Non-stationary Anderson acceleration with optimized damping

Chen, Kewang; Vuik, Cornelis

DOI
10.1016/j.cam.2024.116077

Publication date
2024

Document Version
Final published version

Published in
Journal of Computational and Applied Mathematics

Journal of Computational and Applied Mathematics 451 (2024) 116077

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Non-stationary Anderson acceleration with optimized damping ☆

Kewang Chen [a,b,*], Cornelis Vuik [b]

[a] College of Mathematics and Statistics, Nanjing University of Information Science and Technology, Nanjing, 210044, China
[b] Delft Institute of Applied Mathematics, Delft University of Technology, Delft, 2628XE, The Netherlands

ARTICLE INFO

MSC:
65H10
65F10

Keywords:
Anderson acceleration
Fixed-point iteration
Optimal damping

ABSTRACT

Anderson acceleration (AA) has a long history of use and a strong recent interest due to its potential ability to dramatically improve the linear convergence of the fixed-point iteration. Most authors are simply using and analyzing the stationary version of Anderson acceleration (sAA) with a constant damping factor or without damping. Little attention has been paid to nonstationary algorithms. However, damping can be useful and is sometimes crucial for simulations in which the underlying fixed-point operator is not globally contractive. The role of this damping factor has not been fully understood. In the present work, we consider the non-stationary Anderson acceleration algorithm with optimized damping (AAoptD) in each iteration to further speed up linear and nonlinear iterations by applying one extra inexpensive optimization. We analyze the convergence rate this procedure and develop an efficient and inexpensive implementation scheme. We show by extensive numerical experiments that the proposed non-stationary Anderson acceleration with optimized damping procedure often converges much faster than stationary AA with constant damping, adaptive damping or without damping, especially in the cases larger window sizes are needed. We also observe that simple strategies like using constant damping factors and adaptive damping factors, sometimes, work very well for some problems while sometimes they are even worse than AA without damping. Our proposed method is usually more robust than AA with constant damping and adaptive damping. Moreover, we also observed from our numerical results that damping can be good, but choosing the wrong damping factors may slow down the convergence rate. Theoretical analysis of the effects of damping factors are needed and important.

1. Introduction

In this part, we first give a literature review on Anderson Acceleration method. Then we discuss our main motivations and the structure for the present paper. To begin with, let us consider the nonlinear acceleration for the following general fixed-point problem

$$x = g(x),\ g : R^n \to R^n$$

or its related nonlinear equations problem

$$f(x) = g(x) - x = 0.$$

☆ **Funding:** This work was partially supported by the National Natural Science Foundation of China [grant number 12001287]; the Startup Foundation for Introducing Talent of Nanjing University of Information Science and Technology, China [grant number 2019r106].

* Corresponding author at: College of Mathematics and Statistics, Nanjing University of Information Science and Technology, Nanjing, 210044, China.
E-mail addresses: kwchen@nuist.edu.cn (K. Chen), c.vuik@tudelft.nl (C. Vuik).
URL: https://homepage.tudelft.nl/d2b4e/ (C. Vuik).

https://doi.org/10.1016/j.cam.2024.116077
Received 9 February 2022; Received in revised form 2 July 2023
Available online 18 June 2024

The associated basical fixed-point iteration is given in Algorithm 1.

Algorithm 1 Picard iteration

Given: x_0.
for $k = 0, 1, 2, \cdots$ **do**
  Set $x_{k+1} = g(x_k)$.
end for

Algorithm 2 Anderson acceleration: $AA(m)$

Given: x_0 and $m \geq 1$.
Set: $x_1 = g(x_0)$.
for $k = 0, 1, 2, \cdots$ **do**
  Set: $m_k = \min\{m, k\}$.
  Set: $F_k = (f_{k-m_k}, \cdots, f_k)$, where $f_i = g(x_i) - x_i$.
  Determine: $\alpha^{(k)} = \left(\alpha_0^{(k)}, \cdots, \alpha_{m_k}^{(k)}\right)^T$ that solves

$$\min_{\alpha=(\alpha_0,\cdots,\alpha_{m_k})^T} \|F_k\alpha\|_2 \quad s.\,t. \sum_{i=0}^{m_k} \alpha_i = 1.$$

  Set: $x_{k+1} = (1-\beta_k)\sum_{i=0}^{m_k} \alpha_i^{(k)} x_{k-m_k+i} + \beta_k \sum_{i=0}^{m_k} \alpha_i^{(k)} g(x_{k-m_k+i})$.
end for

The main concern related to this basic fixed-point iteration is that the iterates may not converge or may converge extremely slowly (only linear convergent). Therefore, various acceleration methods are proposed to alleviate this slow convergence problem. Among these algorithms, one popular acceleration procedure is called the Anderson acceleration method [1]. For the above basic Picard iteration, the usual general form of Anderson acceleration with damping is given in Algorithm 2. In the above algorithm, f_k is the residual for the kth iteration; m is the window size which indicates how many history residuals will be used in the algorithm. The value of m is typically no larger than 3 in the early days of applications and now this value could be as large as up to 100, see [2]. It is usually a fixed number during the procedure, varying m can also make the algorithm to be non-stationary. We will come back to this point in section Section 2; $\beta_k \in (0, 1]$ is a damping factor (or a relaxation parameter) at kth iteration. We have, for a fixed window size m:

$$\beta_k = \begin{cases} 1, & \text{no damping,} \\ \beta, \text{ (a constant independent of } k) & \text{stationary AA,} \\ \beta_k, \text{ (depending on } k) & \text{non-stationary AA.} \end{cases}$$

The constrained optimization problem can also be formulated as an equivalent unconstrained least-squares problem [3,4]:

$$\min_{(\omega_1,\ldots,\omega_{m_k})^T} \left\| f_k + \sum_{i=1}^{m_k} \omega_i (f_{k-i} - f_k) \right\|_2 \tag{1}$$

One can easily recover the original problem by setting

$$\omega_0 = 1 - \sum_{i=1}^{m_k} \omega_i.$$

This formulation of the linear least-squares problem is not optimal for implementation, we will discuss this in more detail in Section 4.

Anderson acceleration method dates back to the 1960s. In 1962, Anderson [1] developed a technique for accelerating the convergence of the Picard iteration associated with a fixed-point problem which is called Extrapolation Algorithm. This technique is now called Anderson Acceleration (AA) in the applied mathematics community and Anderson Mixing in the physics and chemistry communities. This method is "essentially" (or nearly) similar to the nonlinear GMRES method or Krylov acceleration [5–8] and the direct inversion on the iterative subspace method (DIIS) [9–11]. And it is also in a broad category with methods based on quasi-Newton updating [12–16]. However, unlike Newton-like methods, AA does not require the computation or approximation of Jacobians or Jacobian-vector products which could be an advantage.

