SIAM J. SCI. COMPUT. © 2018 Society for Industrial and Applied Mathematics
Vol. 40, No. 5, pp. A3157–A3179

ACCELERATED RESIDUAL METHODS FOR THE ITERATIVE SOLUTION OF SYSTEMS OF EQUATIONS[*]

N. C. NGUYEN[†], P. FERNANDEZ[†], R. M. FREUND[‡], AND J. PERAIRE[†]

Abstract. We present accelerated residual methods for the iterative solution of systems of equations by leveraging recent developments in accelerated gradient methods for convex optimization. The stability properties of the proposed method are analyzed for linear systems of equations by using the finite difference equation theory. Next, we introduce a residual descent restarting strategy and an adaptive computation of the acceleration parameter to enhance the robustness and efficiency of our method. Furthermore, we incorporate preconditioning techniques into the proposed method to accelerate its convergence. We demonstrate the performance of our method on systems of equations resulting from the finite element approximation of linear and nonlinear partial differential equations. In a variety of test cases, the numerical results show that the proposed method is competitive with the pseudo–time-marching method, Nesterov's method, and Newton–Krylov methods. Finally, we discuss some open issues that should be addressed in future research.

Key words. accelerated residual methods, accelerated first-order methods, nonlinear systems, linear systems, partial differential equations

AMS subject classifications. 65H10, 65N22, 65F10

DOI. 10.1137/17M1141369

1. Introduction.

In this paper, we are concerned with the iterative solution of a general square system of equations of the form

$$\boldsymbol{f}(\boldsymbol{u}^*) = 0. \tag{1}$$

Here $\boldsymbol{f} = (f_1(\boldsymbol{u}), \ldots, f_N(\boldsymbol{u})) \in \mathbb{R}^N$ is a vector-valued function of the vector $\boldsymbol{u} = (u_1, \ldots, u_N) \in \mathbb{R}^N$, where N denotes the dimension of the system. The system (1) often arises when numerically solving partial differential equations (PDEs) or unconstrained optimization problems (by solving $\boldsymbol{f}(\boldsymbol{u}^*) := \nabla c(\boldsymbol{u}^*) = 0$ for some objective function $c(\cdot)$). The development of iterative methods for solving the system of equations (1) is a very important part of applied mathematics because it has a wide range of applications in engineering and science. Numerous methods have been developed and studied in the literature.

For a linear system of equations, $\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{A}\boldsymbol{u} - \boldsymbol{b}$, where $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ is a matrix and $\boldsymbol{b} \in \mathbb{R}^N$ is a vector, classical iterative methods such as Jacobi, Richardson, and Gauss–Seidel methods have been used with some success. A generalization of the Gauss–Seidel method led to the successive overrelaxation (SOR) method devised by Young and Frankel (see [27]). An alternative to classical iterative methods are Krylov subspace methods. One such scheme is the conjugate gradient (CG) method, developed by Hestenes and Stiefel [11]. The CG method is particularly suited for linear

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section July 31, 2017; accepted for publication (in revised form) July 16, 2018; published electronically September 25, 2018.
http://www.siam.org/journals/sisc/40-5/M114136.html

Funding: This work was supported by AFOSR grants FA9550-15-1-0276 and FA9550-16-1-0214. The second author's research was partially supported by a "la Caixa" Foundation Graduate Studies Fellowship. The third author's research was supported by the MIT-Belgium Université Catholique de Louvain Fund.

[†]MIT Department of Aeronautics and Astronautics, Cambridge, MA 02139 (cuongng@mit.edu, pablof@mit.edu, peraire@mit.edu).
[‡]MIT Sloan School of Management, Cambridge, MA 02139 (rfreund@mit.edu).

systems whose matrix is symmetric and positive-definite. Other Krylov methods for linear systems include CGS [23], BiCGSTAB [26], MINRES [20], GMRES [21], and QMR [8], to name a few.

The most well known method for solving nonlinear systems of equations is Newton's method. Newton's method generates a sequence of linear systems, and Krylov methods are often used to solve them, resulting in the so-called Newton–Krylov methods. For many problems, computing the exact Jacobian, as required by Newton's method, can be computationally expensive. Instead of constructing the exact Jacobian matrix at every iteration, quasi-Newton methods such as the BFGS method [5, 7, 9, 22] approximate it using low-rank updates. A popular alternative approach is Jacobian-free Newton–Krylov methods [4, 6, 12], in which the matrix-vector products in the Krylov iteration are computed without explicitly forming the Jacobian matrix, such as via finite difference approximation and residual evaluations. Fixed-point methods such as the Richardson iteration can also be used to solve nonlinear systems of equations without having to compute the Jacobian matrix. The Richardson method is very simple to implement, and the cost of each iteration is very low. However, a major drawback of the Richardson method is that it may not converge, or if it does, it may suffer from slow convergence. Multigrid methods [2, 3, 10, 13] have also been widely used to solve linear and nonlinear systems arising from the discretization of partial differential equations (PDEs). However, multigrid methods are not always applicable to general systems of equations arising from other contexts.

In optimization theory, the optimality condition of an unconstrained minimization of a differentiable objective function results in the problem (1). Iterative methods mentioned in the preceding paragraph can be used to solve unconstrained optimization problems as well. Newton's method and the BFGS method are second-order methods because they use both the first and second derivatives of the objective function, whereas gradient descent methods are first-order methods. Second-order methods converge in many fewer iterations than first-order methods, but the computational cost per iteration of the former is significantly higher than that of the latter. It is known that when a gradient descent method is applied to convex optimization problems, the provable convergence rate of the error in the objective function is $O(1/k)$ [16], where k is the number of iterations.

In a seminal paper published in 1983 [15], Nesterov proposed an accelerated gradient method that exhibits the worst-case convergence rate of $O(1/k^2)$ for minimizing smooth convex functions. Nesterov's method is an optimal gradient method for convex optimization problems because no method based solely on first-order information can achieve a faster convergence rate than $O(1/k^2)$ [14]. Since the introduction of Nesterov's method, there has been much work on the development of first-order accelerated methods; see [17, 18, 16] for theoretical developments and [25] for a unified analysis of these ideas. One drawback of Nesterov's method is that it does not converge smoothly but exhibits oscillatory behavior that can severely slow its convergence. To remedy this problem, O'Donoghue and Candès [19] propose a simple restarting technique to improve the convergence rate of Nesterov's method. Recently, Su, Boyd, and Candès [24] showed that Nesterov's method can be interpreted as a finite difference approximation of a second-order ordinary differential equation (ODE), thereby providing a better understanding of Nesterov's method. More recently, Attouch and Peypouquet [1] extended the ODE theory of the work [24] to show that the convergence rate of Nesterov's method is actually $o(1/k^2)$, rather than $O(1/k^2)$.

The main contributions of this paper are as follows. First, we extend the ideas of accelerated gradient methods for convex optimization to solve linear and nonlinear systems of equations. These methods will be referred to as *accelerated residual methods* because, in general, the vector $\boldsymbol{f}(\cdot)$ is not the gradient but the residual vector. Second, we present an interpretation of existing accelerated methods as finite difference approximations (FDAs) to a second-order ODE and provide a linear stability analysis for various accelerated schemes. Based on the stability analysis, we introduce a residual descent restarting strategy and an adaptive computation of the acceleration parameter to enhance the robustness and efficiency of our method. Furthermore, we incorporate preconditioning techniques into the proposed method to improve its convergence.

We apply the proposed method as well as other iterative methods to solve systems of equations resulting from the finite element approximation of linear and nonlinear PDEs. The following empirical conclusions can be drawn from the results of our numerical experiments. First, the proposed method converges much faster and thus requires significantly fewer iterations than Nesterov's method. Second, it is found out that restarting is key to improving the convergence rate and robustness of any accelerated methods. For problems in which the systems are nonlinear and nonconvex, they often do not converge without restarting. Third, our choice of the acceleration parameter performs better than the standard value proposed by Nesterov since it reduces the number of iterations required for convergence to the same error tolerance. And fourth, our method is competitive with Newton–Krylov methods for solving nonlinear systems, while requiring much less memory storage.

The paper is organized as follows. In section 2, we give an overview of the development of accelerated gradient methods. In section 3, we propose an accelerated residual method for solving systems of equations and generalize it to encompass a family of methods. In section 4, we present numerical results to demonstrate the method on both linear and nonlinear systems resulting from the finite element approximation of PDEs. Finally, in section 5, we discuss some open issues that should be addressed in future research.

2. Overview of accelerated gradient methods.

To put our contributions in perspective, we present a brief overview of the past and recent developments of accelerated gradient methods for convex optimization.

2.1. Nesterov's method.

Before discussing Nesterov's method, we consider the simplest fixed-point iteration for solving the system of equations (1):

$$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k - \alpha_k \boldsymbol{f}(\boldsymbol{u}_k), \qquad k \geq 0, \tag{2}$$

where α_k is a stability parameter. The iteration is stopped if the following criterion is met:

$$\|\boldsymbol{f}(\boldsymbol{u}_k)\| < \epsilon, \tag{3}$$

where ϵ is a given error tolerance. Note that the above fixed-point iteration is nothing but the forward Euler method applied to the following ODE system:

$$\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0, \qquad \boldsymbol{u}(t=0) = \boldsymbol{u}_0, \tag{4}$$

in which the time-step size is set to the stability parameter α_k. Hence, the fixed-point iteration (2) will be referred to as the pseudo–time-marching method. Although this method is very simple and cost-effective, it can suffer from extremely slow convergence because of the severe restriction on the stability parameter α_k.

