✉ Yingzhou Li
yingzhouli@fudan.edu.cn

Weiguo Gao
wggao@fudan.edu.cn

Bichen Lu
balu18@fudan.edu.cn

[1] School of Mathematical Sciences, Fudan University, Shanghai 200433, China

[2] Shanghai Aritifical Intelligence Laboratory, Shanghai 200433, China

[3] Shanghai Center for Mathematical Sciences, Shanghai 200438, China

Published online: 22 October 2022

where $A \in \mathbb{R}^{n\times n}$, $A^\top = A$, $\Lambda \in \mathbb{R}^{p\times p}$ is a diagonal matrix with A's p smallest eigenvalues on the diagonal in ascending order, and the columns of U are the corresponding eigenvectors. The proposed methods target some specific applications in computational chemistry, in which areas smallest eigenpairs are desired as the ground-state and low-lying excited-states. Though we introduce algorithms for p smallest eigenpairs, all algorithms in this paper can be adapted to compute p largest eigenpairs. Besides computational chemistry, solving extreme eigenvalue problems is a fundamental computational step in a wide range of applications, including but not limited to the principal component analysis, dimension reduction, spectral clustering, etc.

In this paper, we specifically concern extreme eigenvalue problems with two properties:

(i) Orthogonalization of the iteration variable X is not permitted;
(ii) Eigenvectors are sparse vectors.

At least two important applications from computational chemistry, linear-scaling density functional theory (DFT) [30] and full configuration interaction (FCI) [16] for low-lying excited states, admit these two properties. In linear-scaling DFT, the number of desired eigenpairs is of the same order as the problem size. The orthogonalization step then scales cubically, which is not permitted in linear-scaling DFT. Regarding the sparsity, linear-scaling DFT adopts localized basis sets, and the eigenvectors therein are indeed sparse [4, 37]. Although FCI also admits the above two properties, it has its own unique feature. In FCI, the desired number of eigenpairs p is usually a small constant, e.g., $p = 5, 10$. While the size of the matrix n grows factorially as the system increases. For example, considering a single water molecule with 48 spin-orbitals and 10 electrons, the matrix is of size $\sim 10^8$. Due to the factorially increasing matrix size, orthogonalization is too expensive in both computational and memory costs to be applied in practice. Sparsity is also an important feature of FCI. Thanks to the two-body interaction feature of the electrons, the matrix is extremely sparse. Regarding the water molecule example, each column of the matrix has roughly 10^4 nonzero entries. The eigenvectors of the ground-state and low-lying excited-states are sparse. FCI is the motivating application of this work, and hence some of our algorithm designs would prefer FCI to DFT.

1.1 Related Work

For linear symmetric eigenvalue problems as (1), there are many classical eigensolvers from textbooks of numerical linear algebra. Readers are referred to [12] for references. In electronic structure calculation, variants of classical eigensolvers, like Davidson [8], conjugate gradient(CG) [15, 32], locally optimal block preconditioned conjugate gradient method (LOBPCG) [17], projected preconditioned conjugate gradient (PPCG) [38], Chebyshev filtering [1, 2, 20, 45], pole expansion [25, 33], Rayleigh quotient based optimization methods [13, 21, 35], are widely used in the self-consistent field iteration in DFT. All these methods are related to Krylov subspace. A recent software ELSI [42, 43] provides an interface to many of these eigensolvers for DFT calculation.

Besides Krylov subspace methods, another family of methods view the symmetric eigenvalue problem as a constrained optimization problem and solve it using either first-order or second-order methods [7, 9, 14, 36, 41, 44]. These methods usually target more general objective functions with orthonormal constraints. However, the linear eigenvalue problem is always one of their important applications. Since the feasible set of the orthonormality constraint is the Stiefel manifold, these methods are also known as manifold optimization methods. They take either the Euclidean gradient or Riemannian gradient step with certain

strategies in calculating the stepsize. A retraction or projection step is needed to maintain the feasibility of the iteration variable. Recently, in order to enhance the parallelizability, the retraction step is avoided through either the augmented Lagrangian method [10, 40] or extend gradient [6].

Linear symmetric eigenvalue problems can also be written as an unconstrained optimization problem. The most well-known one is minimizing the Rayleigh quotient, which can be generalized to the multicolumn case. Another two unconstrained optimization problems are

$$\min_{X\in\mathbb{R}^{n\times p}} \left\| A + XX^\top \right\|_{\mathrm{F}}^2, \tag{Obj1}$$

and

$$\min_{X\in\mathbb{R}^{n\times p}} \operatorname{tr}\left(\left(2I - X^\top X\right) X^\top A X\right), \tag{Obj2}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm and tr $(\cdot)$ denotes the trace operation. (Obj1) has been adopted to address the extreme eigenvalue problems arsing from several areas [19, 24, 27], including FCI [23, 39]. (Obj2) is widely known as the orbital minimization method (OMM) [5, 28–31], which is popular in the area of (linear-scaling) DFT. More details about (Obj1) and (Obj2)are deferred to Sect. 2.

For all methods aforementioned in this section, some of them are orthogonalization-free, and some of them converge to eigenvectors directly. Nevertheless, none of them is an orthogonalization-free method converging to eigenvectors directly.

1.2 Contribution

In this paper, a novel iterative method named triangularized orthogonalization-free method (TriOFM) is proposed, which is orthogonalization-free and converges to eigenvectors directly. The method is inspired by the unconstrained optimization methods (denoted as OFM throughout this paper). In OFM, the updating direction is the gradient of the objective function, whereas, in TriOFM, the updating direction is a triangularized version of the gradient, which decouples earlier columns from later ones. When the gradient is triangularized in TriOFM, the updating direction is no longer a gradient of any function. Hence the underlying dynamic is not a conservative flow. The analysis is then very different from traditional analysis in optimization. In this paper, we triangularize two objective functions, i.e., (Obj1) and (Obj2), and obtain two iterative algorithms named TriOFM-(Obj1) and TriOFM-(Obj2) respectively.

The convergence analysis of TriOFM-(Obj1) is carried out in detail. First, we discuss the stable and unstable fixed points of our algorithm. We then provide local convergence analysis with a convergence rate. The rate is carried out through a careful analysis of the accumulated error term. All analyses can be extended to TriOFM-(Obj2), and we state the corresponding theorems without detailed proof. Global convergence can be also be established. We leave the detail in a companion paper [11]. Notice that the global convergence is given without a rate.

After the analyses, we propose a few techniques to accelerate the convergence and reduce the computational cost. Conjugate gradient direction and linesearch strategies are proposed to accelerate both algorithms. These two techniques were also applied in OFM which are tailored for TriOFM in this paper. While, in OFM, the locking technique is not feasible due to the existence of the orthogonalization step. In TriOFM, the locking technique is incorporated to reduce the computational cost.

Table 1 Notations

Notation	Explanation
n	The size of the matrix
q	The number of negative eigenvalues of the matrix
p	The number of desired eigenpairs and $p \leq q$
A	The n-by-n symmetric matrix
Λ	A diagonal matrix with diagonal entries being eigenvalues of A in increasing ordering
λ_i	The i-th smallest eigenvalue of A
Λ_i	The first i-by-i principal submatrix of Λ
U	An orthogonal matrix satisfying $U^\top AU = \Lambda$
u_i	The eigenvector of A corresponding to λ_i
U_i	The first i columns of U
ρ	The 2-norm of A, *i.e.*, $\rho = \|A\|_2$
$X^{(t)}$	An n-by-p matrix denoting the iteration variable at t-th iteration
$x_i^{(t)}$	The i-th column of $X^{(t)}$
$X_i^{(t)}$	The first i columns of $X^{(t)}$
$f_1(X)$, $f_2(X)$	The objective function in (Obj1), (Obj2)
$\nabla f_1(X)$, $\nabla f_2(X)$	The gradient of $f_1(X)$, $f_2(X)$
α	The stepsize
e_i	The i-th standard basis vector [a]

[a] A vector of length n with one on the i-th entry and zero elsewhere

Finally, numerical examples are provided to demonstrate the effectiveness of TriOFM. All suggested techniques are first explored on random matrices and then applied to two practical examples, one from DFT and another one from FCI. In both practical examples, we observe that the proposed framework achieves both the orthogonalization-free and converging to eigenvectors properties while not losing much efficiency of the computational cost compared with their original OFM counterparts.

1.3 Organization

In the rest of this paper, Sect. 2 provides detailed introductions to both (Obj1) and (Obj2) with an analysis of the energy landscape. Section 3 introduces TriOFM and its two iterative algorithms, TriOFM-(Obj1) and TriOFM-(Obj2), in detail. The convergence analysis is carried out in Sect. 4. Algorithmic techniques are proposed in Sect. 5. In Sect. 6, all algorithms are numerically explored on random matrices and matrices from practice. Finally, Sect. 7 concludes the paper with a discussion on future directions.

2 Preliminary

We introduce OFM eigensolvers based on (Obj1) and (Obj2) in this section. Notations used throughout the paper are summarized in Table 1, which would be used without further explanation.

The orthogonalization step is a key step in most traditional eigensolvers, *e.g.*, power method, QR iteration, Lanczos, etc. While, the orthogonalization step is difficult to be efficiently parallelized on modern computer architectures, *i.e.*, distributed-memory computers and GPUs. OFM eigensolvers, in contrast, do not involve the orthogonalization step and only require matrix-matrix multiplication, which is one of the most parallel efficient operations. Hence OFM is plausible for solving large problems in a massively parallel environment.

Given that A is symmetric, $A + XX^\top$ in (Obj1) is the residual of a symmetric low-rank approximation. In [27], (Obj1) is shown to be equivalent to a trace-penalty minimization model with a specific penalty parameter. In both [24, 27], the energy landscape of (Obj1) has been analyzed. (Obj1) does not have any spurious local minimum, and all local minima are global minima. We rephrase and summarize the analysis result as follows under the aforementioned assumption that A has at least p negative eigenvalues.

Theorem 1 *All **stationary points** of* (Obj1) *are of form* $X = U_q\sqrt{-\Lambda_q}SP$ *and all **local minima** are of form* $X = U_p\sqrt{-\Lambda_p}Q$, $S \in \mathbb{R}^{q\times q}$ *is a diagonal matrix with diagonal entries being 0 or 1 (at most p 1s),* $P \in \mathbb{R}^{q\times p}$ *and* $Q \in \mathbb{R}^{p\times p}$ *are unitary matrices. Further, any local minimum is also a global minimum.*

On the other hand, when $p > q$, stationary points are exactly of the same form, whereas local minima need to be updated as $X = U_q\sqrt{-\Lambda_q}Q$ with $Q \in \mathbb{R}^{q\times p}$ having orthogonal columns. For almost all chemistry problems, the assumption $p \le q$ holds in practice. Hence we stick to this assumption for (Obj1) throughout the paper to simplify our presentation.

The intuition behind (Obj2) is more complicated. There are two ways to motivate the objective function: the approximated inverse and the Lagrange multiplier.

A multi-column version of Rayleigh quotient admits $\mathrm{tr}\left(\left(X^\top X\right)^{-1}X^\top AX\right)$, which could also be an objective function option for OFM. Assuming the spectrum of $X^\top X$ is bounded by one, we have the Neumann series expansion of the inversion and the first order approximation as,

$$\left(X^\top X\right)^{-1} = \left(I - \left(I - X^\top X\right)\right)^{-1} = \sum_{k=0}^{\infty}\left(I - X^\top X\right)^k \approx 2I - X^\top X.$$

Substituting the approximation into the multi-column Rayleigh quotient leads to (Obj2).

Another way to motivate (Obj2) is via the Lagrange multiplier method. Lagrangian function for eigenvalue problem admits

$$\mathcal{L}(X, \Xi) = \mathrm{tr}\left(X^\top AX\right) - \mathrm{tr}\left(\Xi\left(X^\top X - I\right)\right),$$

where Ξ denotes the Lagrange multiplier. The first order optimality condition leads to an expression for the Lagrange multiplier, $\Xi = X^\top AX$. Substituting this expression into the Lagrangian function gives (Obj2).

Previous work [28] characterizes the energy landscape of (Obj2). (Obj2) does not have any spurious local minimum. The theorem therein is rephrased as follows.