Although the Anderson acceleration method has been around for decades, convergence analysis has been reported in the literature only recently. Fang and Saad [14] had clarified a remarkable relationship of AA to quasi-Newton methods and extended it to define a broader Anderson family method. Later, Walker and Ni [17] showed that, on linear problems, AA without truncation is "essentially equivalent" in a certain sense to the GMRES method. For the linear case, Toth and Kelley [3] first proved the stationary version of AA (sAA) without damping is locally r-linearly convergent if the fixed point map is a contraction and the coefficients in the linear combination remain bounded. This work was later extended by Evans et al. [18] to AA with damping and the authors proved the new convergence rate is $\theta_k((1-\beta_{k-1})+\beta_{k-1}\kappa)$, where κ is the Lipschitz constant for the function $g(x)$ and θ_k is the ratio quantifying the convergence gain provided by AA in step k. However, it is not clear how θ_k may be evaluated or bounded in practice and how it may translate to improved asymptotic convergence behavior in general. In 2019, Pollock et al. [19] applied sAA to the Picard iteration

for solving steady incompressible Navier–Stokes equations (NSE) and proved that the acceleration improves the convergence rate of the Picard iteration. Then, De Sterck [20] extended the result to more general fixed-point iteration $x = g(x)$, given knowledge of the spectrum of $g'(x)$ at fixed-point x^* and Wang et al. [21] extended the result to study the asymptotic linear convergence speed of sAA applied to Alternating Direction Method of Multipliers (ADMM) method. Sharper local convergence results of AA remain a hot research topic in this area. More recently, Zhang et al. [22] proved a global convergent result of type-I Anderson acceleration for nonsmooth fixed-point iterations without resorting to line search or any further assumptions other than nonexpansiveness. For more related results about Anderson acceleration and its applications, we refer the interested readers to [2,23–28] and references therein.

As mentioned above, the local convergence rate $\theta_k((1-\beta_{k-1}) + \beta_{k-1}\kappa)$ at stage k is closely related to the damping factor β_{k-1}. However, questions like how to choose those damping values in each iteration [2] and how it will affect the global convergence of the algorithm have not been deeply studied. In practice, β is often chosen by experimenting with several representative constant β values. Anderson [2] suggested a conceptual procedure for adaptively choosing β_k. However, he has not had an opportunity to assess its practical utility. Evans et al. [18] developed a new strategy to adaptively choose the damping factors, where those β_k are chosen by a simple heuristic strategy based on the gain θ_k ($\beta_k = 0.9 - 1/2 * \theta_k$). The heuristic choice of damping yields $0.4 \leq \beta_k \leq 0.9$, and leads to fewer iterations to convergence than with the uniform damping factors tested on the p-Laplacian problem, where p-Laplacian is a noncontractive operators. We do not know how well this strategy works for other problems. For the linear problems, Potra and Engler [29] proposed an optimized version of Anderson acceleration, where at each step the mixing parameter is chosen so that it minimizes the residual of the current iterate. Can we generalize this idea to solve nonlinear equations? How good will this new strategy be compared with other constant damping and adaptive damping strategies? Besides, AA is often combined with globalization methods to safeguard against erratic convergence away from a fixed point by using damping. One similar idea in the optimization context for nonlinear GMRES is to use line search strategies [30]. This is an important strategy but not yet fully explored in the literature. Moreover, the early days of Anderson Mixing method (the 1980s, for electronic structure calculations) initially dictated the window size $m \leq 3$ due to the storage limitations and costly g evaluations involving large N. However, in recent years and a broad range of contexts, the window size m ranging from 20 to 100 has also been considered by many authors. For example, Walker and Ni [17] used $m = 50$ in solving the nonlinear Bratu problem. A natural question will be should we try to further steep up Anderson acceleration method or try to use a larger size of the window? No such comparison results have been reported. Motivated by the above works, in this paper, we propose, analyze and numerically study non-stationary Anderson acceleration with optimized damping to solve fixed-point problems. The goal of this paper is to explore the role of damping factors in non-stationary Anderson acceleration and comparing different strategies for choosing those damping factors.

The paper is organized as follows. Our new algorithms and analysis are in Section 2, the convergence analysis are given in Section 3, the implementation of the new algorithm is in Section 4, experimental results and discussion are in Section 5. Conclusions follow in Section 6.

2. Anderson acceleration with optimized dampings

In this section, we focus on developing the algorithm for Anderson acceleration with optimized dampings at each iteration and studying its convergence rate explicitly.

$$\begin{aligned} x_{k+1} &= (1-\beta_k)\sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i} + \beta_k\sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}) \\ &= \sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i} + \beta_k\left(\sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}) - \sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i}\right). \end{aligned} \tag{2}$$

Define the following averages given by the solution α^k to the optimization problem by

$$x_k^\alpha = \sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i}, \quad \tilde{x}_k^\alpha = \sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}). \tag{3}$$

Then (2) becomes

$$x_{k+1} = x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha). \tag{4}$$

A natural way to choose "best" β_k at this stage is that choosing β_k such that x_{k+1} gives a minimal residual. This is similar to the original idea of Anderson acceleration with window size equal to one. The idea has been used for solving linear iteration problems by Potra and Engler [29]. So we just need to solve the following unconstrained optimization problem:

$$\min_{\beta_k}\|x_{k+1} - g(x_{k+1})\|_2 = \min_{\beta_k}\|x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha) - g(x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha))\|_2. \tag{5}$$

Noting the fact that $g(x)$ is usually a nonlinear function, thus we proceed by linearization to allow for inexpensive computation of β_k. Using the approximations

$$\begin{aligned} g(x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha)) &\approx g(x_k^\alpha) + \beta_k\frac{\partial g}{\partial x}\Big|_{x_k^\alpha}(\tilde{x}_k^\alpha - x_k^\alpha) \\ &\approx g(x_k^\alpha) + \beta_k\left(g(\tilde{x}_k^\alpha) - g(x_k^\alpha)\right), \end{aligned} \tag{6}$$

we arrive at

$$\begin{aligned}
&\min_{\beta_k} \|x_{k+1} - g(x_{k+1})\|_2 \\
&= \min_{\beta_k} \|x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha) - g(x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha))\|_2 \\
&\approx \min_{\beta_k} \|x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha) - \left[g(x_k^\alpha) + \beta_k(g(\tilde{x}_k^\alpha) - g(x_k^\alpha))\right]\|_2 \\
&\approx \min_{\beta_k} \| \left(x_k^\alpha - g(x_k^\alpha)\right) - \beta_k \left[(g(\tilde{x}_k^\alpha) - g(x_k^\alpha)) - (\tilde{x}_k^\alpha - x_k^\alpha)\right] \|_2. \qquad (7)
\end{aligned}$$