In recent years, there has been considerable interest in accelerated first-order methods driven primarily by their optimal performance for solving convex optimization problems. These methods were first developed by Nesterov in the context of convex optimization and subsequently extended by many other researchers. Nesterov's method for solving the system (1) takes the following form: Starting with $\boldsymbol{u}_0$ and $\boldsymbol{u}_1 = \boldsymbol{u}_0 - \alpha_0 \boldsymbol{f}(\boldsymbol{u}_0)$, we perform the following iteration:

(5a) $$\boldsymbol{v}_k = \boldsymbol{u}_k + \beta_k(\boldsymbol{u}_k - \boldsymbol{u}_{k-1}),$$

(5b) $$\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{f}(\boldsymbol{v}_k).$$

The first step (5a) is an extrapolation step in which $\boldsymbol{v}_k$ is determined from the two previous iterates $\boldsymbol{u}_{k-1}$ and $\boldsymbol{u}_k$. The second step (5b) is a solution update in which the next iterate $\boldsymbol{u}_{k+1}$ is computed using $\boldsymbol{v}_k$ instead of $\boldsymbol{u}_k$. The parameter α_k is typically chosen to be the same as the one in the fixed-point iteration (2).

The acceleration parameter β_k is crucial because it has a significant impact on the convergence of Nesterov's method. In the original paper [15], Nesterov suggested setting $\beta_k = k/(k+3)$. It is recommended to choose $\alpha_k \leq \alpha \equiv 1/L$, where the Lipschitz constant L of $\boldsymbol{f}(\cdot)$ is defined as follows:

(6) $$\|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\| \qquad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^N,$$

assuming that $\boldsymbol{f}(\cdot)$ is Lipschitz continuous.

When being applied to solve a convex optimization problem $\min_{\boldsymbol{x} \in \mathbb{R}^N} c(\boldsymbol{x})$ with smooth objective function $c(\cdot)$, Nesterov's method exhibits the worst-case convergence rate [15]:

(7) $$c(\boldsymbol{u}_k) - c(\boldsymbol{u}^*) \leq \frac{2\|\boldsymbol{u}_0 - \boldsymbol{u}^*\|^2}{\alpha(k+1)^2}.$$

Here $\boldsymbol{u}^*$ is a minimizer of the convex function $c(\boldsymbol{u})$, which is also a solution of the system (1). It is well known that the convergence rate $O(1/k^2)$ is optimal for convex optimization among all methods using only information about the gradient [15]. This is in contrast to the fixed-point method (2), which has the same computational cost per step but can only achieve the worst-case convergence rate of $O(1/k)$.

In a recent paper [24], Su, Boyd, and Candès showed that Nesterov's method can be viewed as an FDA of the following second-order ODE system in the limit of an infinitesimal step size:

(8) $$\ddot{\boldsymbol{u}} + \frac{3}{t}\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0,$$

with the initial conditions

(9) $$\boldsymbol{u}(t=0) = \boldsymbol{u}_0, \qquad \dot{\boldsymbol{u}}(t=0) = 0.$$

Moreover, Su, Boyd, and Candès [24] show that the solution of the ODE system (8) satisfies

(10) $$c(\boldsymbol{u}(t)) - c(\boldsymbol{u}^*) \leq \frac{2\|\boldsymbol{u}_0 - \boldsymbol{u}^*\|^2}{t^2}.$$

Note that, since $t \approx (k+1)\sqrt{\alpha_k}$, we recover the convergence rate (7). More recently, Attouch and Peypouquet [1] proved that if $\beta_k = k/(k+\gamma)$ for $\gamma > 3$, then the convergence rate of Nesterov's method is actually $o(1/k^2)$, rather than $O(1/k^2)$. It should be noted that the theoretical results discussed in this section are restricted to convex optimization only.

2.2. Restarting strategies for Nesterov's method. Even for strongly convex optimization problems, Nesterov's method exhibits periodic oscillatory behavior that can degrade its convergence rate [19, 24]. This behavior can be explained by examining the ODE system (8). The second-order time derivative represents acceleration, while the first-order time derivative represents damping. When t is small, the ODE system is overdamped because the ratio $3/t$ is large. This results in a smooth decrease in the objective value. As t increases, the solution of the ODE system becomes more oscillatory because the acceleration term dominates the damping term. This leads to oscillatory convergence in the objective function, as illustrated in Figure 1. To suppress the oscillatory behavior, we can re-solve the ODE system (8) with a new initial solution $\boldsymbol{u}_0$, which is the latest solution before the oscillatory behavior appears. This amounts to restarting Nesterov's method whenever the convergence stagnates or gets worse, as illustrated in Figure 2.

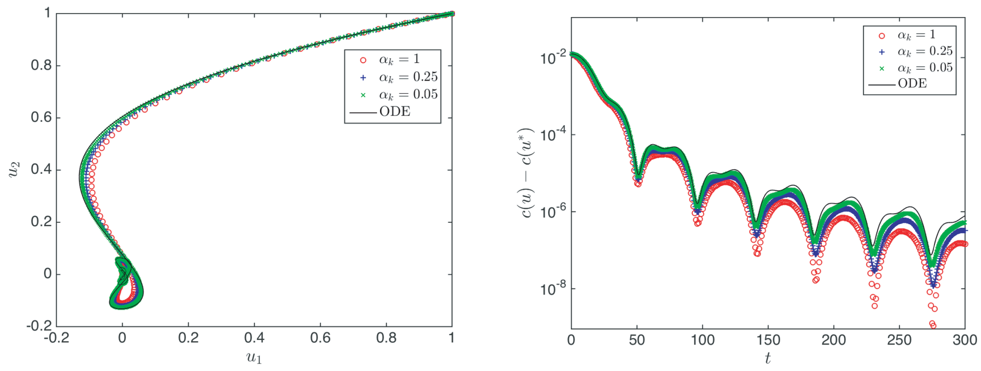

FIG. 1. *Convergence trajectory of Nesterov's method when minimizing the convex function* $c(\boldsymbol{u}) = 0.02u_1^2 + 0.005u_2^2$*:* (left) *trajectory of the iterates and* (right) *convergence of the error in the objective value. This example is taken from* [24].

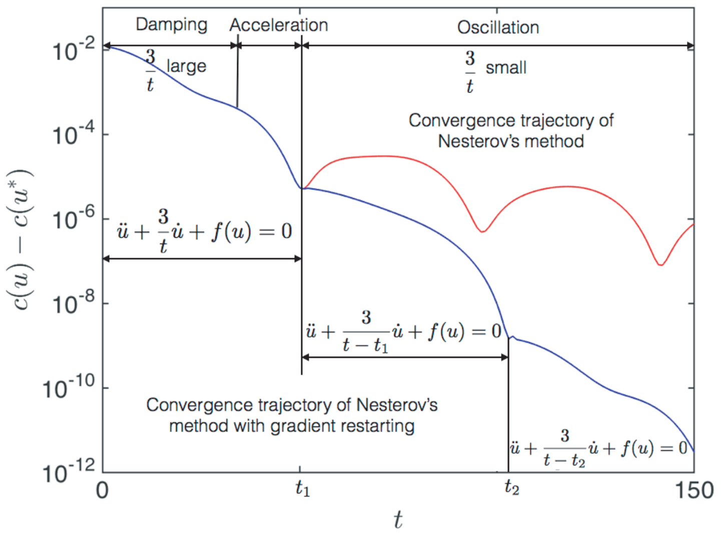

FIG. 2. *Demonstration of the oscillatory convergence behavior of Nesterov's method and the suppression of this behavior by a restarting technique.*

The restarted version of Nesterov's method for solving the system (1) is presented in Algorithm 1. The algorithm needs a convergence criterion to stop the iteration as well as a restarting condition to restart the iteration. One may stop the iteration if $\|\boldsymbol{f}(\boldsymbol{u}_k)\| \leq \epsilon$. However, because this criterion requires an additional evaluation of the residual vector at every iteration, one may want to use a heuristic criterion instead, such as $\|\boldsymbol{f}(\boldsymbol{v}_k)\| \leq \epsilon$.

Algorithm 1 Nesterov's method with restarting.

```
0. Start with $\{\alpha_k, \beta_k\}$, $\boldsymbol{u}_0$, and $k = 0$
1. **Repeat**
2.     $\boldsymbol{v}_k = \begin{cases} \boldsymbol{u}_k, & k = 0 \\ \boldsymbol{u}_k + \beta_k(\boldsymbol{u}_k - \boldsymbol{u}_{k-1}), & k > 0 \end{cases}$
3.     $\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{f}(\boldsymbol{v}_k)$
4.     $k = k + 1$
5.     **If** a convergence criterion is met **then** exit loop
6.     **If** a restarting condition is met **then** reset $\boldsymbol{u}_0 = \boldsymbol{u}_k$ and $k = 0$
7. **End Repeat**
```

Several restarting strategies were proposed in the literature. In a recent paper [19], O'Donoghue and Candès proposed two restarting strategies to improve the convergence rate of Nesterov's method: (1) the *function restarting* restarts whenever $c(\boldsymbol{u}_{k+1}) > c(\boldsymbol{u}_k)$, and (2) the *gradient restarting* restarts whenever $\boldsymbol{f}(\boldsymbol{v}_k)^T(\boldsymbol{u}_{k+1} - \boldsymbol{u}_k) > 0$. Recently, Su, Boyd, and Candès [24] proposed the so-called *speed restarting* that restarts whenever $\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\| > \|\boldsymbol{u}_k - \boldsymbol{u}_{k-1}\|$. It is shown in [24] that speed restarting can ensure linear convergence of the resulting scheme when minimizing strongly convex functions, albeit at a rate that is usually associated with nonaccelerated methods. Figure 3 shows the results obtained by using speed restarting and gradient restarting. We see that both restarting techniques are effective at suppressing the oscillatory convergence of Nesterov's method.