Theorem 2 *Let A be a symmetric negative semi-definite matrix. All **stationary points** of* (Obj2) *are of form* $X = USP$ *and all **local minima** are of form* $X = U_pQ$, *where* $S \in \mathbb{R}^{n\times n}$ *is a diagonal matrix with diagonal entries being 0 or 1 (at most p 1s),* $P \in \mathbb{R}^{n\times p}$ *and* $Q \in \mathbb{R}^{p\times p}$ *are unitary matrices. Further, any local minimum is also a global minimum.*

Notice that the matrix A in (Obj2) must be negative semi-definite. Otherwise, X can be scaled eigenvectors corresponding to the positive eigenvalues, and (Obj2) is unbounded

from below. For eigenvalue problems, the matrix can be shifted to be negative semi-definite. Comparing to (Obj1), an extra step is needed to estimate the shift, and shifting is needed every iteration.

Based on the analysis of the energy landscape of both (Obj1) and (Obj2), any algorithm avoiding saddle points converges to the global minimum. Such algorithms include but are not limited to regular gradient descent [18], conjugate gradient descent, stochastic gradient descent [3, 22], etc. Using the notation defined in Table 1, gradients of (Obj1) and (Obj2) are

$$\nabla f_1(X) = 4AX + 4XX^\top X, \tag{2}$$

and

$$\nabla f_2(X) = 4AX - 2XX^\top AX - 2AXX^\top X, \tag{3}$$

respectively. The gradient descent iterations are defined as,

$$X^{(t+1)} = X^{(t)} - \alpha \left(AX^{(t)} + X^{(t)} \left(X^{(t)} \right)^\top X^{(t)} \right), \tag{4}$$

and

$$X^{(t+1)} = X^{(t)} - \alpha \left(2AX^{(t)} - X^{(t)} \left(X^{(t)} \right)^\top AX^{(t)} - AX^{(t)} \left(X^{(t)} \right)^\top X^{(t)} \right), \tag{5}$$

where the constant is absorbed into the stepsize. Unfortunately, the Hessian of both (Obj1) and (Obj2) are unbounded from above. The valid set for the choice of the stepsize over the entire domain is empty. For both (Obj1) and (Obj2), one can find a bounded domain such that iterations are guaranteed to stay within the domain. Then Hessians are bounded over the domain and the valid set for the stepsize is non-empty.

3 Triangularized Optimization Eigensolvers

We propose triangularized orthogonalization-free methods (TriOFM) as eigensolvers based on (Obj1) and (Obj2), which are denoted as TriOFM-(Obj1) and TriOFM-(Obj2).

Our goal, as mentioned in Sect. 1 is to find p extreme eigenpairs with two properties: (i). orthogonalization of X is not permitted; (ii). eigenvectors are sparse vectors. Optimizing (Obj1) and (Obj2) almost achieves the first required property except requiring an extra step of applying a Rayleigh-Ritz step which distinguishes eigenvectors from an eigenspace, while the second property is not taken into consideration. Due to the existence of the arbitrary orthogonal matrix Q, the iterations (4) and (5) converge to points with destroyed sparsity in the original eigenvectors. Adding ℓ_1 penalty to (Obj2) [29] is proposed to achieve the sparsity as much as possible in DFT problems, which is not likely to be applicable to FCI problems.

Another way of explicitly getting the eigenpairs rather than a point in the eigenspace is to solve the single-column version of (Obj1) or (Obj2) recursively. For example, first, we solve the single column version of either (Obj1) or (Obj2) for $A_1 = A$ and obtain the smallest eigenpair λ_1 and u_1. Then we apply the method to $A_2 = A_1 - \lambda_1 u_1 u_1^\top$ and obtain λ_2 and u_2. At k-th time, the method is applied to $A_k = A_{k-1} - \lambda_{k-1} u_{k-1} u_{k-1}^\top = A - \sum_{i=1}^{k-1} \lambda_i u_i u_i^\top$ and λ_k and u_k are computed. Such a recursive procedure has two drawbacks. First, single column operations are composed of BLAS1-level and BLAS2-level operations, which are not as efficient as BLAS3-level operations in modern computer architecture. The second

drawback is the lack of efficient representation of the transformed matrix A_k. The sparsity in A plays a crucial role in designing algorithms for FCI problems. While, A_k is not as sparse as A in almost all cases.

Although the aforementioned recursive procedure is not ideal for our problems, it inspires TriOFM-(Obj1) and TriOFM-(Obj2). We will first motivate and derive TriOFM-(Obj1). Then TriOFM-(Obj2) can be derived in an analogy way.

In the above recursive procedure, the single column version of (4) is applied to $A_k = A - \sum_{i=1}^{k-1} \lambda_i u_i u_i^\top$. Notice that if the column-by-column procedure is applied, the convergent point of x_i is $\pm\sqrt{-\lambda_i}u_i$. Hence, A_k can be viewed as the summation of A with the outer product of convergent vector of $x_1, x_2, \ldots, x_{k-1}$. If we assume all columns update together, and the single column version of (4) is applied to a closed approximation of A_k, *i.e.*, $A_k \approx \widetilde{A}_k = A + \sum_{i=1}^{k-1} x_i x_i^\top$, then we obtain the following iterative schemes,

$$\begin{aligned}
x_1^{(t+1)} =& x_1^{(t)} - \alpha \left(A x_1^{(t)} + x_1^{(t)} \left(x_1^{(t)} \right)^\top x_1^{(t)} \right), \\
x_2^{(t+1)} =& x_2^{(t)} - \alpha \left(A x_2^{(t)} + x_1^{(t)} \left(x_1^{(t)} \right)^\top x_2^{(t)} + x_2^{(t)} \left(x_2^{(t)} \right)^\top x_2^{(t)} \right), \\
& \cdots \\
x_k^{(t+1)} =& x_k^{(t)} - \alpha \left(A x_k^{(t)} + \sum_{i=1}^{k} x_i^{(t)} \left(x_i^{(t)} \right)^\top x_k^{(t)} \right), \\
& \cdots .
\end{aligned} \tag{6}$$

Using matrix notations, the above iterative schemes admit the following representation,

$$X^{(t+1)} = X^{(t)} - \alpha \left(A X^{(t)} + X^{(t)} \mathrm{triu} \left(\left(X^{(t)} \right)^\top X^{(t)} \right) \right), \tag{7}$$

where triu $(\cdot)$ denote the upper triangular part of a given matrix. The key difference between (4) and (7) is that the gradient is modified as,

$$g_1(X) = AX + X \mathrm{triu} \left(X^\top X \right). \tag{8}$$

Unfortunately, g_1 in (8) is not a gradient of any energy function. Hence, instead of analyzing the stationary points of the energy function, we analyze the fixed points of (7) in Theorem 3.

Theorem 3 *All **fixed points** of (7) are of form $X = U_q \sqrt{-\Lambda_q} P S$, where $\sqrt{\cdot}$ is applied entry-wise, $P \in \mathbb{R}^{q \times p}$ is the first p columns of an arbitrary q-by-q permutation matrix, and $S \in \mathbb{R}^{p \times p}$ is a diagonal matrix with diagonal entries being 0 or ± 1. Within these points all **stable fixed points** are of form $X = U_p \sqrt{-\Lambda_p} D$, where $D \in \mathbb{R}^{p \times p}$ is a diagonal matrix with diagonal entries being ± 1. Others are **unstable fixed points**.*

Proof All fixed points of (7) satisfy $g_1(X) = 0$ with $g_1(X)$ being a n-by-p matrix. We prove the theorem by induction. Here we introduce notations in addition to that in Table 1: $P_i \in \mathbb{R}^{q \times i}$ is the first i columns of an arbitrary q-by-q permutation matrix, $S_i \in \mathbb{R}^{i \times i}$ is a diagonal matrix with diagonal entries being 0 or ± 1, and $D_i \in \mathbb{R}^{i \times i}$ is a diagonal matrix with diagonal entries being ± 1.

Consider the first column of $g_1(X) = 0$,

$$A x_1 + x_1 x_1^\top x_1 = 0, \tag{9}$$

where $x_1^\top x_1$ is a non-negative scalar. When $x_1 = 0$, (9) naturally holds. When $x_1 \neq 0$, x_1 must be a scalar multiple of an eigenvector of A and $x_1^\top x_1$ is the negative of the corresponding eigenvalue, which must be negative. Hence $X_1 = x_1$ is of the form, $X_1 = U_q\sqrt{-\Lambda_q}P_1S_1$.

Now assume the first i columns of X obeys $X_i = U_q\sqrt{-\Lambda_q}P_iS_i$. Then the $(i+1)$-th column of $g_1(X) = 0$ obeys

$$0 = Ax_{i+1} + X_iX_i^\top x_{i+1} + x_{i+1}x_{i+1}^\top x_{i+1} = \widetilde{A}x_{i+1} + x_{i+1}x_{i+1}^\top x_{i+1}, \tag{10}$$

where $\widetilde{A} = A + X_iX_i^\top = A + U_q\sqrt{-\Lambda_q}P_iS_i^2P_i^\top\sqrt{-\Lambda_q}U_q^\top$. $\widetilde{A}$ is the original matrix A zeroing out a few eigenvalues corresponding to the selected columns in P_i with ± 1 in S_i. Applying the similar analysis as in the case of (9) to (10), we conclude that X_{i+1} is of the form, $X_{i+1} = U_q\sqrt{-\Lambda_q}P_{i+1}S_{i+1}$.

Since $q \geq p$, we have a sufficient number of negative eigenpairs to be added to X. The induction can be processed until $i = p$, and we obtain the expression for all fixed points as in the theorem.

The stabilities of fixed points are determined by the spectrum of their Jacobian matrices of g_1, *i.e.*, $\mathrm{D}g_1(X)$. Since both $g_1(X)$ and X are matrices, the Jacobian is a 4-way tensor, which is unfolded as a matrix here. In order to avoid over complicated index in subscripts, we denote the matrix $g_1(X)$ as G. Notation G_{ij} and x_{ij} denote the (j,i)-th element of G and X respectively. Then the Jacobian matrix is written as a p-by-p block matrix,

$$\mathrm{D}g_1(X) = \mathrm{D}G = \begin{pmatrix} J_{11} & \cdots & J_{1p} \\ \vdots & \ddots & \vdots \\ J_{p1} & \cdots & J_{pp} \end{pmatrix}, \tag{11}$$

with block J_{ij} being,

$$J_{ij} = \begin{pmatrix} \frac{\partial G_{i1}}{\partial x_{j1}} & \cdots & \frac{\partial G_{in}}{\partial x_{j1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial G_{i1}}{\partial x_{jn}} & \cdots & \frac{\partial G_{in}}{\partial x_{jn}} \end{pmatrix}. \tag{12}$$

Notice that the i-th column of G, $G_i = Ax_i + X_iX_i^\top x_i$, is independent of $x_{i+1}, \ldots x_p$, which means $J_{ij} = 0$ for $i < j$. $\mathrm{D}g_1(X)$ is a block upper triangular matrix. The spectrum of $\mathrm{D}g_1(X)$ is determined by the spectrum of J_{ii} for $i = 1, 2, \ldots, p$. Through a multivariable calculus, we obtain the explicit expression for J_{ii},

$$J_{ii} = A + X_iX_i^\top + x_i^\top x_i I + x_i x_i^\top. \tag{13}$$

We first show the stability of the fixed points of form $X = U_p\sqrt{-\Lambda_p}D$. Substituting these points into (13), we have,

$$J_{ii} = A - U_i\Lambda_iU_i^\top - \lambda_i I - u_i\lambda_i u_i^\top. \tag{14}$$

Since λ_i is negative and strictly smaller than all eigenvalues of $A - U_i\Lambda_iU_i^\top$, J_{ii} is strictly positive definite for all $i = 1, 2, \ldots, p$. Therefore we have all eigenvalues of $\mathrm{D}g_1\left(U_p\sqrt{-\Lambda_p}D\right)$ are strictly positive and $X = U_p\sqrt{-\Lambda_p}D$ are stable fixed points.

Next, we show the instability of the rest fixed points. If X is a fixed points but not of the form $U_p\sqrt{-\Lambda_p}D$, then there exist indices s such that $x_s^\top u_s = 0$. Denote s as the first such index. Substituting this point into J_{ss} and computing the bilinear form of J_{ss} with respect to u_s, we have,

$$u_s^\top J_{ss}u_s = \lambda_s - x_s^\top x_s < 0, \tag{15}$$

where the inequality comes from the fact that x_s is zero or corresponds to eigenvalues greater than λ_s. Hence the Jacobian matrix has negative eigenvalues. Hence these points are unstable fixed points. □

Algorithm TriOFM-(Obj1) is the pseudocode for (7). The choice of the stepsize is unspecified, which will be revealed in later sections.