Thus, we just need to calculate the projection

$$\beta_k = \frac{\left(x_k^\alpha - g(x_k^\alpha)\right) \cdot \left[\left(x_k^\alpha - g(x_k^\alpha)\right) - (\tilde{x}_k^\alpha - g(\tilde{x}_k^\alpha))\right]}{\| \left[\left(x_k^\alpha - g(x_k^\alpha)\right) - (\tilde{x}_k^\alpha - g(\tilde{x}_k^\alpha))\right] \|_2^2}. \qquad (8)$$

Set

$$r_p = \left(x_k^\alpha - g(x_k^\alpha)\right), \quad r_q = \left(\tilde{x}_k^\alpha - g(\tilde{x}_k^\alpha)\right),$$

we have

$$\beta_k = \frac{(r_p - r_q)^T r_p}{\|r_p - r_q\|_2^2}. \qquad (9)$$

We will discuss how much work is needed to calculate this β_k in Section 4. Since the damping factor β_k is restricted in the interval $(0, 1]$, if the local "best" β_k is not in this interval (i.e. β_k is negative or β_k is larger than one), to avoid a large value of the estimated residual, we can use a constant damping factor for this iteration. Without any pre-experiment or further calculations, a natural guess for this constant damping factor is $\beta_k = 1/2$. Besides, $\beta_k = 1$ is also a safe option, which means we do not use damping for this iteration. Finally, our analysis leads to the following non-stationary Anderson acceleration algorithm with optimized damping: *AAoptD*(m).

Algorithm 3 Anderson acceleration with optimized dampings: *AAoptD*(m)

Given: x_0 and $m \geq 1$.
Set: $x_1 = g(x_0)$.
for $k = 0, 1, 2, \cdots$ **do**
  Set: $m_k = \min\{m, k\}$.
  Set: $F_k = (f_{k-m_k}, \cdots, f_k)$, where $f_i = g(x_i) - x_i$.
  Determine: $\alpha^{(k)} = \left(\alpha_0^{(k}, \cdots, \alpha_{m_k}^{(k)}\right)^T$ that solves

$$\min_{\alpha=(\alpha_0,\cdots,\alpha_{m_k})^T} \|F_k\alpha\|_2 \quad s.\,t. \sum_{i=0}^{m_k} \alpha_i = 1.$$

  Set: $x_k^\alpha = \sum_{i=0}^{m_k} \alpha_i^{(k)} x_{k-m_k+i}, \quad \tilde{x}_k^\alpha = \sum_{i=0}^{m_k} \alpha_i^{(k)} g(x_{k-m_k+i})$.
  Set: $r_p = \left(x_k^\alpha - g(x_k^\alpha)\right), \quad r_q = \left(\tilde{x}_k^\alpha - g(\tilde{x}_k^\alpha)\right)$.
  Set: $\beta_k = \frac{(r_p - r_q)^T r_p}{\|r_p - r_q\|_2^2}$. (If $\beta_k \notin (0, 1]$, then set $\beta_k = 1/2$).
  Set: $x_{k+1} = x_k^\alpha + \beta_k(\tilde{x}_k^\alpha - x_k^\alpha)$.
end for

Remark 2.1. As mentioned in Section 1, changing the window size m at each iteration can also make a stationary Anderson acceleration to be non-stationary. Comparing with the stationary Anderson acceleration with fixed window $sAA(m)$, our proposed nonstationary procedure ($AAoptD(m)$) of choosing optimal β_k is somewhat related to packaging $sAA(m)$ and $sAA(1)$ in each iteration in a cheap way. Here "packaging $sAA(m)$ and $sAA(1)$ in each iteration" means that we alternate the window size every other iteration in Anderson acceleration algorithm by using a window size equals to m and then using a window size equals to 1. Since our proposed method AAoptD(m) uses a window size equal m first, then we do an extra optimization on choosing the local optimal damping factor. The idea of doing this extra optimization is somewhat like applying stationary AA with a window size equal to 1.

Remark 2.2. Here this optimized damping step is a "local optimal" strategy at kth iteration. It usually will speed up the convergence rate compared with the one with constant damping, but not always. Because in $(k+1)$th iteration, it uses a combination of all previous m history information. Usually, the more optimized damping β_k values are in the interval $(0, 1]$, the faster the convergence speed. Moreover, when β_k is close to zero, the system may be over-damped, which, sometimes, will also slow down the convergence speed. See more discussion in our numerical results in Section 5.

3. Convergence analysis

In this section, we first prove that the non-stationary Anderson acceleration with optimized damping converges and its residual converges q-linearly to zero in the case of linear problems. For nonlinear problems, we prove that it is locally r-linearly convergent if the fixed point map is a contraction and the coefficients in the linear combination remain bounded. The main idea of this proof is adopted from Toth and Kelley's paper [3] with necessary modifications. We extend their results to our non-stationary Anderson acceleration with optimized damping.

Recall that a sequence $\{u_k\}$ converges q-linearly with q-factor $c \in [0, 1)$ to u^* if

$$\|u_{k+1} - u^*\| \leq c\|u_k - u^*\|$$

for all $k \geq 0$. Similarly, a sequence converges r-linearly means that there is $\hat{c} \in (0, 1)$ and $M > 0$ such that

$$\|u_k - u^*\| \leq M\hat{c}^k\|u_0 - u^*\|.$$

3.1. Linear problems and local q-linear convergence

Let A be a linear operator with $\|A\| = c < 1$, we consider the following fixed point problem

$$x = g(x) = Ax + b.$$

The residual in this case is

$$f(x) = g(x) - x = b - (I - A)x. \tag{10}$$

Theorem 3.1. *Let $m_k = \min(m, k)$ and assume that $\|A\| = c < 1$, then the non-stationary Anderson acceleration algorithm with optimized damping*

$$x_{k+1} = \min_{\beta_k} \|f(x_{k+1}^{\beta_k})\|_2 = \min_{\beta_k} \|g(x_{k+1}^{\beta_k}) - x_{k+1}^{\beta_k}\|_2, \tag{11}$$

where

$$x_{k+1}^{\beta_k} = (1 - \beta_k)x_k^{\alpha} + \beta_k\tilde{x}_k^{\alpha} \quad (with\ \ damping\ \ factor), \tag{12}$$

$$\tilde{x}_k^{\alpha} = \sum_{j=0}^{m_k} \alpha_j^k g(x_{k-m_k+j}) \qquad (by\ \ solving\ \ a\ \ optimized\ \ problem) \tag{13}$$

converges to $x^* = (I - A)^{-1}b$ and the residuals converge q-linearly to zero with a q-factor equal c.