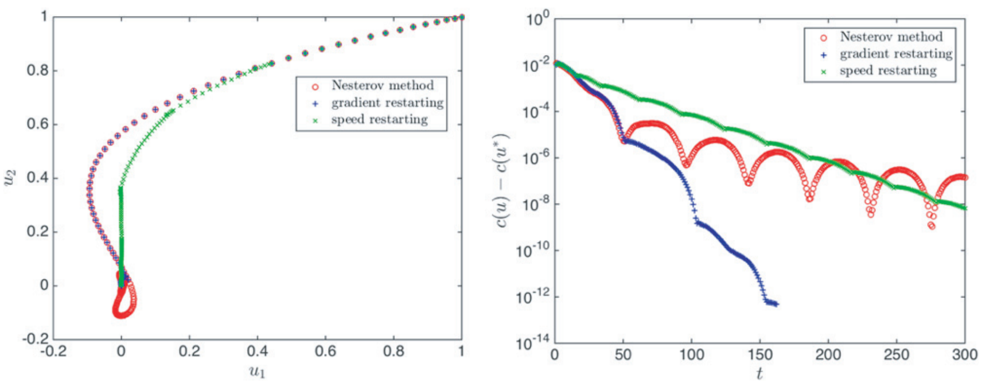

FIG. 3. *Convergence trajectory of Nesterov's method and the restarted Nesterov's method when minimizing the convex function $c(\boldsymbol{u}) = 0.02u_1^2 + 0.005u_2^2$: Trajectory of the iterates on the left and convergence of the error in the objective value on the right.*

3. Accelerated residual methods.

3.1. The basic accelerated residual method.

We introduce the following method for iteratively solving the system (1):

$$\boldsymbol{v}_k = \boldsymbol{u}_k + \beta_k(\boldsymbol{u}_k - \boldsymbol{u}_{k-1}) - \alpha_k(1+\beta_k)\boldsymbol{f}(\boldsymbol{u}_k), \tag{11a}$$

$$\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{f}(\boldsymbol{v}_k). \tag{11b}$$

Compared to Nesterov's method (5), our method has an additional term $-\alpha_k(1+\beta_k)\boldsymbol{f}(\boldsymbol{u}_k)$ in the extrapolation step and thus requires one additional evaluation of the residual vector $\boldsymbol{f}(\boldsymbol{u}_k)$ at every iteration. Nevertheless, as discussed later, our method has some important advantages over Nesterov's method owing to the additional residual evaluation. First, the effective time-step size of our scheme is larger than that of Nesterov's method. Second, we exploit the availability of $\boldsymbol{f}(\boldsymbol{u}_k)$ to define a rigorous convergence criterion and a new restarting strategy. And third, the additional residual evaluation allows for an adaptive selection of the acceleration parameter β_k that can improve the convergence rate of our method. Furthermore, the method can be generalized to a richer family of accelerated residual methods.

3.2. Finite difference approximation.

In this section, we show that the proposed method (11) can be seen as an FDA of a second-order ODE system. To this end, we substitute (11a) into (11b) to arrive at the following equation:

$$\boldsymbol{u}_{k+1} = (\boldsymbol{u}_k + \beta_k(\boldsymbol{u}_k - \boldsymbol{u}_{k-1})) - \alpha_k(1+\beta_k)\boldsymbol{f}(\boldsymbol{u}_k) - \alpha_k \boldsymbol{f}(\boldsymbol{v}_k).$$

Dividing both sides by $\alpha_k(2+\beta_k)$ and rearranging the terms, we obtain

$$\frac{\boldsymbol{u}_{k+1} - 2\boldsymbol{u}_k + \boldsymbol{u}_{k-1}}{\alpha_k(2+\beta_k)} + \frac{1-\beta_k}{\sqrt{\alpha_k(2+\beta_k)}}\frac{(\boldsymbol{u}_k - \boldsymbol{u}_{k-1})}{\sqrt{\alpha_k(2+\beta_k)}} + \left(\frac{1+\beta_k}{2+\beta_k}\boldsymbol{f}(\boldsymbol{u}_k) + \frac{1}{2+\beta_k}\boldsymbol{f}(\boldsymbol{v}_k)\right) = 0.$$

Now suppose that we choose β_k by the following equation:

$$\beta_k = 1 - \frac{\gamma}{k+3} \tag{12}$$

for some $\gamma \geq 0$. Letting $\Delta t_k = \sqrt{\alpha_k(2+\beta_k)}$, we get

$$\frac{\boldsymbol{u}_{k+1} - 2\boldsymbol{u}_k + \boldsymbol{u}_{k-1}}{\Delta t_k^2} + \frac{\gamma}{(k+3)\Delta t_k}\frac{(\boldsymbol{u}_k - \boldsymbol{u}_{k-1})}{\Delta t_k} + \left(\frac{1+\beta_k}{2+\beta_k}\boldsymbol{f}(\boldsymbol{u}_k) + \frac{1}{2+\beta_k}\boldsymbol{f}(\boldsymbol{v}_k)\right) = 0. \tag{13}$$

In the limit $\Delta t_k \to 0$, the above equation is nothing but an FDA of the following second-order ODE system:

$$\ddot{\boldsymbol{u}} + \frac{\gamma}{t}\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0. \tag{14}$$

We see that this γ-parametrized ODE system is exactly the same as the ODE system (8) with $\gamma = 3$. In the particular case $\gamma = 3$, we have from (12) that $\beta_k = k/(k+3)$, which is exactly the same value proposed by Nesterov [15]. Choosing a different value for γ results in a different value for β_k at each iteration.

We observe that the effective time-step size of the proposed method is $\Delta t_k = \sqrt{\alpha_k(2+\beta_k)}$. In [24], it is shown that Nesterov's method (5) can also be interpreted as an FDA of the ODE system (8) with the effective time-step size being equal to $\sqrt{\alpha_k}$. Therefore, the effective time-step size of our method is $\sqrt{2+\beta_k}$ larger than the effective time-step size $\sqrt{\alpha_k}$ of Nesterov's method when the same value of α_k is used for both methods. We thus expect that our scheme converges faster than Nesterov's method in terms of the number of iterations. Since our method requires twice as many residual evaluations per iteration as Nesterov's method, the worst-case convergence rate of our method is slower than that of Nesterov's method by a factor of $2/\sqrt{2+\beta_k}$ when measuring in terms of the number of residual evaluations. However, we observe through the numerical experiments presented in section 4 that our method converges significantly faster than Nesterov's method in terms of the number of residual evaluations.

In practice, the stability parameter α_k can be chosen differently for all methods based on their stability property for a particular problem at hand. For the pseudo–time-marching method and Nesterov's method, it should be chosen such that $\alpha_k \leq 1/L$, where L is the Lipschitz continuity constant in (6). As we shall see in the numerical experiments presented in section 4, the maximum stability-preserving value of α_k for our method can be chosen larger than that for Nesterov's method because our method has a larger stability region than that of Nesterov's method, as discussed next.

3.3. Linear stability analysis. Both Nesterov's method and the accelerated residual method can be interpreted as different FDAs of the same ODE. In this section, we perform a linear stability analysis to determine the stability limit of both schemes in the case of a linear system $\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{A}\boldsymbol{u} - \boldsymbol{b} = \boldsymbol{0}$.

First, we analyze the stability of our method for nonsingular matrix $\boldsymbol{A}$. Upon diagonalization of $\boldsymbol{A}$, the difference equation associated with the accelerated residual method (11) is given by

$$u_{i,k+1} - \left((1+\beta_k) - 2\frac{1+\beta_k}{2+\beta_k}\Delta t^2\lambda_i + \frac{1+\beta_k}{(2+\beta_k)^2}\Delta t^4\lambda_i^2\right)u_{i,k} + \beta_k\left(1 - \frac{1}{2+\beta_k}\Delta t^2\lambda_i\right)u_{i,k-1} = 0.$$

Here $\lambda_i > 0,\ i = 1, \dots, N$, denote the eigenvalues of $\boldsymbol{A}$ and $u_{i,k}$ is the ith component of $\boldsymbol{u}_k$ in the basis of eigenvectors. To simplify the notation, we drop the subscript i and introduce $\eta := \Delta t^2\lambda$ and obtain

$$u_{k+1} - b(\eta,\beta_k)u_k + c(\eta,\beta_k)u_{k-1} = 0, \tag{15}$$

where

$$b(\eta,\beta_k) = (1+\beta_k) - 2\frac{1+\beta_k}{2+\beta_k}\eta + \frac{1+\beta_k}{(2+\beta_k)^2}\eta^2, \qquad c(\eta,\beta_k) = 1 - \frac{1}{2+\beta_k}\eta. \tag{16}$$

The characteristic polynomial of this difference equation is then given by

$$\mathcal{P}^{\mathrm{ARM}}(\xi) = \xi^2 - b(\eta,\beta_k)\xi + c(\eta,\beta_k). \tag{17}$$

The accelerated residual method is stable if the two roots of the characteristic equation $\mathcal{P}^{\mathrm{ARM}}(\xi^*) = 0$ lie within the unit circle of the complex plane. Hence, we obtain the stability region of our method as follows:

$$S^{\mathrm{ARM}}(\beta_k) = \left\{\eta \ : \ \left|b(\eta,\beta_k) \pm \sqrt{b^2(\eta,\beta_k) - 4c(\eta,\beta_k)}\right| \leq 2\right\}. \tag{18}$$

Obviously, the stability region depends on β_k.The stability region of Nesterov's method can be derived in the same way.