Algorithm 1 TriOFM-(Obj1)/TriOFM-(Obj2)

Input: a symmetric matrix A, an initial point $X^{(0)}$
$t = 0$
while not converged **do**

$$g^{(t)} = \begin{cases} AX^{(t)} + X^{(t)}\mathrm{triu}\left(\left(X^{(t)}\right)^\top X^{(t)}\right) & \text{(TriOFM-(Obj1))} \\ 2AX^{(t)} - AX^{(t)}\mathrm{triu}\left(\left(X^{(t)}\right)^\top X^{(t)}\right) & \\ \quad - X^{(t)}\mathrm{triu}\left(\left(X^{(t)}\right)^\top AX^{(t)}\right) & \text{(TriOFM-(Obj2))} \end{cases}$$

Choose a stepsize $\alpha^{(t)}$
$X^{(t+1)} = X^{(t)} - \alpha^{(t)} g^{(t)}$
$t = t + 1$

There is another way to understand the iterative scheme. The column with a smaller index is decoupled from columns with larger indices. For example, the iterative scheme of x_1 is independent of all later columns. For the second column x_2, the iterative scheme on x_2 is the same as the second column in the 2-column version of (Obj1). Recursively applying the idea, we also reach Algorithm TriOFM-(Obj1).

Similar idea can be applied to solve (Obj2) as well. We notice that there are two terms in (5) coupling columns together, *i.e.*, $AX^{(t)}\left(X^{(t)}\right)^\top X^{(t)}$ and $X^{(t)}\left(X^{(t)}\right)^\top AX^{(t)}$. Using the decoupling idea, we can replace the $\left(X^{(t)}\right)^\top X^{(t)}$ and $\left(X^{(t)}\right)^\top AX^{(t)}$ by their upper triangular parts and result the following iterative scheme,

$$X^{(t+1)} = X^{(t)} - \alpha\left(2AX^{(t)} - AX^{(t)}\mathrm{triu}\left(\left(X^{(t)}\right)^\top X^{(t)}\right) - X^{(t)}\mathrm{triu}\left(\left(X^{(t)}\right)^\top AX^{(t)}\right)\right). \tag{16}$$

Comparing to (5), the gradient is modified as,

$$g_2(X) = 2AX - AX\mathrm{triu}\left(X^\top X\right) - X\mathrm{triu}\left(X^\top AX\right). \tag{17}$$

The fixed points of (16) can be analyzed in a similar way. We summarize the properties in Theorem 4 and leave the proof in Appendix A.

Theorem 4 *Let A be a negative definite matrix. All **fixed points** of (16) are of form $X = UPS$ and all **stable fixed points** are of form $X = U_p D$, where $P \in \mathbb{R}^{n\times p}$ is the first p columns of an arbitrary n-by-n permutation matrix, $S \in \mathbb{R}^{p\times p}$ is a diagonal matrix with diagonal entries being 0 or ± 1, and $D \in \mathbb{R}^{p\times p}$ is a diagonal matrix with diagonal entries being ± 1.*

Algorithm TriOFM-(Obj2) illustrates the pseudocode for (16) and the choice of the stepsize is also deferred to later sections.

We claim a few advantages of Algorithm TriOFM-(Obj1) and Algorithm TriOFM-(Obj2) over other related methods. First, both algorithms converge to the eigenvectors or their scaled ones without mixing them. Hence the sparsity of the eigenvectors is preserved. Although we do not benefit from the sparsity during the iteration in Algorithm TriOFM-(Obj1) and Algorithm TriOFM-(Obj2) directly, we expect that the coordinate descent methods would benefit from the sparsity and achieve fast convergence and small memory cost for FCI problems. Second, the orthogonalization step is totally removed, which makes the algorithm friendly to parallel computing. Third, all cubic scaling operations can be processed through BLAS3-level routines. Algorithms, therefore, benefit from the memory hierarchy of modern computer architecture.

Although we only propose Algorithm TriOFM-(Obj1) and Algorithm TriOFM-(Obj2) and analyze their convergence in this paper, the idea of TriOFM can be applied to a wide range of algorithms to remove the redundancy introduced by the rotation invariance. The key point here is decoupling each column from later columns while ensuring that the iterative scheme for a column remains the same as solving the multicolumn version of the objective function. The question of where and how TriOFM can be applied is open.

4 Convergence Analysis

In this section, we focus on the local convergence of the proposed TriOFM-(Obj1). A similar result holds for TriOFM-(Obj2) as well. We denote the set of stable fixed points as $\mathcal{X}^*$ and a stable fixed point as $X^* \in \mathcal{X}^*$. Further, x_i^* denotes the i-th column of X^* and X_i^* denotes the first i columns of X^*. The conclusion of the local convergence to X^* is given in Theorem 5, whereas Lemmas 1 and 2 provide per-iteration bound on the residual of the first column and later columns, respectively. Finally, the rate of local convergence is given in Corollary 1.

Lemma 1 *Assume the stepsize α satisfies $\alpha < \min_{1\le j\le p}\left\{\frac{1}{4\rho}, \frac{1}{\lambda_{j+1}-\lambda_j}\right\}$. Let $\varepsilon_1^{(t)}$ be the error of the first column after the t-th iteration, $\varepsilon_1^{(t)} = x_1^{(t)} - x_1^*$. If $\left\|\varepsilon_1^{(t)}\right\| \le \frac{\lambda_2-\lambda_1}{8\sqrt{-\lambda_1}}$, then $\left\|\varepsilon_1^{(t+1)}\right\| \le \left(1-\alpha\frac{\lambda_2-\lambda_1}{2}\right)\left\|\varepsilon_1^{(t)}\right\|$.*

Proof Without loss of generality, we assume that A is a diagonal matrix. For simplicity, we drop the iteration index superscript and use $x_1 = x_1^{(t)}, \varepsilon_1 = \varepsilon_1^{(t)}, \widetilde{x}_1 = x_1^{(t+1)}$ and $\widetilde{\varepsilon}_1 = \varepsilon_1^{(t+1)}$ instead. Further we denote the first column of X^* as $v_1 = x_1^*$. From Theorem 3, we have $v_1^\top v_1 = -\lambda_1$ and $v_1 v_1^\top = -\lambda_1 u_1 u_1^\top = -\lambda_1 e_1 e_1^\top$.

Based on the iterative scheme on the first column, *i.e.*, $\widetilde{x}_1 = x_1 - \alpha A x_1 - \alpha x_1^\top x_1 x_1$, we have,

$$\begin{aligned}\widetilde{\varepsilon}_1 = \widetilde{x}_1 - v_1 &= \varepsilon_1 - \alpha A(v_1+\varepsilon_1) - \alpha\left(v_1+\varepsilon_1\right)^\top\left(v_1+\varepsilon_1\right)\left(v_1+\varepsilon_1\right)\\ &= \left((1+\alpha\lambda_1)I - \alpha A - 2\alpha v_1 v_1^\top\right)\varepsilon_1 - \alpha v_1\left\|\varepsilon_1\right\|^2 - 2\alpha v_1^\top\varepsilon_1\varepsilon_1 - \alpha\left\|\varepsilon_1\right\|^2\varepsilon_1.\end{aligned} \tag{18}$$

The assumption on α implies that $1+\alpha\lambda_1 \pm \alpha\lambda_i > 0$ holds for all i. Hence, the 2-norm of the diagonal matrix $(1+\alpha\lambda_1)I - \alpha A - 2\alpha v_1 v_1^\top$ admits

$$\left\|(1+\alpha\lambda_1)I - \alpha A - 2\alpha v_1 v_1^\top\right\| = 1 + \alpha\lambda_1 - \alpha\lambda_2. \tag{19}$$

The norm of $\widetilde{\varepsilon}_1$ is bounded as,

$$\|\widetilde{\varepsilon}_1\| \leq (1+\alpha\lambda_1-\alpha\lambda_2)\|\varepsilon_1\| + 3\alpha\sqrt{-\lambda_1}\|\varepsilon_1\|^2 + \alpha\|\varepsilon_1\|^3 \leq \left(1-\alpha\frac{\lambda_2-\lambda_1}{2}\right)\|\varepsilon_1\|, \tag{20}$$

where the second inequality adopts the fact $\|\varepsilon_1\| \leq \frac{\lambda_2-\lambda_1}{8\sqrt{-\lambda_1}}$. □

In Lemma 1, we prove that the error of x_1 converges linearly in a neighborhood of the stable fixed point. Now we move on to the multicolumn case. For the i-th column x_i, we have the following lemma.

Lemma 2 *Assume the stepsize α satisfies $\alpha < \min_{1\leq j\leq p}\left\{\frac{1}{4\rho}, \frac{1}{\lambda_{j+1}-\lambda_j}\right\}$. Let $\varepsilon_i^{(t)}$ be the error of the i-th column after the t-th iteration, $\varepsilon_i^{(t)} = x_i^{(t)} - x_i^*$. If $\left\|\varepsilon_j^{(t)}\right\| \leq \frac{\lambda_{j+1}-\lambda_j}{8\sqrt{-\lambda_j}}$ for all $j \leq i$, then we have $\left\|\varepsilon_i^{(t+1)}\right\| \leq \left(1-\alpha\frac{\lambda_{i+1}-\lambda_i}{2}\right)\left\|\varepsilon_i^{(t)}\right\| + \alpha\sum_{j=1}^{i-1}\frac{2\|A\|^2}{\sqrt{\lambda_j\lambda_i}}\left\|\varepsilon_j^{(t)}\right\|$.*

Proof Similarly, we drop the superscript in the proof. We denote the i-th column of X^* as $v_i = x_i^*$. From Theorem 3, we have $v_i^\top v_i = -\lambda_i$ and $v_i v_i^\top = -\lambda_i u_i u_i^\top$ for $i = 1, \ldots, p$. Based on the iterative scheme, $\widetilde{x}_i = x_i - \alpha A x_i - \alpha\sum_{j=1}^{i} x_j x_j^\top x_i$, there is

$$\begin{aligned}
\widetilde{\varepsilon}_i = \widetilde{x}_i - v_i =& \varepsilon_i - \alpha A(v_i+\varepsilon_i) - \alpha\sum_{j=1}^{i}\left(v_j v_j^\top + \varepsilon_j\varepsilon_j^\top + v_j\varepsilon_j^\top + \varepsilon_j v_j^\top\right)(v_i+\varepsilon_i) \\
=& \left((1+\alpha\lambda_i)I - \alpha A - \alpha v_i v_i^\top - \alpha\sum_{j=1}^{i} v_j v_j^\top\right)\varepsilon_i - \alpha\sum_{j=1}^{i-1} v_j v_i^\top \varepsilon_j \\
& - \alpha\sum_{j=1}^{i}\varepsilon_j\varepsilon_j^\top v_i - \alpha\sum_{j=1}^{i}\left(v_j\varepsilon_j^\top + \varepsilon_j v_j^\top\right)\varepsilon_i - \alpha\sum_{j=1}^{i}\varepsilon_j\varepsilon_j^\top\varepsilon_i.
\end{aligned} \tag{21}$$

The norm of the prefactor of ε_i can be bounded as,

$$\left\|(1+\alpha\lambda_i)I - \alpha A - \alpha v_i v_i^\top - \alpha\sum_{j=1}^{i} v_j v_j^\top\right\| \leq 1 + \alpha\lambda_i - \alpha\lambda_{i+1}. \tag{22}$$

The norm of (21) is bounded as,

$$\begin{aligned}
\|\widetilde{\varepsilon}_i\| \leq& (1+\alpha\lambda_i-\alpha\lambda_{i+1})\|\varepsilon_i\| + \alpha\sum_{j=1}^{i-1}\sqrt{\lambda_i\lambda_j}\left\|\varepsilon_j\right\| + \alpha\sqrt{-\lambda_i}\sum_{j=1}^{i}\left\|\varepsilon_j\right\|^2 \\
& + 2\alpha\sum_{j=1}^{i}\sqrt{-\lambda_j}\left\|\varepsilon_j\right\|\|\varepsilon_i\| + \alpha\sum_{j=1}^{i}\left\|\varepsilon_j\right\|^2\|\varepsilon_i\| \\
=& (1-\alpha\lambda_{i+1}+\alpha\lambda_i)\|\varepsilon_i\| + 3\alpha\sqrt{-\lambda_i}\|\varepsilon_i\|^2 + \alpha\|\varepsilon_i\|^3 \\
& + \alpha\sum_{j=1}^{i-1}\left[\sqrt{\lambda_i\lambda_j}\left\|\varepsilon_j\right\| + \sqrt{-\lambda_i}\left\|\varepsilon_j\right\|^2 + 2\sqrt{-\lambda_j}\left\|\varepsilon_j\right\|\|\varepsilon_i\| + \left\|\varepsilon_j\right\|^2\|\varepsilon_i\|\right].
\end{aligned} \tag{23}$$