Proof. Using the fact that $\sum_{j=0}^{m_k} \alpha_j^k = 1$ and if $\alpha^k = \left(\alpha_0^k, \ldots, \alpha_{m_k}^k\right)^T$ is the solution of the following least squares problem at iteration k, then by definition, we have

$$\left\|\sum_{j=0}^{m_k} \alpha_j^k f(x_{k-m_k+j})\right\| \leq \|f(x_k)\|, \tag{14}$$

since $\alpha^k = (0, \ldots, 0, 1)^T$ is just one of all possible choices.

Another important fact is that our way of choosing "best" β_k gives a minimal residual, thus from (11) and (12), we obtain

$$\left\|f(x_{k+1})\right\| \leq \|f(\tilde{x}_k^{\alpha})\|, \tag{15}$$

because $\beta_k = 1$ is just one special case. Therefore, the new residual

$$\begin{aligned}
\left\|f(x_{k+1})\right\| &\leq \left\|f(\tilde{x}_k^{\alpha})\right\| \quad (by\ (15)) \\
&= \left\|b - (I - A)\tilde{x}_k^{\alpha}\right\| \\
&= \left\|\sum_{j=0}^{m_k} \alpha_j^k [b - (I - A)(b + Ax_{k-m_k+j})]\right\| \quad (use \sum_{i=0}^{n} \alpha_i^k = 1) \\
&= \left\|\sum_{j=0}^{m_k} \alpha_j^k A[b - (I - A)x_{k-m_k+j}]\right\| \\
&= \left\|A\sum_{j=0}^{m_k} \alpha_j^k f(x_{k-m_k+j})\right\| \quad (by\ \|A\| = c < 1) \\
&\leq c\left\|f(x_k)\right\|. \quad (by\ (14))
\end{aligned} \tag{16}$$

Thus, we have

$$\left\|f(x_{k+1})\right\| \leq c\left\|f(x_k)\right\| \quad (c < 1),$$

this proves the residuals converge q-linearly to zero with a q-factor equals c. □

3.2. Nonlinear problems and local r-linear convergence

In this section, we prove the local convergence result for more general nonlinear iterations with suitable assumption. Again, we set

$$f(x) = g(x) - x.$$

As is standard we will let F' denotes the Jacobian of F and $e = x - x^*$. The assumptions we make on the nonlinearity $g(x)$ and the solution x^* follow the standard assumptions for local convergence of Newton's method [3].

Assumption 1. We assume that

- $g : R^n \to R^n$ has a fixed point $x^* \in R^n$ such that $f(x^*) = g(x^*) - x^* = 0$.
- g is uniformly Lipschitz continuously differentiable in the ball $B(\hat{\rho}) = \{x | \|x - x^*\|_2 \le \hat{\rho}\}$.
- There exists $c \in (0, 1)$ such that $\|g(y) - g(x)\|_2 \le c\|y - x\|_2$ for all $x, y \in R^n$.
- There is a N_α such that for all $k \ge 0$, $\sum_{j=0}^{m_k} |\alpha_j| \le N_\alpha$ hold.

The second and the third assumptions indicate that $\|g'(x)\| \le c < 1$ for all $x \in B(\hat{\rho})$, and hence $f'(x)$ is nonsingular. For the last assumption, although we have not seen large coefficient values in our numerical experiments, we are not able to prove they remain bounded. Thus we keep it here. We also denote $g^* = g'(x^*)$ and let γ be the Lipschitz constant of f' (the Jacobian of $f(x)$) in $B(\rho)$. We will need a special case of the results (Lemma 4.3.1) from [31].

Lemma 3.2. *For $\rho \le \hat{\rho}$ sufficiently small and all $x \in B(\rho)$, we have*

$$\|f(x) - f'(x^*)e\| \le \frac{\gamma}{2}\|e\|^2 \tag{17}$$

and

$$\|e\|(1 - c) \le \|f(x)\| \le (1 + c)\|e\|. \tag{18}$$

Theorem 3.3. *Assume that the above assumptions hold and let $c < \hat{c} < 1$. Then if x_0 is sufficiently close to x^*, then the proposed non-stationary Anderson acceleration algorithm with optimized damping*

$$x_{k+1} = \min_{\beta_k} \|f(x_{k+1}^{\beta_k})\|_2 = \min_{\beta_k} \|g(x_{k+1}^{\beta_k}) - x_{k+1}^{\beta_k}\|_2, \tag{19}$$

where

$$x_{k+1}^{\beta_k} = (1 - \beta_k)x_k^{\alpha} + \beta_k \tilde{x}_k^{\alpha} \quad (with\ \ damping\ \ factor), \tag{20}$$

$$\tilde{x}_k^{\alpha} = \sum_{j=0}^{m_k} \alpha_j^k g(x_{k-m_k+j}) \qquad (by\ \ solving\ \ a\ \ optimized\ \ problem) \tag{21}$$

converges to x^* r-linearly. In fact,

$$\|f(x_k)\| \le \hat{c}^{k+1}\|f(x_0)\| \tag{22}$$

and

$$\|e_k\| \le \frac{1 + c}{1 - c}\hat{c}^k\|e_0\| \tag{23}$$

Proof. We proceed this proof by induction. To begin with, when $k = 0$, it is easy to check that

$$\|f(x_0)\| \le \hat{c}^0\|f(x_0)\| = \|f(x_0)\|. \tag{24}$$

So we assume that when $k = K$ $(K \ge 1)$ that

$$\|f(x_k)\| \le \hat{c}^k\|f(x_0)\|. \tag{25}$$

We want to show that for $k = K + 1$, we have

$$\|f(x_{K+1})\| \le \hat{c}^{K+1}\|f(x_0)\|. \tag{26}$$

Then, by induction, we can prove Theorem 3.3. The following part of this section is devoted to proof (26).