Figure 4 show the stability region of both the accelerated residual method and Nesterov's method for various values of β_k. It is interesting to note that the stability

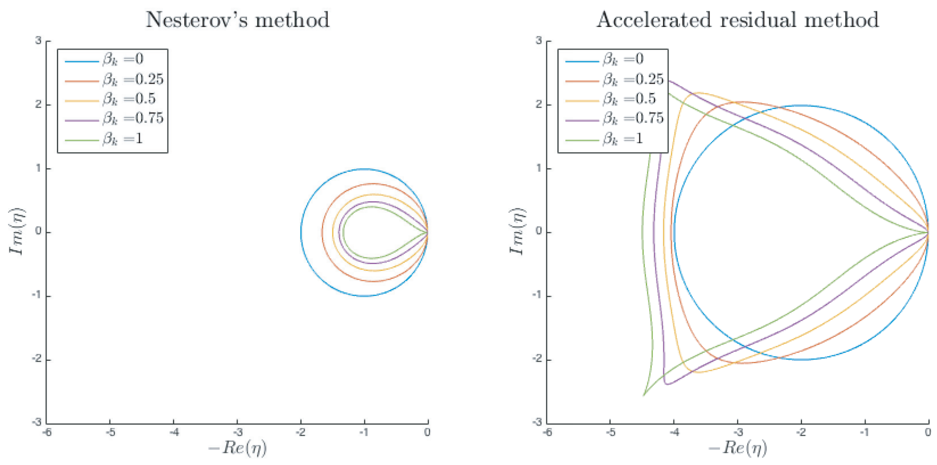

FIG. 4. *Stability regions of Nesterov's method (left) and the accelerated residual method (right) for various values of β_k.*

region of our method expands on the real axis as β_k increases, whereas the stability region of Nesterov's method shrinks as β_k increases. Specifically, for $\beta_k = 1$, the maximum absolute value of η for which our method remains stable is $\eta^{\text{ARM}}_{\text{max,real}} = 9/2$, while that of Nesterov's method is $\eta^{\text{NM}}_{\text{max,real}} = 4/3$. It implies that for symmetric positive-definite systems (i.e., the eigenvalues of $\boldsymbol{A}$ are all real and positive) our method converges faster than Nesterov's method as β_k increases. It also implies that our method is more robust than Nesterov's method in the sense that our method becomes more stable as β_k increases, whereas Nesterov's method becomes less stable. When the eigenvalues of $\boldsymbol{A}$ are complex, the same conclusion can also be made if the real parts dominate the imaginary parts. However, if the imaginary parts dominate the real part, both methods become less stable as β_k increases. Note that both methods are not stable when the eigenvalues are purely imaginary.

It should be pointed out that the stability comparison between Nesterov's method and our method is carried out for the same time step Δt. It means that $\alpha_k = \Delta t^2$ for Nesterov's method and $\alpha_k = \frac{\Delta t^2}{2+\beta_k}$ for our method. This provides insights on how fast both schemes would converge if, given β_k, the maximum step size for stability was known a priori and therefore taken. Because our method has a larger stability region than Nesterov's method, and also larger Δt for given α_k and β_k, it is more appropriate to compare the performance of the two methods for the same value of α_k instead of the same value of Δt. This will be the tactic used in the numerical examples in section 4.

3.4. Residual descent restarting. Here we introduce a restarting strategy and a stopping criterion for our method. The goal is to ensure that the magnitude of the residual is reduced in every iteration and that if it is smaller than a specified tolerance, then the iteration stops. By making use of the already-computed residual vector $\boldsymbol{f}(\boldsymbol{u}_k)$, we can achieve this goal without additional computations. In particular, the iteration stops successfully if $\|\boldsymbol{f}(\boldsymbol{u}_k)\| < \epsilon$ and restarts if

$$\|\boldsymbol{f}(\boldsymbol{u}_k)\| > \|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|. \tag{19}$$

The stopping criterion and restarting strategy do not require extra computation since all the quantities involved are already calculated in our scheme. The resulting iteration is summarized in Algorithm 2. We note that our restarting strategy (19) ensures that the residual norm decreases monotonically throughout the iteration. For this reason, we will refer to it as the accelerated residual descent method (ARDM).

Algorithm 2 Accelerated residual descent method.

```
0. Start with $\{\alpha_k, \beta_k\}$, $\boldsymbol{u}_0$, $\epsilon$, and $k = 0$
1. **Repeat**
2.     $\tilde{\boldsymbol{u}}_k = \boldsymbol{u}_k - \alpha_k \boldsymbol{f}(\boldsymbol{u}_k)$
3.     $\boldsymbol{v}_k = \begin{cases} \tilde{\boldsymbol{u}}_k, & k = 0 \\ \tilde{\boldsymbol{u}}_k - \beta_k(\tilde{\boldsymbol{u}}_k - \boldsymbol{u}_{k-1}), & k > 0 \end{cases}$
4.     $\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{f}(\boldsymbol{v}_k)$
5.     $k = k + 1$
6.     **If** $\|\boldsymbol{f}(\boldsymbol{u}_k)\| < \epsilon$ **then** exit loop
7.     **If** $\|\boldsymbol{f}(\boldsymbol{u}_k)\| > \|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|$ **then** reset $\boldsymbol{u}_0 = \boldsymbol{u}_{k-1}$ and $k = 0$
8. **End Repeat**
```

According to the stability analysis presented earlier, the stability region varies with the acceleration parameter β_k and the time-step size increases with β_k. In particular, the latter effect dominates the former and stability is thus improved by reducing β_k. An increase in the magnitude of the residual from one iteration to the next iteration can be an indication that the time-step size is outside the stability region for the current value of β_k. By restarting the method, we reset the value of β_k to zero and thus increase stability.

It remains to choose α_k and β_k in Algorithm 2. These parameters are crucial to the convergence rate of the method and should be related to mathematical properties of the vector-valued function $\boldsymbol{f}(\boldsymbol{u})$. On the one hand, α_k must be chosen to satisfy the stability condition of the forward Euler method when numerically solving PDEs. As such, α_k depends on both the underlying PDE and the particular numerical discretization. On the other hand, when solving convex optimization problems, α_k must be chosen such that $\alpha_k \leq \alpha \equiv 1/L$, where the Lipschitz continuity constant L is defined in (6). The Lipschitz constant L can be estimated by using backtracking line search. Typically, α_k is set to $1/L$. The computation of β_k will be discussed next.

3.5. Adaptive computation of the acceleration parameter. Instead of using Nesterov's formula $\beta_k = \frac{k}{k+3}$, we propose computing β_k as follows:

$$\beta_k = \frac{\|\boldsymbol{f}(\boldsymbol{u}_k)\|}{\|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|}. \tag{20}$$

Note that this formula together with our restarting strategy (19) ensures $\beta_k \leq 1$. The formula for the acceleration parameter by (20) is motivated by the following observation. When the convergence is relatively slow (i.e., $\|\boldsymbol{f}(\boldsymbol{u}_{k-1})\| \approx \|\boldsymbol{f}(\boldsymbol{u}_k)\|$), β_k is close to 1 to make the convergence faster since the acceleration effect dominates the damping effect. On the other hand, when the convergence is relatively fast, β_k is quite smaller than 1 to make the convergence smooth since adding acceleration promotes the oscillatory convergence in this case. In contrast, Nesterov's formula, $\beta_k = \frac{k}{k+3}$, is an increasing function of k. Hence, as k increases, the acceleration effect becomes more dominant and the stability region of Nesterov's method gets smaller, thereby making the convergence oscillatory. If the formula (20) were used in

Nesterov's method, it would require an additional residual evaluation per iteration, whereas it does not require extra computation for our method since the residual norms are already calculated.

The results shown in Figure 5 illustrate how the proposed formula (20) speeds up the convergence of our method. We see that our method converges faster than both Nesterov's method and its restarted variant. The convergence rate is further improved when switching from $\beta_k = k/(k+3)$ to $\beta_k = \|\boldsymbol{f}(\boldsymbol{u}_k)\|/\|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|$. It is also interesting to point out that using $\beta_k = \|\boldsymbol{f}(\boldsymbol{u}_k)\|/\|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|$ incurs no restarting, whereas using $\beta_k = k/(k+3)$ incurs one restarting. This agrees well with our above analysis.

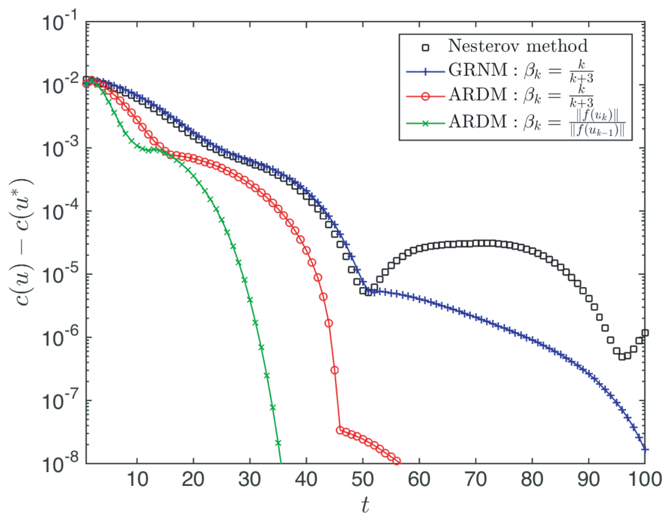

FIG. 5. *Convergence of the error in the objective value $c(\boldsymbol{u}) = 0.02u_1^2 + 0.005u_2^2$ for Nesterov's method, the gradient restarting Nesterov method (GRNM), and the ARDM with $\beta = k/(k+3)$ and $\beta_k = \|\boldsymbol{f}(\boldsymbol{u}_k)\|/\|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|$.*

3.6. Preconditioning the accelerated residual method.

We have devised the residual descent restarting and adaptive computation of the acceleration parameter to improve the convergence rate of our method. In this section, we consider how to further improve the convergence rate through preconditioning techniques. Preconditioning techniques are not widely used in accelerated gradient methods for convex optimization problems because of the prohibitive computational cost in constructing effective preconditioners. However, in the context of the numerical solution of PDEs, preconditioning techniques, such as Jacobi or incomplete LU (ILU), are effective and usually affordable.

We assume that we are given invertible matrices $\boldsymbol{M}_k$ for $k \geq 0$ and that the matrix-vector product of the inverse, $\boldsymbol{M}_k^{-1}\boldsymbol{v}$, is inexpensive to compute. The accelerated residual method with preconditioning is devised as follows:

$$\boldsymbol{v}_k = \boldsymbol{u}_k + \beta_k(\boldsymbol{u}_k - \boldsymbol{u}_{k-1}) - \alpha_k(1+\beta_k)\boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{u}_k), \tag{21a}$$

$$\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{v}_k). \tag{21b}$$

The preconditioning accelerated residual method is presented in Algorithm 3.