Denote $\Delta_j = \lambda_{j+1} - \lambda_j$ as the j-th eigengap. Using the assumption $\|\varepsilon_j\| \le \frac{\Delta_j}{8\sqrt{-\lambda_j}}$ for all $1 \le j \le i$, we have

$$\|\widetilde{\varepsilon}_i\| \le \left(1 - \alpha\frac{\Delta_i}{2}\right)\|\varepsilon_i\| + \alpha\sum_{j=1}^{i-1}\left(\sqrt{\lambda_i\lambda_j} + \sqrt{-\lambda_i}\frac{\Delta_j}{8\sqrt{-\lambda_j}} + \sqrt{\lambda_j}\frac{\Delta_i}{4\sqrt{-\lambda_i}} + \frac{\Delta_i\Delta_j}{64\sqrt{\lambda_i\lambda_j}}\right)\|\varepsilon_j\|, \tag{24}$$

where the first term $\left(1 - \alpha\frac{\Delta_i}{2}\right)\|\varepsilon_i\|$ is bounded in the same way as that in Lemma 1. The second term can further be controlled recursively,

$$\begin{aligned}\|\widetilde{\varepsilon}_i\| &\le \left(1 - \alpha\frac{\Delta_i}{2}\right)\|\varepsilon_i\| + \alpha\sum_{j=1}^{i-1}\frac{64\lambda_i\lambda_j - 8\lambda_i\Delta_j - 16\lambda_j\Delta_i + \Delta_i\Delta_j}{64\sqrt{\lambda_i\lambda_j}}\|\varepsilon_j\| \\ &\le \left(1 - \alpha\frac{\Delta_i}{2}\right)\|\varepsilon_i\| + \alpha\sum_{j=1}^{i-1}\frac{116\|A\|^2}{64\sqrt{\lambda_i\lambda_j}}\|\varepsilon_j\| \\ &\le \left(1 - \alpha\frac{\Delta_i}{2}\right)\|\varepsilon_i\| + \alpha\sum_{j=1}^{i-1}\frac{2\|A\|^2}{\sqrt{\lambda_i\lambda_j}}\|\varepsilon_j\|.\end{aligned} \tag{25}$$

Here $\|A\|$ is adopted to simplify the final bound since all λs are controlled by $\|A\|$. □

Lemma 1 is a linear convergence result directly whereas Lemma 2 is slightly different from the standard linear convergence result. In Theorem 5 we investigate the extra term $2\alpha\frac{\|A\|^2}{\sqrt{\lambda_j\lambda_i}}\|\varepsilon_j\|$ and find out that the overall local convergence is linear.

Theorem 5 *Assume the stepsize α satisfies $\alpha < \min_{j\in[1,p]}\left\{\frac{1}{4\rho}, \frac{1}{\lambda_{j+1}-\lambda_j}\right\}$. Let $\varepsilon_i^{(t)}$ be the error of the i-th column after the t-th iteration, $\varepsilon_i^{(t)} = x_i^{(t)} - x_i^*$. If $\left\|\varepsilon_j^{(0)}\right\| \le \frac{\lambda_{j+1}-\lambda_j}{8\sqrt{-\lambda_j}}$ for all $j \le i$, then for any $i = 1, \ldots, p$ there exists a polynomial $C_i(t)$ of degree $i-1$ such that*

$$\left\|\varepsilon_i^{(t)}\right\| \le C_i(t)r_i^t. \tag{26}$$

where $r_i = 1 - \frac{\alpha}{2}\min_{j\in[1,i]}\{\lambda_{j+1} - \lambda_j\}$.

Proof The theorem is proved by induction. First, Lemma 1 shows that $\left\|\varepsilon_1^{(t)}\right\|$ satisfies (26) for $C_1 = \left\|\varepsilon_1^{(0)}\right\|$. Given $i \le p$, we assume that the theorem holds for all $j < i$. We further assume that all polynomials in the theorem are non-decreasing. Denoting $a_j = \alpha\frac{2\|A\|^2}{\sqrt{\lambda_j\lambda_i}}$, the inequality in Lemma 2 can be further bounded as,

$$\left\|\varepsilon_i^{(t)}\right\| \le r_i\left\|\varepsilon_i^{(t-1)}\right\| + \sum_{j=1}^{i-1}a_j\left\|\varepsilon_j^{(t-1)}\right\| \le r_i\left\|\varepsilon_i^{(t-1)}\right\| + C_{max}(t)r_{i-1}^{t-1}, \tag{27}$$

where $C_{max}(t) = \sum_{j=1}^{i-1}a_jC_j(t)$ and the relationship $r_1 \le \cdots \le r_{i-1}$ is used so that all r_js are bounded by r_{i-1}. Notice that for each j, a_j is positive and $C_j(t)$ is a non-decreasing polynomial of degree $j-1$. $C_{max}(t)$ is then a non-decreasing polynomial of degree $i-2$.

Since the inequality above holds for all $t \geq 1$, we apply it repeatedly and obtain,

$$\left\|\varepsilon_i^{(t)}\right\| \leq r_i^t \left\|\varepsilon_i^{(0)}\right\| + \sum_{k=0}^{t-1} r_i^{t-1-k} C_{max}(k) r_{i-1}^k \leq \left(\left\|\varepsilon_i^{(0)}\right\| + \frac{t}{r_i} C_{max}(t)\right) r_i^t = C_i(t) r_i^t, \tag{28}$$

where $C_i(t) = \left\|\varepsilon_i^{(0)}\right\| + \frac{t}{r_i} C_{max}(t)$ is a non-decreasing polynomial of degree $i-1$. Hence the theorem is proved. □

Theorem 3 states that there is a set of stable fixed points of TriOFM-(Obj1). Next Corollary 1 shows that the iterative scheme TriOFM-(Obj1) locally has linear convergence to the set of stable fixed points. We define the distance from a point to a set as, $\|X - \mathcal{X}^*\|_{\mathrm{F}} = \min_{X^* \in \mathcal{X}^*} \|X - X^*\|_{\mathrm{F}}$.

Corollary 1 *Assume the stepsize α satisfies $\alpha < \min_{j\in[1,p]} \left\{\frac{1}{4\rho}, \frac{1}{\lambda_{j+1}-\lambda_j}\right\}$. Let $\delta^{(t)}$ be the distance from the stable fixed points after the t-th iteration, $\delta^{(t)} = \left\|X^{(t)} - \mathcal{X}^*\right\|_{\mathrm{F}}$. If $\delta^{(0)} \leq \min_{j\in[1,p]} \frac{\lambda_{j+1}-\lambda_j}{8\sqrt{-\lambda_j}}$, then there exists a polynomial $C(t)$ of degree $p-1$ such that $\delta^{(t)} \leq C(t)r^t$, where $r = 1 - \frac{\alpha}{2} \min_{j\in[1,p]}\{\lambda_{j+1} - \lambda_j\}$.*

Proof We first notice that for any two distinct points in $\mathcal{X}^*$, the smallest distance in F-norm is $2\sqrt{-\lambda_p}$, which is greater than twice initial error $\delta^{(0)} \leq \min_{j\in[1,p]} \frac{\lambda_{j+1}-\lambda_j}{8\sqrt{-\lambda_j}}$. Hence for any initial point, it can only be attracted by one stable fixed point. By the definition of $\delta^{(t)}$ and Theorem 5, we have,

$$\delta^{(t)} = \sqrt{\sum_{i=1}^{p} \left\|\varepsilon_i^{(t)}\right\|^2} \leq \sqrt{\sum_{i=1}^{p} \left(C_i(t) r_i^t\right)^2} \leq \sum_{i=1}^{p} C_i(t) r_p^t = C(t) r^t \tag{29}$$

where $C(t) = \sum_{i=1}^{p} C_i(t)$ is a polynomial of degree $p-1$ and the second inequality is due to the non-negativity of $C_i(t)$. □

We shall notice that the estimation $\delta^{(t)} \leq C(t)r^t$ satisfies $\lim_{t\to\infty} \frac{C(t+1)r^{t+1}}{C(t)r^t} = r \lim_{t\to\infty} \frac{C(t+1)}{C(t)} = r$, which is the definition of linear convergence. Hence we claim the iterative scheme TriOFM-(Obj1) locally converges linearly to stable fixed points. A similar proof procedure can be applied to show the local linear convergence for TriOFM-(Obj2).

Remark 1 When deriving the exponential term r^t in Theorem 5, as in (20) and (24) we sacrifice half of constant in the leading term in the convergence rate to have a control on the higher order terms of ε. Hence we achieve the rate $r_i = 1 - \frac{\alpha}{2} \min_{j\in[1,i]}\{\lambda_{j+1} - \lambda_j\}$ in Theorem 5. However, if we adopt a smaller portion of the leading term to control the higher order term, which asymptotically could be viewed as igonring the higer order term, we are able to get an asymptotic convergence rate as fast as $r_i \approx 1 - \alpha \min_{j\in[1,i]}\{\lambda_{j+1} - \lambda_j\}$. The drawback is that such a bound also asymptotically shrink the local convergence domain size to zero. From our numerical resutls as in Sect. 6.1.1, we observed that the asymptotic convergence rate agrees well with estimated empirical convergence rate.

Remark 2 We noticed that, for general eigensolvers, the convergence rate of each eigenvector is shift-invariant because it only relies on the spectrum gap. However, things are different for

the objective functions in this paper, since the spectrum of (4) and (5) can be regarded as the spectrum of A combined with an extra 0. So, if we shift the spectrum of A far away from 0, the algorithms in our paper would converge slower.

In addition to local convergence, TriOFM-(Obj1) and TriOFM-(Obj2) also converge globally. In our companion paper [11], the global convergence of TriOFM-(Obj1) and is proved in detail, which is rephrased in Theorem 6. Similar global and local convergence results of TriOFM-(Obj2) [26] have been proved in [26], following the idea in [11] and this paper. Both results rely on the stable manifold theorem for discrete dynamical systems.

Theorem 6 *If the initial point* $X^{(0)} = \left(x_1^{(0)}\ x_2^{(0)}\ \cdots\ x_p^{(0)}\right)$ *satisfies* $\left\|x_i^{(0)}\right\| \leq R_i$ *for all* $1 \leq i \leq p$, *where* $R_i = 2^{i-1}\sqrt{3\rho}$ *and the stepsize satisfies* $\alpha \leq \frac{1}{10R_p^2}$, *then the fixed stepsize version of Algorithm* TriOFM-(Obj1) *converges to* $\mathcal{X}^*$ *for all initial points besides a set of measure zero.*

Comparing the local and global convergence, as in Corollary 1 and Theorem 6, the restrictions on stepsizes are different, *i.e.*, the stepsize of global convergence is much smaller. Such a small stepsize is needed in global convergence to overcome the unbounded Lipschitz constant of the underlying objective function but would be more flexible in practice, especially when the stepsize is chosen in a sophisticated way, as stated in Sect. 5.2.

5 Implementation Details

In previous sections, we introduce TriOFM algorithms based on the gradient descent method with a constant stepsize and prove their convergence properties. TriOFM can be regarded as a modified gradient descent method. In this section, we explore traditional accelerating techniques for gradient methods and adapt them to TriOFM. Such techniques include momentum acceleration, stepsize choices, and column locking.

5.1 Momentum Acceleration

Momentum is a widely-used technique to accelerate gradient descent methods. In traditional gradient descent methods, with the help of momentum, the oscillatory trajectory could be smoothed, and the convergence rate depends on the square root of the condition number rather than the condition number.

Momentum method, instead of moving along the gradient direction directly, moves along with an accumulation of gradient directions with a discounting parameter $\beta \in (0, 1]$, *i.e.*,

$$V^{(t)} = \beta g\left(X^{(t)}\right) + (1-\beta)V^{(t-1)}, \tag{30}$$

where $V^{(t)}$ denotes the accumulated direction and g is the gradient. Then the iteration moves along $V^{(t)}$ with a stepsize α, *i.e.*, $X^{(t+1)} = X^{(t)} - \alpha V^{(t)}$. Since V is a linear combination of gradient directions, an explicit way to generalize it to the triangularized method is to replace the gradient g by our triangularized direction function either g_1 or g_2. Then we obtain the momentum accelerated algorithms for TriOFM-(Obj1) and TriOFM-(Obj2).