To begin with, using the fact that our way of choosing "best" β_k gives a minimal residual, thus from (19) and (20), we have

$$\|f(x_{K+1})\| \le \|f(\tilde{x}_K^{\alpha})\|. \tag{27}$$

thus, we just need to show

$$\|f(\tilde{x}_K^{\alpha})\| \le \hat{c}^{K+1}\|f(x_0)\|. \tag{28}$$

According to Toth and Kelley [3], let $x_0 \in B(\rho)$ and assume that $\rho \le \hat{\rho}$, where $\hat{\rho}$ is from the statement of Lemma 3.2. Reduce ρ if needed so that

$$\rho \le \frac{2(1-c)}{\gamma}$$

and

$$\frac{\frac{c}{\hat{c}} + \left(\frac{N_\alpha \gamma \rho}{2(1-c)}\right)\hat{c}^{-m-1}}{\left(1 - \frac{\gamma\rho}{2(1-c)}\right)} \le 1, \tag{29}$$

which will be used later in the proof. Now reduce $\|e_0\|$ further, which means we need a better initial guess, so that

$$\left(\frac{N_\alpha(c+\gamma\rho/2)}{1-c}\right)\hat{c}^{-m}\|f(x_0)\| \le \left(\frac{N_\alpha(1+c)(c+\gamma\rho/2)}{1-c}\right)\hat{c}^{-m}\|e_0\| \le \rho. \tag{30}$$

From Eqs. (25) and (30), we have that $\|e_K\| \le \rho$. Hence, we have

$$f(x_K) = f'(x^*)e_K + \Delta_K, \tag{31}$$

where (by Lemma 3.2),

$$\|\Delta_K\| = \|f(x_K) - f'(x^*)e_K\| \le \frac{\gamma}{2}\|e_K\|. \tag{32}$$

Thus, from (31) and $f(x) = g(x) - x$, we have

$$g(x_K) = x^* + g'(x^*)e_K + \Delta_K = x^* + g^* e_K + \Delta_K. \tag{33}$$

From (21), (33) and use the fact that $\sum_{i=0}^{m_k} \alpha_i^K = 1$, we get

$$\begin{aligned} \tilde{x}_K^\alpha &= \sum_{i=0}^{m_k} \alpha_i^K g(x_{K-m_k+j}) \\ &= x^* + \sum_{i=0}^{m_k} \alpha_i^K (g^* e_{K-m_k+i} + \Delta_{K-m_k+i}) \\ &= x^* + \sum_{i=0}^{m_k} (\alpha_i^K g^* e_{K-m_k+i}) + \sum_{i=0}^{m_k} (\alpha_i^K \Delta_{K-m_k+i}) \end{aligned} \tag{34}$$

Denote the last term in (34) as

$$\bar{\Delta}_K = \sum_{i=0}^{m_k} (\alpha_i^K \Delta_{K-m_k+i}), \tag{35}$$

then we will estimate $\bar{\Delta}_K$. From (32) and (35), we get

$$\|\bar{\Delta}_K\| \le \sum_{i=0}^{m_k} \left|\alpha_i^K\right| \gamma \|e_{K-m_k+i}\|^2/2. \tag{36}$$

Using the induction hypothesis and Lemma 3.2, we have

$$\begin{aligned} \|e_{K-m_k+i}\|^2 &\le \|e_{K-m_k+i}\| \left(\frac{1}{1-c}\right)\|f(x_{K-m_k+i})\| \\ &\le \left(\frac{\rho}{1-c}\right)\|f(x_{K-m_k+i})\| \\ &\le \left(\frac{\rho}{1-c}\right)\hat{c}^{K-m_k+i}\|f(x_0)\| \\ &\le \left(\frac{\rho}{1-c}\right)\hat{c}^{K-m}\|f(x_0)\|. \end{aligned} \tag{37}$$

Thus, use the last assumption that $\sum_{i=0}^{m_k} |\alpha_i^K| \le N_\alpha$ and the fact $\|g^*\| = \|g'(x^*)\| \le c < 1$, we have

$$\left\|\sum_{i=0}^{m_k} \alpha_i^K g^* e_{k-m_k+i}\right\| \le \left(\frac{N_\alpha c}{1-c}\right)\hat{c}^{-m}\|f(x_0)\| \tag{38}$$

and

$$\|\bar{\Delta}_K\| \le \left(\frac{N_\alpha \gamma \rho}{2(1-c)}\right)\hat{c}^{K-m}\|f(x_0)\| \le \left(\frac{N_\alpha \gamma \rho}{2(1-c)}\right)\hat{c}^{-m}\|f(x_0)\|. \tag{39}$$

Therefore, from (34), we obtain that

$$e_K^\alpha = \tilde{x}_K^\alpha - x^* = \sum_{i=0}^{m_k} (\alpha_i^K g^* e_{K-m_k+i}) + \bar{\Delta}_K. \tag{40}$$

Thus, we get the following estimation,

$$\begin{aligned}\|e_K^\alpha\| = \|\tilde{x}_K^\alpha - x^*\| &= \left\|\sum_{i=0}^{m_k}(\alpha_i^K g^* e_{K-m_k+i}) + \bar{\Delta}_K\right\| \\ &\le \left\|\sum_{i=0}^{n}(\alpha_i^K g^* e_{K-m_k+i})\right\| + \|\bar{\Delta}_K\| \\ &\le \left(\frac{N_\alpha c}{1-c}\right)\hat{c}^{-m}\|f(x_0)\| + \left(\frac{N_\alpha \gamma\rho}{2(1-c)}\right)\hat{c}^{-m}\|f(x_0)\| \\ &\le \left(\frac{N_\alpha(c+\gamma\rho/2)}{1-c}\right)\hat{c}^{-m}\|f(x_0)\| \\ &\le \rho.\end{aligned} \tag{41}$$

Since $\|e_K^\alpha\| \le \rho \le \hat{\rho}$, we can use Lemma 3.2 again with $k = K+1$ to obtain

$$f(\tilde{x}_K^\alpha) = f'(x^*)e_K^\alpha + \Delta_K^\alpha = (g^* - I)e_K^\alpha + \Delta_K^\alpha, \tag{42}$$

where, by Lemma 3.2

$$\|\Delta_K^\alpha\| = \|f(\tilde{x}_K^\alpha) - f'(x^*)e_K^\alpha\| \le \frac{\gamma}{2}\|e_K^\alpha\|^2.$$

Use the fact that $\|e_K^\alpha\| \le \rho \le \hat{\rho}$ and (18), we have

$$\|\Delta_K^\alpha\| \le \frac{\gamma}{2}\|e_K^\alpha\|^2 \le \frac{\gamma}{2}\|e_K^\alpha\|\left(\frac{1}{1-c}\right)\|f(\tilde{x}_K^\alpha)\| \le \left(\frac{\gamma\rho}{2(1-c)}\right)\|f(\tilde{x}_K^\alpha)\|. \tag{43}$$

Then, from (40) and (42), we get

$$\begin{aligned}f(\tilde{x}_K^\alpha) &= (g^* - I)e_K^\alpha + \Delta_K^\alpha \\ &= (g^* - I)\left(\sum_{i=0}^{m_k}(\alpha_i^K g^* e_{K-m_k+i}) + \bar{\Delta}_K\right) + \Delta_K^\alpha.\end{aligned} \tag{44}$$