Algorithm 3 ARDM with preconditioning.

```
0. Start with $\{\alpha_k, \beta_k, \boldsymbol{M}_k\}$, $\boldsymbol{u}_0$, $\epsilon$, and $k = 0$
1. **Repeat**
2.     $\tilde{\boldsymbol{u}}_k = \boldsymbol{u}_k - \alpha_k \boldsymbol{M}_k^{-1} \boldsymbol{f}(\boldsymbol{u}_k)$
3.     $\boldsymbol{v}_k = \begin{cases} \tilde{\boldsymbol{u}}_k, & k = 0 \\ \tilde{\boldsymbol{u}}_k - \beta_k(\tilde{\boldsymbol{u}}_k - \boldsymbol{u}_{k-1}), & k > 0 \end{cases}$
4.     $\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{M}_k^{-1} \boldsymbol{f}(\boldsymbol{v}_k)$
5.     $k = k + 1$
6.     **If** $\|\boldsymbol{f}(\boldsymbol{u}_k)\| < \epsilon$ **then** exit loop
7.     **If** $\|\boldsymbol{f}(\boldsymbol{u}_k)\| > \|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|$ **then** reset $\boldsymbol{u}_0 = \boldsymbol{u}_{k-1}$ and $k = 0$
8. **End Repeat**
```

It is easy to show that the resulting method (21) is an FDA of the following second-order PDE:

$$\boldsymbol{M}(t)\ddot{\boldsymbol{u}} + \frac{\gamma}{t}\boldsymbol{M}(t)\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0. \tag{22}$$

For the linear case $\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{A}\boldsymbol{u} - \boldsymbol{b}$, the matrix $\boldsymbol{M}$ should be independent of t. In this case, the role of the preconditioning matrix $\boldsymbol{M}$ is to improve the spectra of the matrix $\boldsymbol{M}^{-1}\boldsymbol{A}$, thereby allowing us to take a much larger time-step size according to the linear stability analysis. For the nonlinear case, instead of updating the preconditioning matrix at every iteration, we update it only when the restarting occurs. Furthermore, the preconditioning matrix can be computed as the block Jacobi or ILU of the Jacobian matrix evaluated for the solution at the restarting.

4. Numerical experiments.

In this section, we demonstrate and compare the performance of various iterative methods on a wide variety of problems arising from the continuous Galerkin finite element approximation of the Poisson equation, a linear convection-diffusion equation, a nonlinear elliptic equation, and a nonlinear convection-diffusion equation. For simplicity of exposition, we will take $\boldsymbol{M}$ to be an identity matrix in sections 4.1, 4.2, 4.3, and 4.4. The performance of the preconditioned version of these methods is investigated in section 4.5. We will set $\beta_k = k/(k+3)$ for Nesterov's method and its gradient restarting variant, as well as $\beta_k = \|\boldsymbol{f}(\boldsymbol{u}_k)\|/\|\boldsymbol{f}(\boldsymbol{u}_{k-1})\|$ for the ARDM. We will also set α_k to a fixed value α.

We will evaluate the performance of all methods based on the norm of the residual vector as a function of the number of residual evaluations. We will set the error tolerance as $\epsilon = 10^{-8}$. A method is said to perform better than the other in a particular example if the former requires a smaller number of residual evaluations than the latter to converge to the prescribed tolerance.

While we will focus on systems of equations arising from the numerical discretization of PDEs, the proposed method can be used to solve systems of equations in other contexts. For instance, like Nesterov's method, the proposed method can be used to solve systems of equations arising from the optimality condition of unconstrained optimization problems. Similarly, Jacobi and ILU preconditioners will be used as a means of demonstrating how preconditioning works with the proposed method in comparison with the performance of preconditioned Newton–Krylov methods.

4.1. Poisson equation. In the first example, we consider the following Poisson problem:

$$-\Delta u = 2\pi^2 \sin(\pi x)\sin(\pi y) \qquad \text{in } \Omega = (0,1)\times(0,1) \tag{23}$$

with a Dirichlet boundary condition $u = 0$ on $\partial\Omega$. The finite element discretization leads us to solving the following linear system:

$$\boldsymbol{A}\boldsymbol{u}^* = \boldsymbol{b}, \tag{24}$$

where the stiffness matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ is a symmetric positive-definite (SPD) matrix. The solution of this linear system is the minimizer of the following convex minimization problem:

$$\min_{\boldsymbol{v}\in\mathbb{R}^N} \frac{1}{2}\boldsymbol{v}^T\boldsymbol{A}\boldsymbol{v} - \boldsymbol{v}^T\boldsymbol{b}. \tag{25}$$

The gradient vector of the objective function is thus the same as the residual vector $\boldsymbol{f}(\boldsymbol{v}) = \boldsymbol{A}\boldsymbol{v} - \boldsymbol{b}$. The finite element approximation employs n_{elem} uniform triangular elements of the polynomial degree p. Hence, the problem size N depends on the polynomial degree p and the number of elements n_{elem}.

Let us now assess the methods described herein for the solution of this problem and compare their performance with that of the CG method. Figure 6 shows the performance of all the methods for different values of $(p, n_{\text{elem}}, \alpha)$. There are a number of interesting observations. We see that the number of residual evaluations required for convergence to a given tolerance increases with the polynomial degree and the number of elements due to the increase in the condition number $L/\mu = \lambda_{\max}/\lambda_{\min}$. Although Nesterov's method exhibits the periodic behavior discussed earlier, it converges much faster than the forward Euler method. The gradient restarting Nesterov method yields a significant improvement in the convergence rate. Our method converges much faster than Nesterov's method and its gradient restarting variant in all cases. Nesterov's method and its gradient restarting variant diverge when $p = 3$ and $\alpha = 0.1$, whereas our method converges in those cases. This is consistent with the improved stability of our scheme, as shown in section 3.3. Finally, the CG method has the best performance for this particular problem, where the Jacobian matrix is SPD. However, the CG method does not converge for the next problem, where the Jacobian matrix is not SPD.

4.2. Linear convection-diffusion problem. In the second example, we consider a linear convection-diffusion problem of the form

$$-\Delta u + c_x\frac{\partial u}{\partial x} + c_y\frac{\partial u}{\partial y} = 10 \qquad \text{in } \Omega = (0,1)\times(0,1) \tag{26}$$

with a Dirichlet boundary condition $u = 0$ on $\partial\Omega$, where (c_x, c_y) is the convective velocity field. We use the continuous Galerkin finite element method to discretize the above equation on a uniform mesh of $n_{\text{elem}} = 800$ triangular elements and polynomial degree $p = 3$. This spatial discretization results in the following linear system:

$$\boldsymbol{A}\boldsymbol{u}^* = \boldsymbol{b}, \tag{27}$$

where $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ is a *nonsymmetric* matrix whenever the convective velocity is nonzero. Figure 7 depicts the numerical solution for $(c_x, c_y) = (1,1)$ and $(c_x, c_y) =$

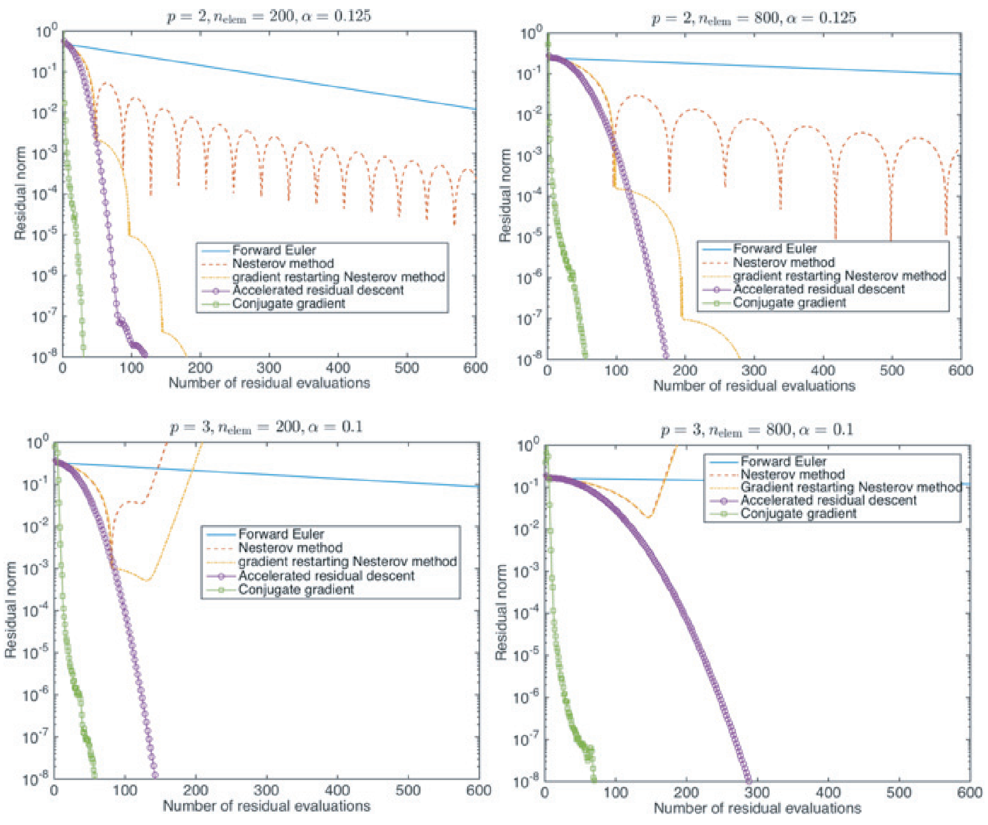

FIG. 6. *Comparison among the iterative methods for solving the Poisson problem.*

$(50, 50)$, which correspond to a diffusion-dominated case and a convection-dominated case, respectively.