Importantly, such a modification will not change the dependency among columns of X, where 'dependency' describes generating one column $x_i^{(t+1)}$ from $x_i^{(t)}$ depends on another

column $x_j^{(t)}$ or not. In TriOFM, previous columns should be independent of later columns, and we require such property to be maintained in enhanced algorithms based on TriOFM. With this momentum enabled, the first i columns remain the same as the algorithm applied on $X = \begin{pmatrix} x_1 & x_2 & \cdots & x_i \end{pmatrix}$. Any column of X still depends only on columns on its left throughout the iterations. However, for momentum methods, choosing an efficient momentum parameter β is an art.

Similarly, we can adopt the idea of conjugate gradient (CG) [12] to triangularized algorithms as well. CG is a momentum method with adaptive momentum parameters and hence choosing β is avoided. CG is widely applied to solve both linear and nonlinear problems. The success of nonlinear CG in solving eigenvalue problems have already been demonstrated in OMM [5]. A typical non-linear CG method is the Polak-Reeves CG (PR-CG) [34], which adopts the following steps per iteration in a single-vector setting:

$$\begin{aligned} \beta^{(t)} =& \frac{\left(g\left(x^{(t)}\right) - g\left(x^{(t-1)}\right)\right)^\top g\left(x^{(t)}\right)}{g\left(x^{(t-1)}\right)^\top g\left(x^{(t-1)}\right)}, \\ v^{(t)} =& -g\left(x^{(t)}\right) + \beta^{(t)} v^{(t-1)}, \\ x^{(t+1)} =& x^{(t)} + \alpha v^{(t)}. \end{aligned} \tag{31}$$

In a multi-vector setting, *i.e.*, the iteration variable is a matrix, the formula for $\beta^{(t)}$ could be extended. However, the multi-vector version for $\beta^{(t)}$ mixes all columns together and destroys the column dependency of TriOFM.

A more favorable choice of $\beta^{(t)}$ for TriOFM is to use different $\beta^{(t)}$s for different columns, which is called the columnwise CG throughout this paper. The parameter for the i-th column, denoted as $\beta_i^{(t)}$, is calculated as the single-vector setting with $x_i^{(t)}$ and applied to update $x_i^{(t)}$. The corresponding algorithm for TriOFM is summarized as Algorithm 2. In Algorithm 2, $g_i^{(t)}$ and $v_i^{(t)}$ denote the i-th column of $G^{(t)}$ and $V^{(t)}$ respectively.

Algorithm 2 Columnwise CG for TriOFM

Input: symmetric matrix A, initial point $X^{(0)}$, stepsize α
$G^{(0)} = g(X^{(0)})$
$V^{(0)} = -G^{(0)}$
$X^{(1)} = X^{(0)} + \alpha V^{(0)}$
$t = 1$
while not converged **do**
  $G^{(t)} = g(X^{(t)})$
  for $i = 1, 2, \ldots, p$ **do**
    $\beta_i^{(t)} = \frac{\left(g_i^{(t)} - g_i^{(t-1)}\right)^\top g_i^{(t)}}{\left(g_i^{(t-1)}\right)^\top g_i^{(t-1)}}$
    $v_i^{(t)} = -g_i^{(t)} + \beta_i^{(t)} v_{i-1}^{(t)}$
  $X^{(t+1)} = X^{(t)} + \alpha V^{(t)}$
  $t = t + 1$

As a remark, there is another way in computing the parameter $\beta_i^{(t)}$s, *i.e.*, $\beta_i^{(t)}$ is calculated using the multi-vector version of (31) with $X_i^{(t)}$. The dependencies among columns are preserved. However, the calculation must be conducted carefully to avoid increasing the computational cost.

5.2 Stepsizes

In previous sections, we describe algorithms with a constant stepsize to simplify the presentation. However, we find that a linesearch strategy could significantly outperform the constant stepsize. In this section, we introduce an exact linesearch strategy as the suggested stepsize strategy.

Since both (Obj1) and (Obj2) are quartic polynomials of X, the exact linesearch can be calculated through minimizing quartic polynomials. Minimizing a quartic polynomial with a positive leading coefficient is equivalent to solve a cubic polynomial. Taking (Obj1) as an example, the cubic polynomial is,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\alpha} f_1(X+\alpha V) =& \operatorname{tr}\left(V^\top \nabla f_1(X+\alpha V)\right)\\ =& \alpha^3 \operatorname{tr}\left(\left(V^\top V\right)^2\right) + 3\alpha^2 \operatorname{tr}\left(V^\top V X^\top V\right)\\ &+ \alpha \operatorname{tr}\left(V^\top A V + \left(V^\top X\right)^2 + V^\top X X^\top V + V^\top V X^\top X\right)\\ &+ \operatorname{tr}\left(V^\top A X\right) + \operatorname{tr}\left(V^\top X X^\top X\right).\end{aligned} \tag{32}$$

Solving the expression above would give possibly one, two, or three real roots. The best stepsize can be selected among real roots through a basic analysis [24]. Similar calculation and analysis can also be carried out for (Obj2). We omit the details here.

However, the stepsize in (32) does not work for TriOFM. Consider such a case for example. If X is in the space spanned by the smallest eigenpairs but not the stable fixed point, *i.e.*, $X = U_p\sqrt{-\Lambda_p}Q$ for Q being a non-diagonal unitary matrix, then X is already a global minimum of (Obj1), so minimizing $f_1(X)$ from any direction V with stepsize α, as in (32), results in $\alpha = 0$. But such X is not the eigenvector we want throughout this paper. This example shows that the above linesearch strategy is not working properly for TriOFM and we need to find a different strategy for the stepsize.

Notice that the exact linesearch solves $\operatorname{tr}\left(V^\top \nabla f(X+\alpha V)\right) = 0$ for the stepsize α. However, TriOFM adopts g_1 or g_2 rather than ∇f_1 or ∇f_2, which means the iteration is not consistency with the linesearch (32). The columnwise stepsize strategy is as follows. First, consider the stepsize for x_1. We solve two identical equations, $v_1^\top g(x_1+\alpha v_1) = 0$ and $v_1^\top \nabla f(x_1+\alpha v_1) = 0$, to obtain the stepsize. Now we consider the stepsize α_i for x_i. We can solve either $\operatorname{tr}\left(V_i^\top \nabla f(X_i+\alpha_i V_i)\right) = 0$ or $\operatorname{tr}\left(V_i^\top g(X_i+\alpha_i V_i)\right) = 0$ for α_i. The former is the same as (32) with X and V replaced by X_i and V_i respectively. The later can be expressed as again a cubic polynomial of α_i,

$$\begin{aligned}p(\alpha_i) =& \alpha_i^3 \operatorname{tr}\left(V_i^\top V_i \operatorname{triu}\left(V_i^\top V_i\right)\right)\\ &+\alpha_i^2 \operatorname{tr}\left(V_i^\top V_i \operatorname{triu}\left(X_i^\top V_i\right) + V_i^\top V_i \operatorname{triu}\left(V_i^\top X_i\right) + V_i^\top X_i \operatorname{triu}\left(V_i^\top V_i\right)\right)\\ &+\alpha_i \operatorname{tr}\left(V_i^\top A V_i + V_i^\top X_i \operatorname{triu}\left(V_i^\top X_i\right) + V_i^\top X_i \operatorname{triu}\left(X_i^\top V_i\right)\right.\\ &\left.+V_i^\top V_i \operatorname{triu}\left(X_i^\top X_i\right)\right) + \operatorname{tr}\left(V_i^\top A X_i + V_i^\top X_i \operatorname{triu}\left(X_i^\top X_i\right)\right).\end{aligned} \tag{33}$$

Using either equation, we are able to avoid $\alpha_i = 0$ if X_i stays in the space spanned by eigenvectors while X_i is not any stable fixed point. The local convergences for both choices of stepsize can be proved in a similar way as in Sect. 4. Regarding the computational cost,

since all trace terms can be computed in an accumulative way, the computational cost for getting coefficients in (33) and (32) remains the same for all i.

According to our numerical experiments, the columnwise stepsize strategy based on the linesearch significantly outperforms the fixed stepsize, while there is not much difference between solving $\mathrm{tr}\left(V_i^\top g\left(X_i+\alpha V_i\right)\right)=0$ and $\mathrm{tr}\left(V_i^\top \nabla f\left(X_i+\alpha V_i\right)\right)=0$. Throughout the rest paper, we solve $\mathrm{tr}\left(V_i^\top g\right)=0$ for stepsize.

5.3 Column Locking

In Sect. 4 we notice that each column has its own convergence rate, and later columns converge slower than earlier ones in terms of the analysis. A similar conclusion is observed numerically. It wastes computation resources if all columns are updated throughout iterations. Hence in addition to the overall stopping criterion of TriOFM methods, $\left\|g\left(X^{(t)}\right)\right\|_{\mathrm{F}}<\epsilon$, we introduce a column locking technique to allow early stopping for converged columns.

The column locking has been widely adopted in many traditional eigensolvers. However, in orthogonalization-free eigensolvers [5, 24, 39], the locking technique is not applicable since all columns are coupled together throughout iterations. TriOFM, differently, can adopt the column locking in a specific ordering. Since the earlier columns in TriOFM are independent of later columns, as long as they have converged, we could lock these columns.

The column locking strategy depends on the error propagation among columns. Lemma 2 hints the error propagation. However, we find that the error estimation in Lemma 2 is pessimistic. Here we give an intuitive but helpful discussion on the error propagation, where higher order terms in the error vector are ignored. Let $\varepsilon_i^{(t)}=\left(\varepsilon_{i,1}^{(t)},\cdots,\varepsilon_{i,n}^{(t)}\right)^\top$ be the error of $x_i^{(t)}$ projected to eigenvectors of A, *i.e.*, $\varepsilon_i^{(t)}=U^\top\left(x_i^{(t)}-x_i^*\right)$. The projected error $\varepsilon_i^{(t)}$ here is consistent with the notations in Sect. 4, where A is assumed to be diagonal. The error in i-th column of TriOFM-(Obj1), without higher order terms of ε, admits,

$$\varepsilon_i^{(t+1)}=\begin{pmatrix}(1+\alpha\lambda_i)\varepsilon_{i,1}^{(t)}-\alpha\sqrt{\lambda_1\lambda_i}\varepsilon_{1,i}^{(t)}\\ \vdots\\ (1+\alpha\lambda_i)\varepsilon_{i,i-1}^{(t)}-\alpha\sqrt{\lambda_{i-1}\lambda_i}\varepsilon_{i-1,i}^{(t)}\\ (1+2\alpha\lambda_i)\varepsilon_{i,i}^{(t)}\\ (1+\alpha(\lambda_{i+1}-\lambda_i))\varepsilon_{i,i+1}^{(t)}\\ \vdots\\ (1+\alpha(\lambda_n-\lambda_i))\varepsilon_{i,n}^{(t)}\end{pmatrix}. \tag{34}$$

The Eq. (34) implies that the lower triangular part of $\left(\varepsilon_1^{(t)},\cdots,\varepsilon_p^{(t)}\right)$ does not depend on other error vectors and thus is able to converge to zero as t goes to infinity even if other columns are locked. For the strict upper triangular part, we consider the case where columns earlier than i are locked with fixed errors. Taking the j-th row ($j<i$) for example, when $\varepsilon_{j,i}^{(t)}$ is fixed, $\varepsilon_{i,j}^{(t+1)}=(1+\alpha\lambda_i)\,\varepsilon_{i,j}^{(t)}-\alpha\sqrt{\lambda_j\lambda_i}\varepsilon_{j,i}^{(t)}$ has the fixed point $\varepsilon_{i,j}=\sqrt{\frac{\lambda_j}{\lambda_i}}\varepsilon_{j,i}$ as t goes to infinity. Notice that each entry in the upper triangular part is only influenced by an error term in the lower triangular part. Through a detailed derivation by induction, we have

an estimation on the norms of error vectors for TriOFM-(Obj1) as $t \to \infty$,

$$\|\varepsilon_1\| \sim \epsilon, \quad \|\varepsilon_2\| \sim \sqrt{\frac{\lambda_1}{\lambda_2}}\epsilon, \quad \cdots, \quad \|\varepsilon_p\| \sim \sqrt{\frac{\lambda_1}{\lambda_p}}\epsilon. \tag{35}$$

The estimation above shows that there is a uniform upper bound on $\sqrt{-\lambda_i}\,\|\varepsilon_i\|$ for all $1 \leq i \leq p$.