Since g^* and $g^* - I$ commute, we obtain

$$f(\tilde{x}_K^\alpha) = g^*\sum_{i=0}^{m_k}\alpha_i^K(g^* - I)e_{K-m_k+i} + (g^* - I)\bar{\Delta}_K + \Delta_K^\alpha. \tag{45}$$

Using the fact that

$$f(x_{K-m_k+i}) = (g^* - I)e_{K-m_k+i} + \Delta_{K-m_k+i},$$

that is

$$(g^* - I)e_{K-m_k+i} = f(x_{K-m_k+i}) - \Delta_{K-m_k+i},$$

we have

$$\begin{aligned}F(\tilde{x}_K^\alpha) =& g^*\sum_{i=0}^{m_k}\alpha_i^K(g^* - I)e_{K-m_k+i} + (g^* - I)\bar{\Delta}_K + \Delta_K^\alpha \\ =& g^*\sum_{i=0}^{m_k}\left(\alpha_i^K f(x_{K-m_k+i}) - \alpha_i^K\Delta_{K-m_k+i}\right) \\ &+ (g^* - I)\bar{\Delta}_K + \Delta_K^\alpha.\end{aligned} \tag{46}$$

Then, by using (35), we arrive at

$$f(\tilde{x}_K^\alpha) = g^*\sum_{i=0}^{m_k}\alpha_i^K f(x_{K-m_k+i}) - \bar{\Delta}_K + \Delta_K^\alpha. \tag{47}$$

Now, use (43), we have

$$\begin{aligned}\|f(\tilde{x}_K^\alpha)\|\left(1 - \frac{\gamma\rho}{2(1-c)}\right) &= \|f(\tilde{x}_K^\alpha)\| - \frac{\gamma\rho}{2(1-c)}\|f(\tilde{x}_K^\alpha)\| \\ &\le \|f(\tilde{x}_K^\alpha)\| - \|\bar{\Delta}_K\|.\end{aligned} \tag{48}$$

Then, use relation (47), we obtain

$$\begin{aligned}\|f(\tilde{x}_K^\alpha)\|\left(1 - \frac{\gamma\rho}{2(1-c)}\right) &\le \|f(\tilde{x}_K^\alpha)\| - \|\bar{\Delta}_K\| \\ &\le c\left\|\sum_{i=0}^{m_k}\alpha_i^K f(x_{K-m_k+i})\right\| + \|\bar{\Delta}_K\|.\end{aligned} \tag{49}$$

Using the fact that

$$\left\| \sum_{i=0}^{m_k} \alpha_i^K f(x_{K-m_k+i}) \right\| \leq \|f(x_K)\|,$$

we get

$$\begin{aligned} \|f(\tilde{x}_K^\alpha)\| \left(1 - \frac{\gamma\rho}{2(1-c)}\right) &\leq c \left\| \sum_{i=0}^{n} \alpha_i^K f(x_{K-m_k+i}) \right\| + \|\bar{\Delta}_K\| \\ &\leq c\|f(x_K)\| + \|\bar{\Delta}_K\|. \end{aligned} \tag{50}$$

Finally, use (28) and (39), we get

$$\begin{aligned} \|f(\tilde{x}_K^\alpha)\| \left(1 - \frac{\gamma\rho}{2(1-c)}\right) &\leq c\|f(x_K)\| + \|\bar{\Delta}_K\| \\ &\leq \left(\frac{c}{\hat{c}} + \left(\frac{N_\alpha \gamma\rho}{2(1-c)}\right)\hat{c}^{-m-1}\right)\hat{c}^{K+1}\|f(x_0)\|. \end{aligned} \tag{51}$$

Therefore, use (51) and the relation (29), we have

$$\begin{aligned} \|f(\tilde{x}_K^\alpha)\| &\leq \frac{\left(\frac{c}{\hat{c}} + \left(\frac{N_\alpha\gamma\rho}{2(1-c)}\right)\hat{c}^{-m-1}\right)}{\left(1 - \frac{\gamma\rho}{2(1-c)}\right)}\hat{c}^{K+1}\|f(x_0)\| \\ &\leq \hat{c}^{K+1}\|f(x_0)\|. \end{aligned} \tag{52}$$

Using (27), we have

$$\|f(x_{K+1})\| \leq \|f(\tilde{x}_K^\alpha)\| \leq \hat{c}^{K+1}\|f(x_0)\|. \tag{53}$$

This completes the proof. □

3.3. Residual bounds estimation

Lastly, we summarize the residual bounds estimation in Theorem 3.4. This theorem can be easily proved by following the steps in [18].

Theorem 3.4 ([18]). *Assume that $g : R^n \to R^n$ is uniformly Lipschitz continuously differentiable and there exists $\kappa \in (0,1)$ such that $\|g(y) - g(x)\|_2 \leq \kappa\|y - x\|_2$ for all $x, y \in R^n$. Suppose also that $\exists M$ and $\epsilon > 0$ such that for all $k > m$, $\sum_{i=0}^{m-1} |\alpha_i| < M$ and $|\alpha_m| \geq \epsilon$. Then*

$$\|f(x_{k+1})\|_2 \leq \theta_{k+1}\left[(1-\beta_k) + \kappa\beta_k\right]\|f(x_k)\|_2 + \sum_{i=0}^{m} O(\|f(x_{k-m+i})\|_2^2), \tag{54}$$

where β_k is our damping factors and

$$\theta_{k+1} = \frac{\|\sum_{i=0}^{m} \alpha_i f(x_{k-m+i})\|_2}{\|f(x_k)\|_2}.$$

4. Implementation

For implementation, we mainly follow the path in [4] and modify it as needed. We first briefly review the implementation of AA without damping. Then we focus on how to implement the optimized damping problem efficiently and accurately.

The constrained linear least-squares problem in Algorithm 2 can be solved in many ways. Here we rewrite it into an equivalent unconstrained form which can be solved efficiently by using QR factorizations. We define $\Delta f_i = f_{i+1} - f_i$ for each i and set $\mathcal{F}_k = (\Delta f_{k-m_k}, \dots, \Delta f_{k-1})$, then the least-squares problem is equivalent to

$$\min_{\gamma=(\gamma_0,\dots,\gamma_{m_k-1})^T} \|f_k - \mathcal{F}_k\gamma\|_2,$$

where α and γ are related by $\alpha_0 = \gamma_0, \alpha_i = \gamma_i - \gamma_{i-1}$ for $1 \leq i \leq m_k - 1$, and $\alpha_{m_k} = 1 - \gamma_{m_k-1}$. We assume $\mathcal{F}$ has a thin QR decomposition i.e., $\mathcal{F}_k = Q_k R_k$ with $Q_k \in \mathcal{R}^{n\times m_k}$ and $R_k \in \mathcal{R}^{m_k\times m_k}$, for which the solution of the least-squares problem is obtained by solving the $m_k \times m_k$ triangular system $R_k\gamma = Q_k^T f_k$. As the algorithm proceeds, the successive least-squares problems can be solved efficiently by updating the factors in the decomposition.