The solution of the above linear system satisfies the following minimization problem:

$$\min_{\boldsymbol{v} \in \mathbb{R}^N} \frac{1}{2}(\boldsymbol{A}\boldsymbol{v} - \boldsymbol{b})^T(\boldsymbol{A}\boldsymbol{v} - \boldsymbol{b}). \tag{28}$$

The gradient vector of the objective function is given by

$$\boldsymbol{g}(\boldsymbol{v}) = \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{v} - \boldsymbol{A}^T\boldsymbol{b}, \tag{29}$$

which differs from the residual vector $\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{A}\boldsymbol{u} - \boldsymbol{b}$. Note that Nesterov's method is intended to solve the normal equation $\boldsymbol{g}(\boldsymbol{u}^*) = 0$ instead of the residual equation $\boldsymbol{f}(\boldsymbol{u}^*) = 0$. When applied to the normal equation $\boldsymbol{g}(\boldsymbol{u}^*) = 0$, the maximum step size for stability reduces dramatically due to the squaring of the Lipschitz constant and Nesterov's method becomes impractical. Furthermore, since the matrix $\boldsymbol{A}$ is not SPD, the CG method is not suited to solve this problem. The CG method on the normal equations (CGN) and the generalized minimum residual (GMRES) method are used instead.

Figures 8 and 9 show the performance of all the methods for the diffusion-dominated case $c_x = c_y = 1$ and the convection-dominated case $c_x = c_y = 50$,

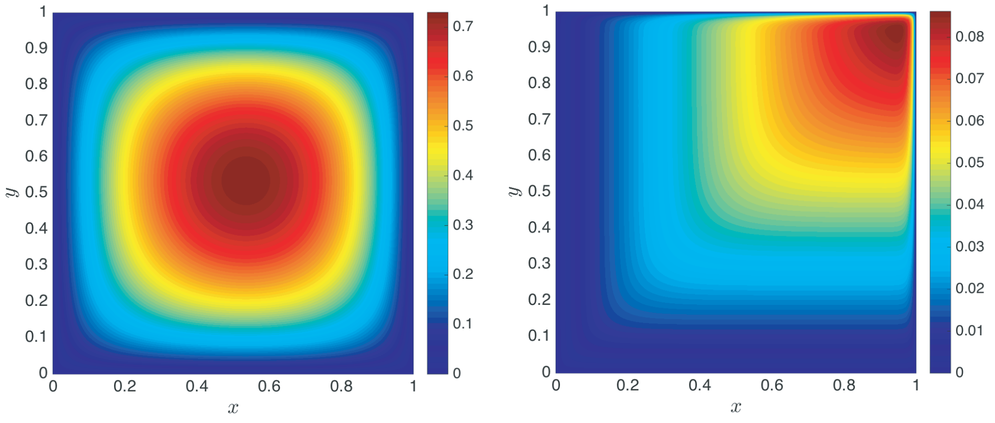

FIG. 7. *Numerical solution of the linear convection-diffusion problem for* $(c_x, c_y) = (1, 1)$ *(left) and* $(c_x, c_y) = (50, 50)$ *(right).*

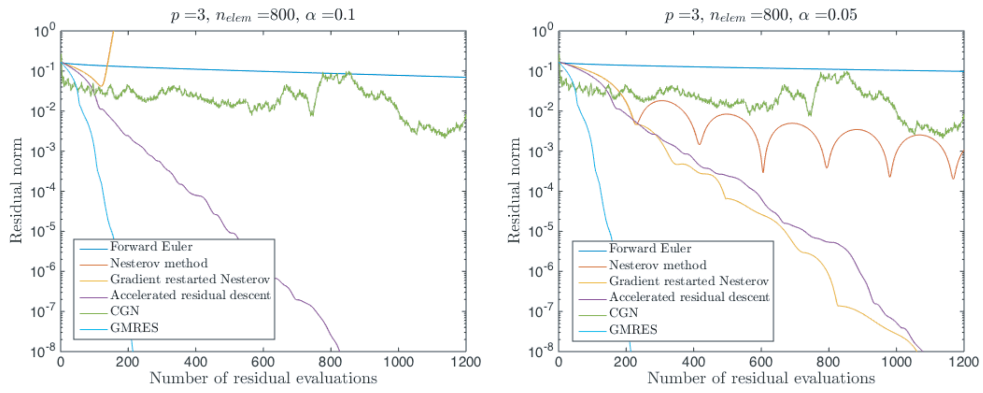

FIG. 8. *Comparison among the iterative methods for solving for the convection-diffusion equation with* $c_x = c_y = 1$.

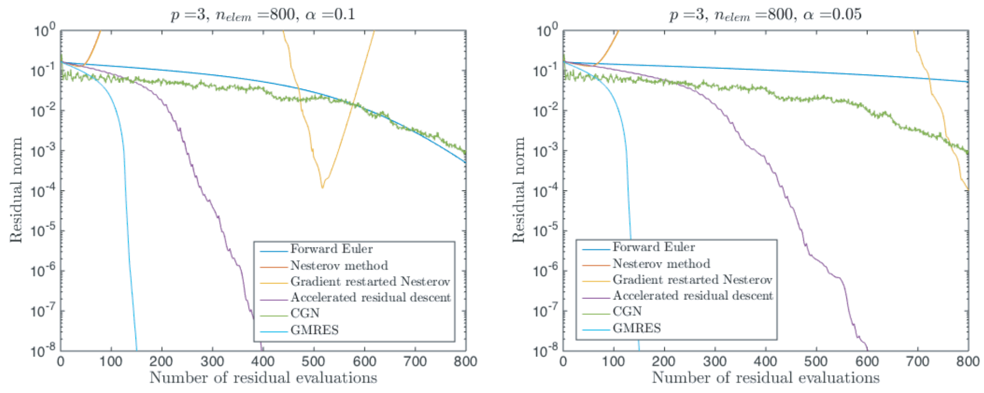

FIG. 9. *Comparison among the iterative methods for solving for the convection-diffusion equation with* $c_x = c_y = 50$.

respectively. In these figures, Nesterov's method is applied to the residual equation. (When applied to the normal equation, Nesterov's method rapidly diverges for all the values of α considered.) We see that our method converges, while Nesterov's method and its gradient restarting variant diverge when α is set to 0.1. When α is reduced to 0.05, Nesterov's method and its gradient restarting variant are able to converge in the diffusion-dominated case. In this case, the gradient restarting Nesterov method converges a little faster than our method. Our method is less sensitive to the variation of the stability parameter α and more robust than Nesterov's method.

Furthermore, it is interesting to note that the performance of our method is better in the convection-dominated case than in the diffusion-dominated case. In particular, our method requires roughly 400 residual evaluations to converge to the error tolerance 10^{-8} in the convection-dominated case, whereas it requires more than 800 residual evaluations in the diffusion-dominated case. This is attributed to the scheme being stable for both regimes and the underlying ODE converging faster to the solution in the convection-dominated case, as illustrated by the forward Euler method. This is in contrast to Nesterov's method, which is stable in the diffusion-dominated regime and unstable in the convection-dominated case. It is consistent with the stability region plotted in Figure 4.

GMRES has the best performance among all the methods. We emphasize that our method is capable of solving both linear and nonlinear systems, whereas GMRES is designed to solve linear systems. In order to solve nonlinear systems, GMRES is coupled with Newton's method, giving rise to the so-called Newton-GMRES method. In the next two examples, we will compare our method against the Newton-GMRES and Newton-CGN methods for solving nonlinear PDEs.

4.3. Nonlinear elliptic problem. In the third example, we consider a nonlinear elliptic problem of the form

$$-\nabla \cdot \big((1+u^2)\nabla u\big) + u = 4\pi^2 \qquad \text{in } \Omega = (0,1)\times(0,1) \tag{30}$$

with a Dirichlet boundary condition $u = 0$ on $\partial\Omega$. The weak formulation of the finite element method is to find $u_h \in V_h$ such that

$$\int_\Omega \big((1+u_h^2)\nabla u_h \cdot \nabla v + u_h v\big)\, dxdy - 4\pi^2 \int_\Omega v dxdy = 0 \qquad \forall\, v \in V_h, \tag{31}$$

where $V_h = \{v \in C_0(\Omega) \ : \ v|_K \in \mathcal{P}^3(K)\ \forall K \in \mathcal{T}_h, \text{ and } v = 0 \text{ on } \partial\Omega\}$. Here $\mathcal{T}_h$ is a finite element mesh of 200 triangular elements and $\mathcal{P}^3(K)$ is the space of polynomials of degree at most 3 on K. The weak formulation (31) is equivalent to the following nonlinear system:

$$\boldsymbol{f}(\boldsymbol{u}^*) = 0, \tag{32}$$

where $\boldsymbol{u}^* \in \mathbb{R}^N$ is the vector of degrees of freedom of u_h, and $\boldsymbol{f} \in \mathbb{R}^N$ is the residual vector whose entries are obtained by setting v in the weak formulation (31) to the basis functions of the space V_h. The initial vector $\boldsymbol{u}_0$ is set to the numerical solution of a linear elliptic problem in which the nonlinear term u^2 is removed from (30).

Figure 10 shows the convergence of various schemes for $\alpha = 0.015$. We observe that the ARDM converges slightly better than the gradient restarting Nesterov method and that the two methods perform much better than the forward Euler method and Nesterov's method. They take more than 600 residual evaluations to converge to a tolerance of $\epsilon = 10^{-8}$, while the forward Euler method and Nesterov's

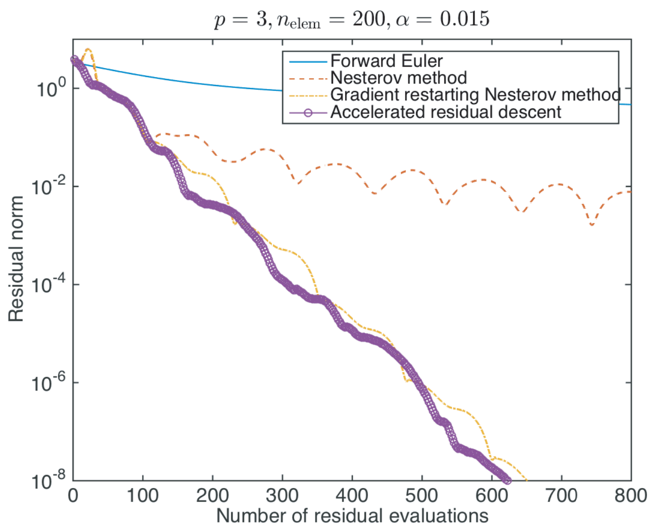

FIG. 10. *Performance of the four different schemes for the nonlinear elliptic problem.*

TABLE 1
Convergence of Newton-CGN and Newton-GMRES for nonlinear elliptic problem.