Further analysis on $g_1\left(X^{(t)}\right)$ in the stopping criterion shows that the norms of columns of $g_1\left(X^{(t)}\right)$ admit the same scaling as that of $\|\varepsilon_1\|, \ldots, \|\varepsilon_p\|$. Hence we could include an additional term with scaling $\sqrt{-\lambda_i}$ for the i-th column. A good choice is $\|Ax_i\|^{\frac{1}{3}}$. The recommended locking criterion for TriOFM-(Obj1) is

$$\left\|g_1\left(x_i^{(t)}\right)\right\| \left\|Ax_i^{(t)}\right\|^{\frac{1}{3}} < \epsilon. \tag{36}$$

An analog estimation can be carried out for TriOFM-(Obj2) as well. The unified locking criterion for TriOFM-(Obj2) is

$$\left\|g_2\left(x_i^{(t)}\right)\right\| \|Ax_i\| < \epsilon. \tag{37}$$

6 Numerical Results

In this section, we show the efficiency of TriOFM applying to three different groups of matrices, *i.e.*, random matrices with different eigenvalue distributions, a synthetic matrix from DFT, and a matrix of Hubbard model under FCI framework.

In Sect. 6.1, we first show that TriOFM with a constant stepsize locally has linear convergence rate on random matrices with different eigenvalue distributions, which agrees with our analysis in Sect. 4. Further, accelerating techniques introduced in Sect. 5 are adopted and compared. Then we apply TriOFM with these techniques to two matrices from DFT and FCI in Sects. 6.2 and 6.3 respectively. In both examples, TriOFM converges to sparse eigenvectors, whereas traditional orthogonalization-free methods fail to recover the sparsity. Regarding the computational cost, TriOFM is, in general, comparable to its non-triangularized counterpart.

For a fair comparison reason, we adopt the same stopping criterion for both TriOFM and OFM: the relative residual is smaller than a tolerance ϵ, *i.e.*, $\frac{\|AXQ - XQ\Lambda_X\|_{\mathrm{F}}}{\|AXQ\|_{\mathrm{F}}} < \epsilon$, where Q and the diagonal matrix Λ_X come from solving a generalized eigenvalue problem, $\left(X^\top AX\right) Q = \left(X^\top X\right) Q\Lambda_X$. Such a stopping criterion is not applicable in practice. For the illustration purpose, it is adopted in this section for a fair comparison. If the column locking is enabled in TriOFM, the algorithm could stop if all columns are locked. Two measurements of accuracies are used. The first one measures the accuracy of eigenvectors,

$$e_{vec} = \min_{X^* \in \mathcal{X}^*} \frac{\|X - X^*\|_{\mathrm{F}}}{\|X^*\|_{\mathrm{F}}}, \tag{38}$$

where $\mathcal{X}^*$ denotes the set of all possible stable fixed points of the used algorithm. The second measures the accuracy of eigenvalues,

$$e_{val} = \frac{\left|\mathrm{tr}\left(\left(X^\top X\right)^{-1} X^\top AX\right) - \sum_{i=1}^{p} \lambda_i\right|}{\left|\sum_{i=1}^{p} \lambda_i\right|}. \tag{39}$$

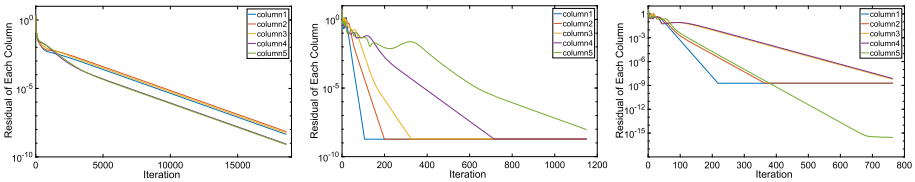

Fig. 1 Convergence behavior of TriOFM-(Obj1) applying to A_{uni} (left), A_{log} (middle), and A_{ushape} (right) with fixed stepsize $\alpha = 0.4$ and column locking

We also define two measurements for computational costs. Since all of our codes are implemented in MATLAB, which favors matrix operations over vector operations, the runtime comparison is not fair. Hence we introduce *number of iterations* and *number of matrix–vector multiplications*. Without column locking, the number of matrix–vector multiplications is simply the number of iterations multiplying the number of columns in $X^{(t)}$. When column locking is enabled, it is the summation of the number of unlocked columns throughout iterations.

6.1 Random Matrices

In this section, we apply different TriOFM algorithms to random matrices and compare the performance against their OFM counterparts. We generate random matrices of size $n = 500$. The number of desired eigenpairs is $p = 5$ and $p = 10$ in Sects. 6.1.1 and 6.1.2 respectively. Random matrices are generated of the form

$$A = U^\top \Lambda U, \tag{40}$$

where U is a random orthogonal matrix generated by a QR factorization of a random matrix with entries sampled from a standard normal distribution independently. Here Λ denotes a diagonal matrix with its elements $\{\lambda_i\}_{i=1}^n$ generated from three different ways,

1. (Uniform) $\lambda_i = \frac{i-1}{500} - 1$ for $1 \le i \le n$;
2. (Logarithm) $\lambda_i = -\frac{2^{10}}{500}\frac{1}{2^i}$ for $1 \le i \le n$;
3. (U-Shape) $\lambda_1 = -\frac{14}{16}, \lambda_2 = -\frac{10}{16}, \lambda_3 = -\frac{8}{16}, \lambda_4 = -\frac{7}{16}, \lambda_5 = -\frac{5}{16}, \lambda_i = -\frac{1}{16}$ for all $6 \le i \le n$.

In the U-shape case, the first 5 eigengaps are $\frac{4}{16}, \frac{2}{16}, \frac{1}{16}, \frac{2}{16}, \frac{4}{16}$, which decays exponentially first and then grows exponentially. We denote these three random matrices as A_{uni}, A_{log}, and A_{ushape}. The eigengaps of A_{uni} and A_{log} are two typical cases for many applications. While, A_{ushape} is constructed to reveal the difference between TriOFM and OFM.

6.1.1 Local Convergence Rate

We first numerically validate the convergence rate proved in Sect. 4. The stepsize is fixed, $\alpha = 0.4$. Initial state $X^{(0)} \in \mathbb{R}^{n\times p}$ is a random matrix with unit column lengths. Column locking technique is applied, whereas momentum techniques are disabled.

Figure 1 shows the convergence behaviors of TriOFM-(Obj1) applied to three random matrices. Nonlinear convergence is observed in all three figures for the first few iterations. Linear convergence is then observed until convergence. This agrees with our analysis.

Table 2 Local convergence rate of TriOFM-(Obj1) applying to A_{log}

Matrix		Convergence rate				
		λ_1	λ_2	λ_3	λ_4	λ_5
A_{log}	Reference rate	0.7952	0.8976	0.9488	0.9744	0.9872
	Numerical rate	0.7952	0.8976	0.9488	0.9744	0.9872
	Difference	2.4×10^{-7}	4.0×10^{-8}	4.4×10^{-7}	3.4×10^{-7}	2.6×10^{-7}

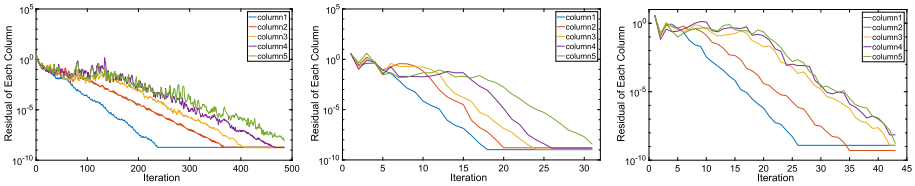

Fig. 2 Convergence behavior of TriOFM-(Obj1) applying to A_{uni} (left), A_{log} (middle), and A_{ushape} (right) with CG, linesearch, and column locking

In Fig. 1 left, all curves are parallel to each other, and their convergence rates are the same, which agrees with our analysis since A_{uni} has all equal eigengaps. In Fig. 1 middle, curves have different slopes and hence different convergence rates. Here we provide a quantitative comparison of the convergence rates for A_{log} in Table 2. We fit the slopes of curves and use them as numerical rates, whereas reference rates are computed as the asymptotic convergence rate in Remark 1. The absolute difference between these two rates are listed in the third row. Table 2 shows that numerical rates agree with reference rates up to seven digits. Hence we claim that the rate in Theorem 5 is tight. In Fig. 1 right, the first four curves have convergence rates that agree with our theoretical results, whereas the last one converges faster than expected. Its convergence rate is theoretically upper bounded by that of previous columns but numerically is faster. Through these numerical results, we claim that our theoretical analysis of the local convergence rate provides a tight upper bound for practice.

In Sect. 4, we prove the local convergence column by column, *i.e.*, the convergence of later column is proved if all earlier columns are close to their stable fixed points. An interesting numerical observation from Fig. 1 is that the linear convergences of columns may have some overlapping iterations, *e.g.*, there are a lot of iterations that all curves converge linearly parallelly for matrix A_{uni}. Such overlapping leads to faster convergence for the overall algorithm.

6.1.2 Accelerating Techniques

In this section, we investigate the accelerating techniques. The convergence behaviors of TriOFM-(Obj1) are included in Fig. 2. Overall, the convergence of TriOFM-(Obj1) with accelerating techniques are much faster than that of vanilla TriOFM-(Obj1). Next, we provide more quantitative comparisons for column locking and momentum accelerations for both TriOFM-(Obj1) and TriOFM-(Obj2).

Specifically, in this section, the tolerance ϵ used for stopping criteria and column locking is 10^{-8}. And each experiment is repeated 500 times, with random matrices and initial values. For the number of iterations (Iter Num) and the number of matrix–vector multiplications

Table 3 Performance comparison of TriOFM-(Obj1) applied to A_{uni} with and without column locking. CG and exact linesearch are enabled

Method	Iter num.			Mat-Vec num.		
	Mean	Max	Min	Mean	Max	Min
TriOFM-(Obj1) + CG +locking	642.2	800	554	4990.2	6905	4353
TriOFM-(Obj1) + CG	643.1	832	518	6431.4	8320	5180

(Mat-Vec Num), we report the mean, max, and min among 500 random tests. In all tests, the linesearch is always enabled.

First, we show the advantage of column locking. Table 3 list the results for TriOFM-(Obj1) applied to A_{uni} with and without column locking. We observe that the numbers of iterations remain the same with and without the column locking. However, the number of matrix–vector multiplication is significantly reduced with column locking, and hence the computational cost is reduced. Similar results are observed for TriOFM-(Obj1) on other matrices and TriOFM-(Obj2) as well. We omit those results for the sake of brevity.

Then we explore the advantages of momentum and CG techniques. For algorithms with vanilla momentum acceleration, the coefficients are chosen as $\beta = 0.9$ for (Obj1) and $\beta = 0.95$ for (Obj2). Several different values of β have been tested for both objective functions and we pick these βs for objective functions with the fastest convergence.

Numerical results are summarized in Table 4 for A_{log}. In all cases, TriOFMs converge in less number of iterations and less number of matrix–vector multiplications. There are two reasons behind the results. First, the convergences of earlier columns in TriOFM are faster than that of the last column, whereas the convergences of all columns in OFM are the same as the last column in TriOFM. Second, in TriOFM, different linesearch stepsizes are applied to different columns, whereas OFM uses a single stepsize for all columns, which is impacted by the smallest eigengap. Overall, the computational costs of TriOFM and OFM depend on

Table 4 Performance comparison of TriOFM and OFM with and without momentum accelerating techniques for A_{log}. Here GD stands for the vanilla gradient descent method. Exact linesearch is enabled for all algorithms and column locking is enabled for TriOFM

Objective Ffunction	Method	Iter num			Mat-Vec num		
		Mean	Max	Min	Mean	Max	Min
(Obj1)	TriOFM+CG	49.0	59	40	414.7	519	334
	OFM+CG	616.1	1881	333	6161.4	18810	3330
	TriOFM+Momentum	46.4	58	38	401.4	510	335
	OFM+Momentum	963.6	1468	614	9635.6	14680	6140
	TriOFM+GD	52.1	67	42	492.0	635	415
	OFM+GD	11460.7	17124	4591	114607.2	171240	75910
(Obj2)	TriOFM+CG	279.0	553	193	1071.0	1499	882
	OFM+CG	953.2	2500	550	9532.2	25000	5500
	TriOFM+Momentum	701.4	997	504	2217.0	2588	1840
	OFM+Momentum	1275.3	2033	738	12752.8	20330	7380
	TriOFM+GD	5150.7	9280	2663	12168.2	16500	7214
	OFM+GD	21222.2	30462	14156	212221.9	304620	141560

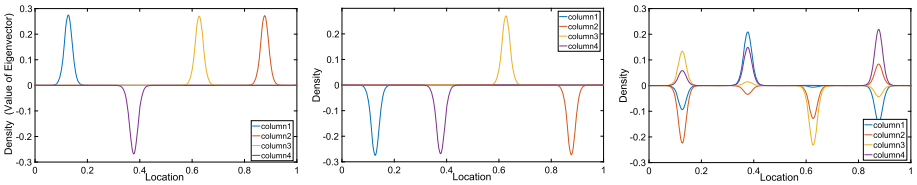

Fig. 3 Left figure plots the ground truth eigenvectors associated with four low-lying eigenvalues of problem (41); middle figure plots scaled four convergent columns from TriOFM-(Obj1); and right figure plots scaled four convergent columns from the OFM-(Obj1)

Table 5 Performance comparison of TriOFM-(Obj1) and OFM-(Obj1) applied to (41)

Method	Iter num	Mat-Vec num	NNZ	e_{vec}	e_{val}
TriOFM+CG	567.5	5134.2	1328	8.26×10^{-8}	2.00×10^{-15}
OFM+CG	413.6	4135.7	4974.0	–	2.50×10^{-15}

the eigengap distribution of the matrix. For A_{log}-like matrices, TriOFM outperforms OFM. Further, algorithms with CG converge faster or equally fast as their momentum accelerated versions with carefully chosen parameter βs. Hence we recommend CG as the momentum acceleration since it is hyper-parameter free.