Assume that $\gamma^k = (\gamma_0^k, \dots, \gamma_{m_k-1}^k)^T$ is the solution to the above modified form of Anderson acceleration, we have

$$x_{k+1} = g(x_k) - \sum_{i=0}^{m_k-1} \gamma_i^k \left[g(x_{k-m_k+i+1}) - g(x_{k-m_k+i})\right] = g(x_k) - \mathcal{G}_k\gamma^k,$$

where $\mathcal{G}_k = (\Delta g_{g_{k-m_k}}, \ldots, \Delta g_{k-1})$ with $\Delta g_i = g(x_{i+1} - g(x_i))$ for each i. For Anderson acceleration with damping

$$\begin{aligned} x_{k+1} &= (1-\beta_k)\sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i} + \beta_k\sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}) \\ &= \sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i} + \beta_k\left(\sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}) - \sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i}\right). \end{aligned}$$

Follow the idea in [4], we have

$$\sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}) = g(x_k) - \mathcal{G}_k\gamma^k, \tag{55}$$

$$\sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i} = \left(g(x_k) - \mathcal{G}_k\gamma^k\right) - \left(f_k - \mathcal{F}_k\gamma^k\right). \tag{56}$$

Then this can be achieved equivalently using the following strategy:

Step 1: Compute the undamped iterate $x_{k+1} = g(x_k) - \mathcal{G}_k\gamma^k$.

Step 2: Update x_{k+1} again by

$$x_{k+1} \leftarrow x_{k+1} - (1-\beta_k)\left(f_k - QR\gamma^k\right).$$

Now we talk about how to efficiently calculate β_k as described in Algorithm 3. Taking benefit of the QR decomposition in the first optimization problem and noting (55) and (56), we have

$$\begin{aligned} \tilde{x}_k^\alpha &= \sum_{i=0}^{m_k}\alpha_i^{(k)}g(x_{k-m_k+i}) = g(x_k) - \mathcal{G}_k\gamma^k, \\ x_k^\alpha &= \sum_{i=0}^{m_k}\alpha_i^{(k)}x_{k-m_k+i} = \tilde{x}_k^\alpha - \left(f_k - \mathcal{F}_k\gamma^k\right). \end{aligned}$$

Then we could calculate optimized β_k by doing two extra function evaluations and two dot products, which are not very expensive:

$$r_p = \left(x_k^\alpha - g(x_k^\alpha)\right), \quad r_q = \left(\tilde{x}_k^\alpha - g(\tilde{x}_k^\alpha)\right), \quad \beta_k = \frac{(r_p - r_q)^T r_p}{\|r_p - r_q\|_2^2}.$$

Again, if $\beta_k \notin (0,1]$, we need to set $\beta_k = 1/2$ or $\beta_k = 1$ for that iteration. In practice, we observe that our way of choosing β_k usually performs much better than the constant damping $\beta_k = 1/2$ and $\beta_k = 1$. Besides, the more optimized damping β_k values are in the interval $(0,1]$, the faster the convergence speed.

5. Experimental results and discussion

In this section, we numerically compare the performance of this non-stationary AAoptD with sAA (with constant damping, adaptive damping or without damping). The first part contains examples where larger window sizes m are needed in order to effectively accelerate the iteration. The second part consists of some harder problems from incompressible fluid dynamics. All these experiments are done in MATLAB environment and the IFISS package [32]. MATLAB codes are available upon request to the authors.

This first example is from Walker and Ni's [17] paper, where a stationary Anderson acceleration with window size $m = 50$ is used to solve the Bratu problem. This problem has a long history, we refer the reader to Glowinski et al. [33] and Pernice and Walker [34], and the references in those papers. It is not a difficult problem for Newton-like solvers.

Problem 5.1 (*The Bratu Problem*)**.** The Bratu problem is a nonlinear PDE boundary value problem as follows:

$$\begin{aligned} \Delta u + \lambda\, e^u &= 0, \quad in \quad D = [0,1]\times[0,1], \\ u &= 0, \quad on \quad \partial D. \end{aligned}$$

In this experiment, we used a centered-difference discretization on a 32×32 and 64×64 grid, respectively. We take $\lambda = 6$ in the Bratu problem and use the zero initial approximate solution in all cases. We also applied preconditioning such that the basic Picard iteration still works. The preconditioning matrix that we used here is the diagonal inverse of the matrix A, where A is a matrix for the discrete Laplace operator.

The results are shown in Figs. 1 to 5. In Fig. 1, we plot the results of applying $AAoptD(5)$ and $AA(5)$ with different constant damping factors and adaptive damping factors to accelerate Picard iteration on a grid of 32×32. As we see from the picture, $AAoptD(5)$ performs much better than $AA(5)$ with constant damping factors and $AA(5)$ with adaptive damping factors. $AA(5)$ with constant damping performs better than the one without damping and $AA(5)$ with a constant $\beta = 0.3$ is the best among those constant damping strategies. We also plot the related β_k values and gains in Fig. 2 for the top three acceleration strategies. As we know, the smaller the θ_k is, the bigger the gains are, which is consistent with the result in Fig. 1. Moreover, although only part of our β_k values in $AAoptD(5)$ are actually obtained for the optimization problem (if the "optimal" β_k does not belong to $(0,1]$, then we set $\beta_k = 1/2$),

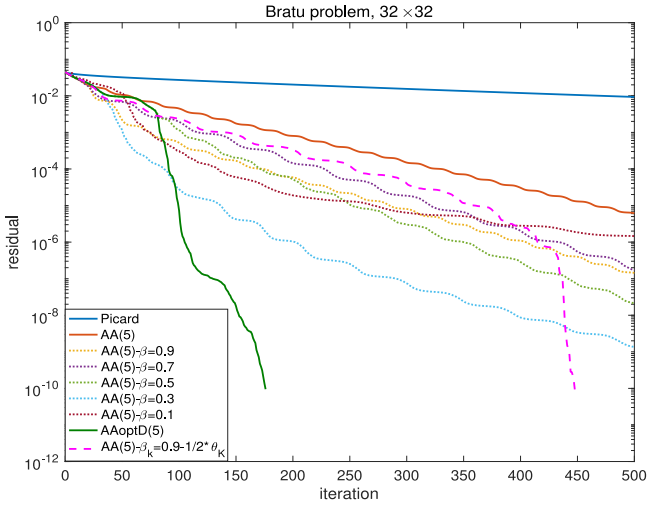


Fig. 1. Solving nonlinear Bratu problem on a 32 × 32 grid with window size $m = 5$.