Newton iteration	Residual norm	# CGN iterations	# GMRES iterations
1	3.9017×10^{0}	841	130
2	9.7774×10^{-1}	841	113
3	1.3988×10^{-1}	712	107
4	4.1785×10^{-3}	676	107
5	3.6006×10^{-6}	457	106
6	2.1354×10^{-12}	no convergence	129
Total		–	692

method take many thousands of residual evaluations. Table 1 shows the convergence of the Newton-CGN and Newton-GMRES methods. In each Newton iteration, CGN and GMRES are used to solve a linear system resulting from the linearization of the weak formulation (30). The number of CGN and GMRES iterations required for the prescribed tolerance of 10^{-8} is tabulated in Table 1. Newton-CGN fails to converge at the sixth Newton iteration because the condition number of the normal equation becomes too large. Newton-GMRES, however, converges to a residual norm of 2.1354×10^{-12} after six Newton iterations with 692 GMRES iterations/matrix-vector products. The Newton-CGN and Newton-GMRES methods commonly require computing and storing the Jacobian matrix, which adds a significant overhead to the computational cost and memory requirements. Furthermore, even with the use of Jacobian-free methods to avoid the construction of the Jacobian matrix and calculation of matrix-vector products, GMRES still requires the orthogonalization and storage of the Krylov vectors. Our method is thus more efficient than the Newton-CGN and Newton-GMRES methods in this particular example.

4.4. Nonlinear convection-diffusion problem. In the last example, we consider solving a Burgers problem

$$-\nabla \cdot (\kappa \nabla u) + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = s \qquad \text{in } \Omega = (0,1)\times(0,1) \tag{33}$$

with $u = 0$ on $\partial\Omega$ and $\kappa = 0.02$. The source term s is specified such that the problem has the following exact solution:

$$u = xy \tanh\left(\frac{1-x}{\kappa}\right)\tanh\left(\frac{1-y}{\kappa}\right). \tag{34}$$

The weak formulation of the finite element method is to find $u_h \in V_h$ such that

$$\int_\Omega \left(\kappa \nabla u_h \cdot \nabla v + u_h \frac{\partial u_h}{\partial x} v + u_h \frac{\partial u_h}{\partial y} v\right) dxdy - \int_\Omega s v dxdy = 0 \qquad \forall\, v \in V_h, \tag{35}$$

where $V_h = \{v \in C_0(\Omega) \ : \ v|_K \in \mathcal{P}^3(K)\ \forall K \in \mathcal{T}_h, \text{ and } v = 0 \text{ on } \partial\Omega\}$ and $\mathcal{T}_h$ is the finite element mesh of 1800 triangular elements. The initial solution $\boldsymbol{u}_0$ is set to the solution of a linear elliptic problem in which the nonlinear convective terms are removed from (33).

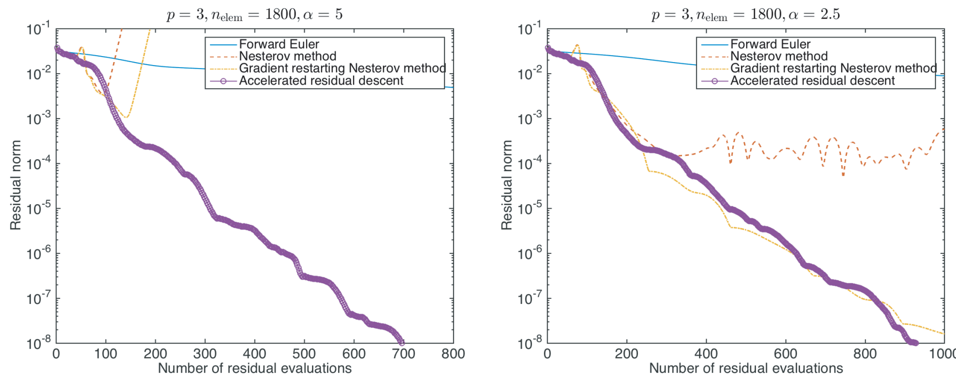

FIG. 11. *Performance of the four different schemes for the Burgers problem.*

Figure 11 shows the convergence of the four different schemes for two different values of the stability parameter α. We observe that the ARDM needs 700 residual evaluations to converge to a tolerance of $\epsilon = 10^{-8}$ for $\alpha = 5$, while both Nesterov's method and its gradient restarting variant do not converge. When α is reduced from 5 to 2.5, the gradient restarting Nesterov method converges to the said tolerance with more than 1000 residual evaluations, while Nesterov's method still does not converge. Table 2 shows the convergence of the Newton-CGN and Newton-GMRES methods, together with the number of CGN and GMRES iterations required for solving the linear system with a prescribed tolerance of 10^{-8}. Again, Newton-CGN fails to converge at the fifth Newton iteration. The Newton-GMRES method, however, converges to a residual norm of 4.0558×10^{-10} after six Newton iterations, requiring a total of 1461 matrix-vector products. This is roughly two times more than the number of residual evaluations of the ARDM. In this example, our method outperforms the gradient restarting Nesterov method, the Newton-CGN method, and the Newton-GMRES method.

TABLE 2
Convergence of Newton-CGN and Newton-GMRES for the Burgers problem.

Newton iteration	Residual norm	# CGN iterations	# GMRES iterations
1	3.8045×10^{-2}	3135	229
2	8.8212×10^{-3}	3862	233
3	1.5580×10^{-3}	3529	234
4	1.2478×10^{-4}	3278	249
5	1.6119×10^{-6}	no convergence	253
6	4.0558×10^{-10}		263
Total		–	1461

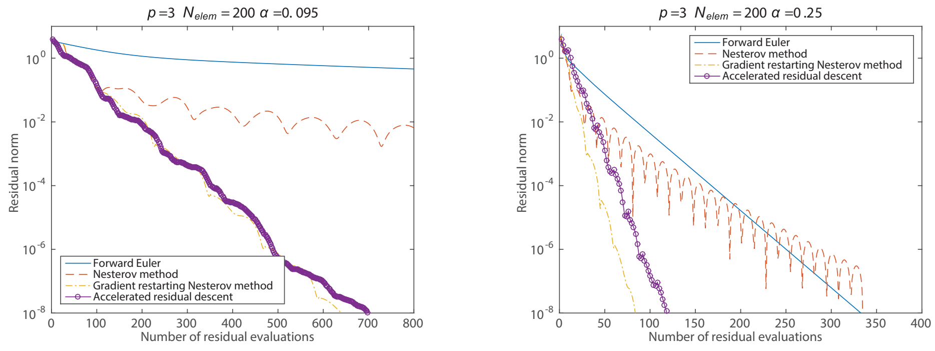

FIG. 12. *Performance of the four residual schemes for the nonlinear elliptic problem with Jacobi (left) and ILU(0) (right) preconditioners.*

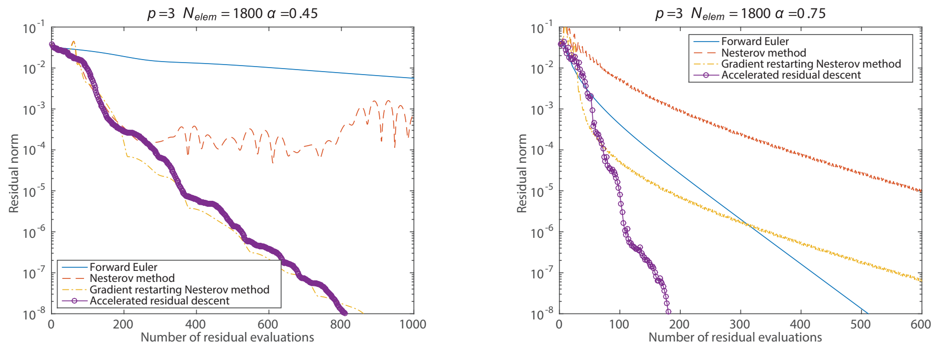

FIG. 13. *Performance of the four residual schemes for the Burgers problem with the Jacobi conditioner (left) and the ILU(0) preconditioner (right).*

4.5. Effect of preconditioning. In this section, we investigate the effect of preconditioning on the ARDM and other methods. Jacobi and incomplete LU factorization with zero fill-in, ILU(0), preconditioners are considered. More specifically, for all methods, we take $\boldsymbol{M}_k$ to be the diagonal (for the Jacobi preconditioner) or the ILU(0) factorization (for the ILU preconditioner) of the Jacobian matrix evaluated at the initial solution $\boldsymbol{u}_0$. Thus, the preconditioning matrix $\boldsymbol{M}_k$ remains the same for all iterations k. We emphasize that other preconditioners, such as Gauss–Seidel and multigrid, can be similarly applied to the ARDM.

Figures 12 and 13 show the convergence for the nonlinear elliptic problem in

Table 3
Convergence of the Newton-GMRES method for the nonlinear elliptic problem with the Jacobi and ILU(0) preconditioners.

Newton iteration	Residual norm	Jacobi	ILU(0)
1	3.90×10^{0}	138	40
2	9.78×10^{-1}	119	33
3	1.40×10^{-1}	111	30
4	4.18×10^{-3}	109	29
5	3.60×10^{-6}	110	29
6	2.13×10^{-12}	139	28
Total GMRES iterations		726	189

Table 4
Convergence of the Newton-GMRES method for the Burgers problem with the Jacobi and ILU(0) preconditioners.

Newton iteration	Residual norm	Jacobi	ILU(0)
1	3.80×10^{-2}	232	57
2	8.82×10^{-3}	240	47
3	1.56×10^{-3}	242	47
4	1.25×10^{-4}	256	51
5	1.61×10^{-6}	261	53
6	4.06×10^{-10}	272	55
Total GMRES iterations		1503	310

section 4.3 and the Burgers problem in section 4.4, respectively. Tables 3 and 4 show the convergence of the preconditioned Newton-GMRES method for the nonlinear elliptic and Burgers problems, respectively. The stability parameter α is modified to account for the preconditioning effect.