Through all these tests, our best choice is to use TriOFM with CG, exact linesearch, and column locking. As in the later sections, this configuration will be the default TriOFM, and we will focus on the sparsity of eigenvectors.

6.2 Synthetic Density Functional Theory

In this section, we perform TriOFM on a synthetic example from DFT computation. The example is a second-order differential operator on the domain [0, 1] with periodic boundary condition,

$$H(x) = -\Delta + V(x), \tag{41}$$

where $-\Delta$ is the Laplace operator denoting the kinetic term and $V(x)$ is a local potential with four Gaussian potential wells,

$$V(x) = -\sum_{i=1}^{4} \alpha_i e^{-\frac{(x-\ell_i)^2}{2\sigma^2}}. \tag{42}$$

The centers of these wells locate at $\ell_i = \frac{2i-1}{8}$, the depths of the wells are $\alpha_i = 850 + 50 \times \mathrm{mod}(i, 4)$, and the constant width of these wells is $\sigma = 0.1$. This second order differential operator, (41), can be viewed as the linear operator in a self-consistent field iteration in DFT computation, simulating four different atoms located periodically on a line. In this example, we are interested in computing the low-lying four eigenpairs. The associated matrix is obtained via discretizing the problem on a uniform grid with $n = 500$ points, where the Laplace operator is discretized using the central difference scheme. In Fig. 3 left, we plot the four eigenvectors corresponding to smallest four eigenvalues. Due to the localized potential and periodicity, the eigenvectors associated with low-lying eigenvalues have localized property, which means that these eigenvectors are sparse.

Numerical results are demonstrated in Fig. 3 and Table 5. The tolerance is 10^{-8}, and each algorithm is perform 100 times with random initial states. Figure 3 middle plots the scaled four convergent columns from TriOFM-(Obj1) and the right figure plots scaled four convergent columns from the non-triangularized counterpart. Table 5 includes the number of iterations, the number of matrix–vector multiplications, the number of nonzeros (NNZ) of the converged point X, and the accuracies. In Table 5, the NNZ is the number of entries with absolute values greater than 10^{-5}. The NNZ of the ground truth eigenvectors is 1328. Since OFM-(Obj1) does not provide eigenvectors without an extra orthogonalization step, the accuracy of eigenvectors is not available.

According to Fig. 3, the convergent columns of TriOFM-(Obj1) recover the eigenvectors up to a sign difference. While the convergent columns of OFM-(Obj1) mix all four eigenvectors and have nonzeros near all four Gaussian centers. Hence the sparsity of eigenvectors is destroyed. Overall, TriOFM-(Obj1) achieves 75% saving in NNZ comparing to that of OFM-(Obj1). Since the memory cost is a key bottleneck in many DFT computations, such a saving is important. Meanwhile, regarding the number of iterations and the number of matrix–vector multiplications, TriOFM-(Obj1) is slightly more expensive than OFM-(Obj1). Hence there is a trade-off between time and space. If the parallelizability of TriOFM is further taken into account, then TriOFM would be a valuable alternative eigensolver for DFT.

6.3 Full Configuration Interaction

This section solves the low-lying eigenpairs for a two-dimensional Hubbard model under the FCI framework. The Hubbard model is widely used in solid-state physics, which only considers the neighboring hopping and on-site interaction. Under the FCI framework, the matrix size scales factorially with respect to the problem size and the number of electrons. The eigenvectors associated with low-lying eigenvalues are sparse. FCI problems are the most important applications of TriOFM.

The Hamiltonian operator in the second quantization notation is,

$$\hat{H} = -t \sum_{\langle r,r'\rangle,\sigma} \hat{a}^{\dagger}_{r,\sigma}\hat{a}_{r',\sigma} + U \sum_{r} \hat{a}^{\dagger}_{r,\uparrow}\hat{a}_{r,\uparrow}\hat{a}^{\dagger}_{r,\downarrow}\hat{a}_{r,\downarrow} \tag{43}$$

where t is the hopping strength, U is the interaction strength, r, r' are lattice index, $\langle r, r'\rangle$ means that r and r' are neighbors on the lattice, $\hat{a}^{\dagger}_{r,\sigma}$ and $\hat{a}_{r,\sigma}$ denotes the creation and annihilation operator of an electron with spin σ on r. The matrix in this section is generated from the Hubbard model in momentum space. The Fourier transform of the creation and annihilation operator is $\hat{a}_{k,\sigma} = \frac{1}{\sqrt{N^{site}}} \sum_r e^{\imath k\cdot r}\hat{a}_{r,\sigma}$, where k is the wave number and N^{site} is the number of lattice sites. The Hamiltonian operator in momentum space is,

$$\hat{H} = t \sum_{k,\sigma} -2(\cos k_1 + \cos k_2)\hat{a}^{\dagger}_{k,\sigma}\hat{a}_{k,\sigma} + \frac{U}{N^{site}} \sum_{k,p,q} \hat{a}^{\dagger}_{p-q,\uparrow}\hat{a}^{\dagger}_{k+q,\downarrow}\hat{a}_{k,\downarrow}\hat{a}_{p,\uparrow} \tag{44}$$

for $k = (k_1, k_2)$.

We adopt a 2D Hubbard model on a lattice of size 4×4 with 8 electrons (4 spin-up and 4 spin-down). The strength of hopping and interaction are $t = 1$ and $U = 0.25N^{site}$ respectively. The FCI matrix has diagonal entries between -20 and 20 and off-diagonal entries being ± 0.25. The matrix size is about $(2 \cdot 10^5) \times (2 \cdot 10^5)$. We compute the smallest $p = 10$ eigenpairs. TriOFM-(Obj1) and OFM-(Obj1) are applied to address this problem. The tolerance is 10^{-10}. For each algorithm, we perform 100 times with random initial states. The mean of the number of iterations, the number of matrix–vector multiplications, NNZ,

Table 6 Performance comparison of TriOFM-(Obj1) and OFM-(Obj1) on (44)

Method	Iter num	Mat-Vec num	NNZ	e_{vec}	e_{val}
TriOFM+CG	1253.0	7708.6	1.115×10^6	3.308×10^{-8}	5.597×10^{-12}
OFM+CG	1381.7	13817.2	1.508×10^6	–	5.197×10^{-12}

and accuracies are reported in Table 6. Similarly, the NNZ is the number of entries with a magnitude greater than 10^{-5}.

According to Table 6, TriOFM-(Obj1) requires less number of iterations and matrix–vector multiplications than OFM-(Obj1). In FCI problems, the number of matrix–vector multiplications is proportional to the actual runtime. Hence we expect that TriOFM-(Obj1) would achieve better runtime than OFM-(Obj1) on FCI problems. Notice that the multiplicity of some eigenvalues in our FCI matrix is not one. Hence the stable fixed points are subspaces. NNZ for TriOFM-(Obj1) varies over 100 executions and Table 6 reports its mean. On average, TriOFM-(Obj1) achieves better sparsity compared to OFM-(Obj1). Through our numerical results, TriOFM-(Obj1) outperforms OFM-(Obj1) on the FCI problem.

Remark 3 Solving practical FCI problems is the major target in designing TriOFM. In FCI problems, low-lying eigenvalues and the associated eigenvectors are computed as the ground state and low-lying excited states. Almost all traditional eigensolvers are not applicable to FCI problems. OFM with coordinate-wise descent method is an option to obtain the sparse eigenvectors. While the arbitrary rotation would significantly increase the memory cost. Hence, we design TriOFM converging to the sparse eigenvectors directly. According to our numerical result of the FCI problem, TriOFM outperforms OFM and is a more promising method to address FCI problems. This paper is the first step toward computing the FCI excited states. Coupling TriOFM together with a parallelized coordinate-wise descent method, we would be able to address FCI problems for transition metals of interest.

7 Conclusion and Discussion

In this work, we introduce a novel TriOFM for solving extreme eigenvalue problems. Using TriOFM, the eigenpairs are directly solved via orthogonalization-free iterative methods, where the orthogonalization-free feature is crucial for large-scale eigenvalue problems with sparse eigenvectors. Two specific algorithms, namely TriOFM-(Obj1) and TriOFM-(Obj2), are proposed for (Obj1) and (Obj2). Global convergences are guaranteed for almost all initial states [11]. Locally, we prove that, in neighbors of stable fixed points, TriOFM-(Obj1) converges linearly. The convergence proof can be adapted to show the linear convergence of TriOFM-(Obj2) as well. Although the proposed algorithms are different from general gradient-based algorithms, acceleration techniques, including momentum, linesearch, and column locking, still work effectively. According to numerical results on both synthetic examples and the example from practice, TriOFM-(Obj1) and TriOFM-(Obj2) converge efficiently and obtain the sparse eigenvectors without any orthogonalization step.

There are many future directions. As has been mentioned before, TriOFM is applicable to many other objective functions beyond (Obj1) and (Obj2). We would like to apply TriOFM to other objective functions and obtain powerful algorithms. Moreover, we claim the advantage of TriOFM in keeping sparsity towards convergent. It is an interesting future direction to explore truncation techniques and coordinate-wise methods so that the sparsity is preserved

throughout iterations. The application to FCI problems would be of great interest to many other communities, including computational physics, chemistry, and material science, etc. In addition to the above two directions, orthogonalization-free algorithms are friendly to parallel computing. Hence the parallelization of these proposed algorithms is another future direction.

Acknowledgements The authors thank Jianfeng Lu and Zhe Wang for helpful discussions.

Funding W. Gao was partially supported by National Key R&D Program of China under Grant No. 2020YFA0711900, 2020YFA0711902 and National Natural Science Foundation of China under Grant No. 71991471, U1811461. Y. Li and B. Lu were partially supported by National Natural Science Foundation of China under Grant No. 12271109.

Data Availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest The authors have not disclosed any competing interest.

A Proof of Theorem 4

Proof of Theorem 4 All fixed points of (16) satisfy $g_2(X) = 0$. We first analyze the fixed points for a single column case and then complete the proof by induction. Notations used in this proof are the same as those in the proof of Theorem 3.

We denote the single column X as x. Obviously, when $x = 0$, we have $g_2(x) = 0$. Now, consider the nontrivial case $x \neq 0$. The equality $g_2(x) = 0$ can be expanded as,

$$\left(\left(2 - x^\top x\right) A - x^\top A x I\right) x = 0. \tag{45}$$

According to (45), for nonzero x, the matrix $B = \left(2 - x^\top x\right) A - x^\top A x I$ must has a zero eigenvalue and x lies in its corresponding eigenspace. When $x^\top x = 2$, the matrix $B = x^\top A x I$ does not have zero eigenvalue due to the negativity assumption on A. Hence x is parallel to one of A's eigenvector, *i.e.*, $Ax = \lambda x$. Substituting this into (45), we obtain,

$$2(1 - x^\top x)\lambda x = 0. \tag{46}$$

Since $\lambda < 0$ and $x \neq 0$, we have $x^\top x = 1$. Hence we conclude that for $g_2(x) = 0$, x is either a zero vector or an eigenvector of A.

Now we consider multicolumn case. The first column of $g_2(X) = 0$ is the same as (45). Hence $X_1 = U P_1 S_1$.