Several remarks follow from these results. First, the use of the ILU(0) preconditioner reduces the number of residual evaluations due to the improved spectra of the resulting system, as discussed in section 3.3. Second, the Jacobi preconditioner is not effective since it has very little impact on the performance of the various methods. This makes sense because taking the diagonal of the initial Jacobian matrix has very little influence on the spectra of the underlying system during the iteration. Third, our method converges faster than Nesterov's method and its gradient restarting version for the Burgers problem when the ILU(0) preconditioner is used. For the other three cases, our method and gradient restarting Nesterov method behave similarly. Fourth, our method outperforms the Newton-GMRES method since the latter incurs more residual evaluations and extra computational cost due to the orthogonalization of the Krylov vectors. Finally, we emphasize that computing, storing, and factoring the Jacobian matrix in Newton-Krylov methods demand high computational cost and memory requirements. Therefore, our method can be more scalable and efficient than the Newton-GMRES method for solving very large scale systems on parallel computers.

5. Conclusions and open research areas.

We have presented accelerated residual methods for the iterative solution of systems of equations. The methods have a number of attractive computational and theoretical features. They are more stable, robust, and faster than Nesterov's method since their stability region is larger than that of Nesterov's method. They are computationally competitive with Newton–Krylov methods for solving nonlinear systems since they do not require the construction and storage of the Krylov vectors. Owing to low memory storage and computational cost per iteration, they are highly scalable and efficient for solving very large systems on parallel computers. We end the paper by discussing open research areas.

5.1. Generalized accelerated residual methods. First, we note that the schemes discussed in this paper belong to the following family of methods:

$$\boldsymbol{v}_k = \sum_{i=1}^{n} \left(a_{ik}\boldsymbol{u}_{k+1-i} + b_{ik}\boldsymbol{f}(\boldsymbol{u}_{k+1-i}) \right), \tag{36a}$$

$$\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{f}(\boldsymbol{v}_k). \tag{36b}$$

In particular, we have $n = 1$, $a_{1k} = 1$, and $b_{1k} = 0$ for the forward Euler method. We also have $n = 2$, $a_{1k} = (1+\beta_k)$, $a_{2k} = -\beta_k$, and $b_{1k} = b_{2k} = 0$ for Nesterov's method. And we have $n = 2$, $a_{1k} = (1+\beta_k)$, $a_{2k} = -\beta_k$, $b_{1k} = \alpha_k(1+\beta_k)$, and $b_{2k} = 0$ for the accelerated residual method. The extrapolation step (36a) can be interpreted as an attempt to accelerate the iterative process using the two subspaces $\{\boldsymbol{u}_{k+1-i}\}_{i=1}^{n}$ and $\{\boldsymbol{f}(\boldsymbol{u}_{k+1-i})\}_{i=1}^{n}$. The n-step extrapolation—as a generalization of the two-step extrapolation in the case of Nesterov's method and the accelerated residual method—may accelerate the convergence even further.

Obviously, the parameters a_{ik} and b_{ik} play a crucial role in the convergence and stability of the resulting method. As discussed in section 3.3, the stability region of the resulting scheme depends on these parameters. It allows us to find an optimal set of parameters for a particular problem at hand. For instance, for SPD systems, we should find these parameters to maximize the stability region on the real axis. On the other hand, for convection-dominated systems, we should find the parameters to maximize the stability region on the imaginary axis. For nonlinear systems in which their spectral properties may change from one iteration to the next, the optimal parameters also vary from one iteration to the next. Therefore, it makes sense to adapt these parameters to the spectral properties.

5.2. Preconditioned accelerated residual methods. Let us consider the following first-order ODE system:

$$\boldsymbol{M}(t)\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0, \tag{37}$$

where $\boldsymbol{M}(t)$ is the preconditioning matrix. The forward Euler method for the above ODE system reads as

$$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k - \alpha_k \boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{u}_k). \tag{38}$$

Note that if we choose $\boldsymbol{M}_k^{-1} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}(\boldsymbol{u}_k)$, then the above iteration is nothing but the damped Newton method for solving the system (1). Next, we recall from section 3.6 that the preconditioned accelerated residual method

$$\boldsymbol{v}_k = \boldsymbol{u}_k + \beta_k(\boldsymbol{u}_k - \boldsymbol{u}_{k-1}) - \alpha_k(1+\beta_k)\boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{u}_k), \tag{39a}$$

$$\boldsymbol{u}_{k+1} = \boldsymbol{v}_k - \alpha_k \boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{v}_k) \tag{39b}$$

is derived as an FDA of the second-order ODE system

$$\boldsymbol{M}(t)\ddot{\boldsymbol{u}} + \frac{\gamma}{t}\boldsymbol{M}(t)\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0. \tag{40}$$

Through both the theoretical analysis and the numerical experiments presented in this paper, we see that the accelerated residual method converges significantly faster than the forward Euler method. Therefore, the preconditioned accelerated residual method can be combined with a Krylov method such as CG or GMRES to solve $\boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{u}_k)$ in (39a) and $\boldsymbol{M}_k^{-1}\boldsymbol{f}(\boldsymbol{v}_k)$ in (39b). The resulting method can converge much faster than the Newton–Krylov method.

We can also extend the above ideas by devising accelerated residual methods as FDAs of the following ODE system:

$$\boldsymbol{M}(t)\ddot{\boldsymbol{u}} + \frac{\gamma}{t}\boldsymbol{C}(t)\dot{\boldsymbol{u}} + \boldsymbol{f}(\boldsymbol{u}) = 0, \tag{41}$$

where $\boldsymbol{M}(t)$ is the preconditioning matrix for the acceleration effect and $\boldsymbol{C}(t)$ is the preconditioning matrix for the damping effect. When being applied to linear systems of equations, the resulting method would generate iterates outside the Krylov subspace, thereby bringing an opportunity to go beyond Krylov methods such as CG and GMRES. When being applied to nonlinear systems, the resulting method would be very interesting to analyze and determine effective preconditioners, thereby allowing us to move beyond traditional preconditioned Newton–Krylov methods.

Acknowledgment. We thank Mohamed Aziz Bhouri for his useful comments and suggestions during the review process.

REFERENCES

[1] H. Attouch and J. Peypouquet, *The rate of convergence of Nesterov's accelerated forward-backward method is actually faster than* $1/k^2$, SIAM J. Optim., 26 (2016), pp. 1824–1834, https://doi.org/10.1137/15M1046095.

[2] A. Brandt, *Multi-level adaptive solutions to boundary-value problems*, Math. Comp., 31 (1977), pp. 333–390.

[3] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, 2nd ed., SIAM, Philadelphia, 2000, https://doi.org/10.1137/1.9780898719505.

[4] P. N. Brown and Y. Saad, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 450–481, https://doi.org/10.1137/0911026.

[5] C. G. Broyden, *The convergence of a class of double-rank minimization algorithms* 1. *General considerations*, IMA J. Appl. Math., 6 (1970), pp. 76–90.

[6] T. F. Chan and K. R. Jackson, *Nonlinearly preconditioned Krylov subspace methods for discrete Newton algorithms*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 533–542, https://doi.org/10.1137/0905039.

[7] R. Fletcher, *A new approach to variable metric algorithms*, Comput. J., 13 (1970), pp. 317–322.

[8] R. W. Freund and N. M. Nachtigal, *QMR: A quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.

[9] D. Goldfarb, *A family of variable-metric methods derived by variational means*, Math. Comp., 24 (1970), pp. 23–26.

[10] W. Hackbusch, *Multi-grid Methods and Applications*, Vol. 4, Springer-Verlag, Berlin, Heidelberg, 2013.

[11] M. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp. 409–436.

[12] D. A. Knoll and David E. Keyes, *Jacobian-free Newton-Krylov methods: A survey of approaches and applications*, J. Comput. Phys., 193 (2004), pp. 357–397.

[13] D. J. Mavriplis, *Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes*, J. Comput. Phys., 145 (1998), pp. 141–165.

[14] A. S. Nemirovsky and D. B. Yudin, *Problem complexity and method efficiency in optimization*, SIAM Rev., 27 (1985), pp. 264–265, https://doi.org/10.1137/1027074.

[15] Y. Nesterov, *A method of solving a convex programming problem with convergence rate* $o(1/k^2)$, Soviet Math. Dokl., 27 (1983), pp. 372–376.

[16] Y. Nesterov, *Introductory Lectures on Convex Optimization*, Appl. Optim. 87, Kluwer Academic Publishers, Boston, MA, 2004.

[17] Y. Nesterov, *Smooth minimization of non-smooth functions*, Math. Program., 103 (2005), pp. 127–152.

[18] Y. Nesterov, *Gradient methods for minimizing composite functions*, Math. Program., 140 (2013), pp. 125–161.

[19] B. O'Donoghue and E. Candès, *Adaptive restart for accelerated gradient schemes*, Found. Comput. Math., 15 (2013), pp. 715–732.

[20] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629, https://doi.org/10.1137/0712047.

[21] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869, https://doi.org/10.1137/0907058.

[22] D. F. Shanno, *Conditioning of quasi-Newton methods for function minimization*, Math. Comp., 24 (1970), pp. 647–656.

[23] P. Sonneveld, *CGS, a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52, https://doi.org/10.1137/0910004.

[24] W. Su, S. Boyd, and E. J. Candès, *A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights*, J. Mach. Learn. Res., 17 (2016), 153.

[25] P. Tseng, *Approximation accuracy, gradient methods, and error bound for structured convex optimization*, Math. Program., 125 (2010), pp. 263–295.

[26] H. A. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644, https://doi.org/10.1137/0913035.

[27] D. Young, *Iterative methods for solving partial difference equations of elliptic type*, Trans. Amer. Math. Soc., 76 (1954), pp. 92–111.