Assume the first i columns of X obey $X_i = U P_i S_i$. Then the $(i+1)$-th column of $g_2(X) = 0$ is

$$2Ax_{i+1} - Ax_{i+1}x_{i+1}^\top x_{i+1} - x_{i+1}x_{i+1}^\top Ax_{i+1} - AX_i X_i^\top x_{i+1} - X_i X_i^\top Ax_{i+1} = 0. \tag{47}$$

Obviously, if $x_{i+1} = 0$, then (47) holds. When $x_{i+1} \neq 0$, we left multiply (47) with $X_i^\top$, adopt the commuting property of diagonal matrices, and obtain,

$$\begin{aligned} 0 =& S_i P_i^\top \left(2\Lambda - x_{i+1}^\top x_{i+1}\Lambda - x_{i+1}^\top Ax_{i+1} I - \Lambda P_i P_i^\top - \Lambda\right) U^\top x_{i+1} \\ =& - S_i P_i^\top \left(x_{i+1}^\top x_{i+1}\Lambda + x_{i+1}^\top Ax_{i+1} I\right) U^\top x_{i+1} \end{aligned} \tag{48}$$

where the second equality adopts the fact that $P_i^\top \Lambda P_i P_i^\top = P_i^\top \Lambda$. Due to the negativity of A, we notice that $x_{i+1}^\top x_{i+1}\Lambda + x_{i+1}^\top A x_{i+1} I$ is a diagonal matrix with strictly negative diagonal entries. Hence the equality (48) is equivalent to

$$S_i P_i^\top U^\top x_{i+1} = 0. \tag{49}$$

As long as (49) holds, we have $X_i^\top x_{i+1} = 0$ and $X_i^\top A x_{i+1} = 0$. Therefore, solving (47) can be addressed via solving

$$2Ax_{i+1} - Ax_{i+1}x_{i+1}^\top x_{i+1} - x_{i+1}x_{i+1}^\top Ax_{i+1} = 0. \tag{50}$$

Hence x_{i+1} satisfies (49). Combining the solution of the single column case (45) and the constraint (49), we conclude that X_{i+1} is of the form $U P_{i+1} S_{i+1}$.

The stabilities of fixed points should also be analyzed through the spectrum properties of their Jacobian matrices. The Jacobian matrix $\mathrm{D}g_2(X)$, again, can be written as a p-by-p block matrix. And using the similar argument as in the proof of Theorem 3, $\mathrm{D}g_2(X) = \mathrm{D}G$ is a block upper triangular matrix whose spectrum is determined by the spectrum of its diagonal blocks. Through a multivariable calculus, we obtain the expression for J_{ii} as,

$$J_{ii} = 2A - AX_iX_i^\top - X_iX_i^\top A - Ax_ix_i^\top - x_i^\top x_i A - x_i^\top Ax_i I - x_i x_i^\top A. \tag{51}$$

We first show the stability of the fixed points of form $X = U_p D$. Substituting these points into (51), we have,

$$J_{ii} = A - 2U_i\Lambda_i U_i^\top - 2\lambda_i u_i u_i^\top - \lambda_i I. \tag{52}$$

Since λ_i is smaller than all eigenvalues of $A - U_i\Lambda_i U_i^\top$, $A - U_i\Lambda_i U_i^\top - \lambda_i I$ is strictly positive definite. The rest part of (51) is, obviously, positive definite. Hence J_{ii} is strictly positive definite for all $i = 1, 2, \ldots, p$ and fixed points of the form $X = U_p D$ are stable fixed points.

Next we show the rest fixed points are not stable. For a fixed point X, we denote the first index s such that $x_s^\top u_s = 0$. Then we estimate $u_s^\top J_{ss} u_s$ as,

$$u_s^\top J_{ii} u_s = 2\lambda_s - x_s^\top x_s \lambda_s - x_s^\top A x_s < 0, \tag{53}$$

since $x_s^\top x_s \leq 1$ and A is negative definite. Therefore, the rest fixed points are not stable. □

References

1. Banerjee, A.S., Lin, L., Hu, W., Yang, C., Pask, J.E.: Chebyshev polynomial filtered subspace iteration in the discontinuous Galerkin method for large-scale electronic structure calculations. J. Chem. Phys. **145**(15), 154101 (2016)
2. Berljafa, M., Wortmann, D., Di Napoli, E.: An optimized and scalable eigensolver for sequences of eigenvalue problems. Concurr. Comput. Pract. Exp. **27**(4), 905–922 (2015)
3. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. SIAM Rev. **60**(2), 223–311 (2018)
4. Brouder, C., Panati, G., Calandra, M., Mourougane, C., Marzari, N.: Exponential localization of Wannier functions in insulators. Phys. Rev. Lett. **98**(4), 046402 (2007)
5. Corsetti, F.: The orbital minimization method for electronic structure calculations with finite-range atomic basis sets. Comput. Phys. Commun. **185**(3), 873–883 (2014)
6. Dai, X., Wang, Q., Zhou, A.: Gradient flow based discretized Kohn-Sham density functional theory. arxiv:1907.06321 (2019a)
7. Dai, X., Zhang, L., Zhou, A.: Adaptive step size strategy for orthogonality constrained line search methods. arxiv:1906.02883 (2019b)

8. Davidson, E.R.: The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. J. Comput. Phys. **17**(1), 87–94 (1975)
9. Gao, B., Liu, X., Chen, X., Yuan, Y.X.: A new first-order algorithmic framework for optimization problems with orthogonality constraints. SIAM J. Optim. **28**(1), 302–332 (2018)
10. Gao, B., Liu, X., Yuan, Y.-X.: Parallelizable algorithms for optimization problems with orthogonality constraints. SIAM J. Sci. Comput. **41**(3), A1949–A1983 (2019)
11. Gao, W., Li, Y., Lu, B.: Global convergence of triangularized orthogonalization-free method for solving extreme eigenvalue problems. arxiv:2110.06212 (2021)
12. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. The Johns Hopkins University Press, Baltimore (2013)
13. Golub, G.H., Ye, Q.: An inverse free preconditioned krylov subspace method for symmetric generalized eigenvalue problems. SIAM J. Sci. Comput. **24**(1), 312–334 (2002)
14. Huang, W., Gallivan, K.A., Absil, P.-A.: A Broyden class of quasi-Newton methods for Riemannian optimization. SIAM J. Optim. **25**(3), 1660–1685 (2015)
15. Kalkreuter, T., Simma, H.: An accelerated conjugate gradient algorithm to compute low-lying eigenvalues-a study for the dirac operator in su (2) lattice qcd. Comput. Phys. Commun. **93**(1), 33–47 (1996)
16. Knowles, P.J., Handy, N.C.: A new determinant-based full configuration interaction method. Chem. Phys. Lett. **111**(4–5), 315–321 (1984)
17. Knyazev, A.V.: Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method. SIAM J. Sci. Comput. **23**(2), 517–541 (2001)
18. Lee, J.D., Panageas, I., Piliouras, G., Simchowitz, M., Jordan, M.I., Recht, B.: First-order methods almost always avoid strict saddle points. Math. Program. **176**(1–2), 311–337 (2019)
19. Lei, Q., Zhong, K., Dhillon, I.S.: Coordinate-wise power method. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems, pp. 2064–2072. Curran Associates Inc, New York (2016)
20. Levitt, A., Torrent, M.: Parallel eigensolvers in plane-wave density functional theory. Comput. Phys. Commun. **187**, 98–105 (2015)
21. Li, R.-C.: Rayleigh quotient based optimization methods for eigenvalue problems. In Matrix Functions and Matrix Equations, World Scientific, pp 76–108 (2015)
22. Li, Y., Lu, J.: Bold diagrammatic Monte Carlo in the lens of stochastic iterative methods. Trans. Math. Appl. **3**(1), 1–17 (2019)
23. Li, Y., Lu, J.: Optimal orbital selection for full configuration interaction (OptOrbFCI): pursuing basis set limit under budget. arxiv:2004.04205 (2020)
24. Li, Y., Lu, J., Wang, Z.: Coordinatewise descent methods for leading eigenvalue problem. SIAM J. Sci. Comput. **41**(4), A2681–A2716 (2019)
25. Li, Y., Yang, H.: Spectrum slicing for sparse Hermitian definite matrices based on Zolotarev's functions. arxiv:1701.08935 (2017)
26. Liu, W.: An algorithm for solving eigenvectors based on unconstrained optimization problem. Master's thesis, Fudan University (2021)
27. Liu, X., Wen, Z., Zhang, Y.: An efficient Gauss-Newton algorithm for symmetric low-rank product matrix approximations. SIAM J. Optim. **25**(3), 1571–1608 (2015)
28. Lu, J., Thicke, K.: Orbital minimization method with l1 regularization. J. Comput. Phys. **336**, 87–103 (2017)
29. Lu, J., Yang, H.: Preconditioning orbital minimization method for planewave discretization. Multiscale Model. Simul. **15**(1), 254–273 (2017)
30. Mauri, F., Galli, G., Car, R.: Orbital formulation for electronic-structure calculations with linear system-size scaling. Phys. Rev. B **47**(15), 9973–9976 (1993)
31. Ordejón, P., Drabold, D.A., Grumbach, M.P., Martin, R.M.: Unconstrained minimization approach for electronic computations that scales linearly with system size. Phys. Rev. B **48**(19), 14646–14649 (1993)
32. Ovtchinnikov, E.E.: Computing several eigenpairs of Hermitian problems by conjugate gradient iterations. J. Comput. Phys. **227**(22), 9477–9497 (2008)
33. Peter Tang, P.T., Polizzi, E.: FEAST as a subspace iteration eigensolver accelerated by approximate spectral projection. SIAM J. Matrix Anal. Appl. **35**(2), 354–390 (2014)
34. Polak, E., Ribiere, G.: Note sur la convergence de méthodes de directions conjuguées. ESAIM Math. Modell. Numer. Anal Modél. Math. Anal. Numér. **3**(R1), 35–43 (1969)
35. Quillen, P., Ye, Q.: A block inverse-free preconditioned Krylov subspace method for symmetric generalized eigenvalue problems. J. Comput. Appl. Math. **233**(5), 1298–1313 (2010)
36. Saad, Y., Chelikowsky, J.R., Shontz, S.M.: Numerical methods for electronic structure calculations of materials. SIAM Rev. **52**(1), 3–54 (2010)

37. Stubbs, K.D., Watson, A.B., Lu, J.: Existence and computation of generalized Wannier functions for non-periodic systems in two dimensions and higher. arxiv:2003.06676 (2020)
38. Vecharynski, E., Yang, C., Pask, J.E.: A projected preconditioned conjugate gradient algorithm for computing many extreme eigenpairs of a Hermitian matrix. J. Comput. Phys. **290**, 73–89 (2015)
39. Wang, Z., Li, Y., Lu, J.: Coordinate descent full configuration interaction. J. Chem. Theory Comput. **15**(6), 3558–3569 (2019)
40. Wen, Z., Yang, C., Liu, X., Zhang, Y.: Trace-penalty minimization for large-scale eigenspace computation. J. Sci. Comput. **66**(3), 1175–1203 (2016)
41. Wen, Z., Yin, W.: A feasible method for optimization with orthogonality constraints. Math. Program. **142**(1–2), 397–434 (2013)
42. Yu, V. W.-Z., Campos, C., Dawson, W., García, A., Havu, V., Hourahine, B., Huhn, W. P., Jacquelin, M., Jia, W., Keçeli, M., Laasner, R., Li, Y., Lin, L., Lu, J., Moussa, J., Roman, J. E., Vázquez-Mayagoitia, Á., Yang, C., Blum, V.: ELSI – an open infrastructure for electronic structure solvers. arxiv:1912.13403 (2019)
43. Yu, V.W.-Z., Corsetti, F., García, A., Huhn, W.P., Jacquelin, M., Jia, W., Lange, B., Lin, L., Lu, J., Mi, W., Seifitokaldani, A., Vázquez-Mayagoitia, Á., Yang, C., Yang, H., Blum, V.: ELSI: a unified software interface for Kohn-Sham electronic structure solvers. Comput. Phys. Commun. **222**, 267–285 (2018)
44. Zhang, X., Zhu, J., Wen, Z., Zhou, A.: Gradient type optimization methods for electronic structure calculations. SIAM J. Sci. Comput. **36**(3), C265–C289 (2014)
45. Zhou, Y., Saad, Y., Tiago, M.L., Chelikowsky, J.R.: Self-consistent-field calculations using Chebyshev-filtered subspace iteration. J. Comput. Phys. **219**(1), 172–184 (2006)