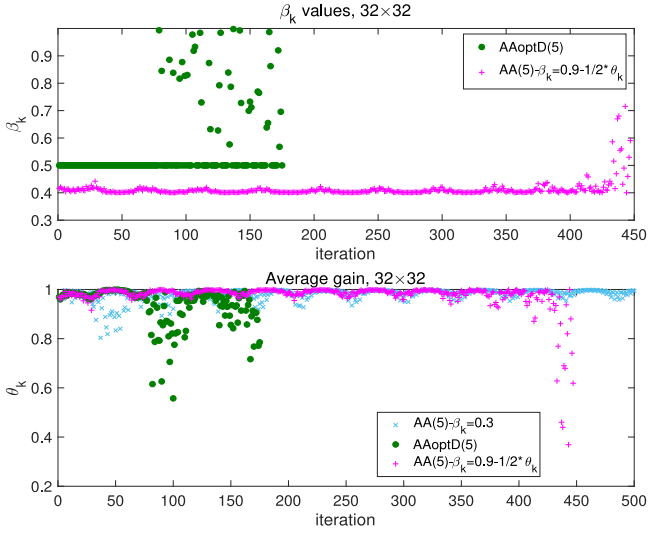


Fig. 2. Compare AA and AAoptD for solving nonlinear Bratu problem: β_k and θ_k values.

the convergence speed is much better than just using a constant damping $\beta = 1/2$. The main reason is that the local average gain is relative bigger (i.e. θ_k is smaller) when those "optimal" β_k are belong to the interval $(0, 1]$ and it will also affect the later iteration.

In Fig. 3 to Fig. 4 and Fig. A.13 to Fig. A.14 as in the appendix, we plot the results of applying $AAoptD(m)$ and $AA(m)$ with different constant damping factors and adaptive damping factors to accelerate Picard iteration with $m = 10$ and $m = 20$ on a grid of 64×64. For $m = 10$, we see from Fig. 3 that $AAoptD(10)$ is the best one. The second one is $AA(10)$ with a constant damping factor $\beta = 0.1$. $AA(10)$ with adaptive damping factors actually performs worse than stationary $AA(10)$. If we take a look at those β_k values in the adaptive damping strategy (i.e. $\beta_k = 0.9 - 1/2 * \theta_k$), most β_k values are close to 0.4 because θ_k are close to 1 as shown in Fig. A.13. Therefore, the performance of $AA(10)$ with adaptive damping factors is similar to $AA(10)$ with a constant damping factor $\beta = 0.5$, which is worse than $AA(10)$ without damping. Again, although a lot of β_k values in $AAoptD(10)$ are set to be $1/2$ since it does not belong to $(0, 1]$, it still performs much better than the constant damping factor $\beta = 0.5$. Similar results can be observed in Fig. 4 as in the appendix and Fig. A.14 for $m = 20$. Moreover, in Fig. 5, we observe that $AAoptD(m)$ with a small window size $m = 10$ works better than stationary $AA(m)$ with larger window size $m = 50$.

Problem 5.2 (*The Nonlinear Convection–Diffusion Problem*)**.** Use AA and AAoptD to solve the following 2D nonlinear convection–diffusion equation in a square region:

$$(-u_{xx} - u_{yy}) + (u_x + u_y) + ku^2 = f(x, y), \quad (x, y) \in D = [0, 1] \times [0, 1]$$

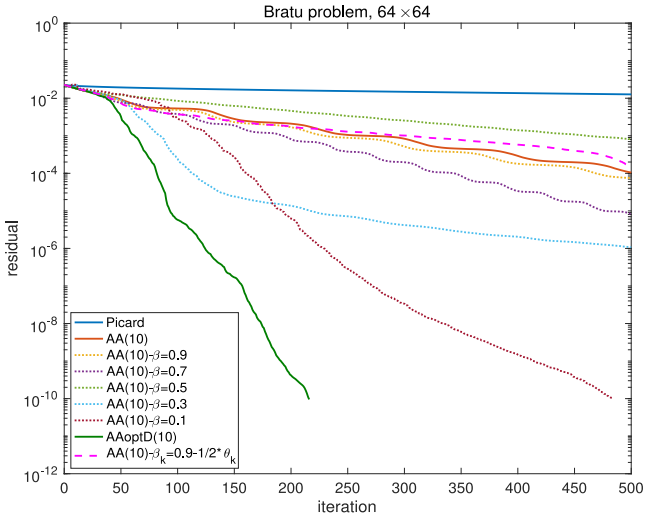


Fig. 3. Solving nonlinear Bratu problem on a 64 × 64 grid with window size $m = 10$.

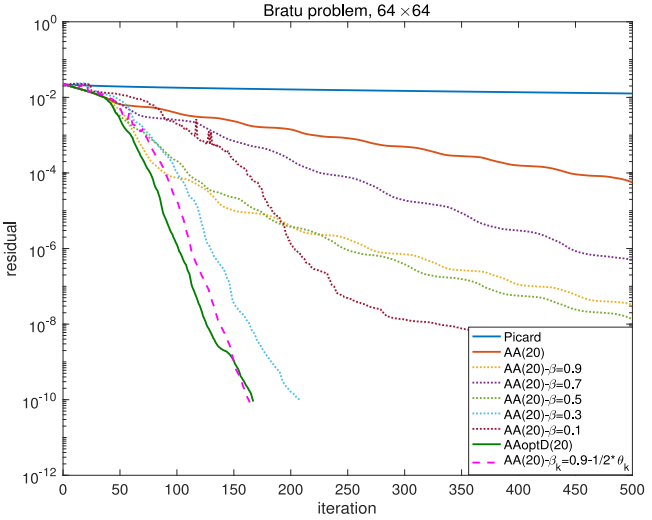


Fig. 4. Solving nonlinear Bratu problem on a 64 × 64 grid with window size $m = 20$.

with the source term

$$f(x, y) = 2\pi^2 \sin(\pi x) \sin(\pi y)$$

and zero boundary conditions: $u(x, y) = 0$ on ∂D.

In this numerical experiment, we use the centered-difference discretization for diffusion term and use the upwind scheme (backward difference) for the convection term on a 64 × 64 grid. We take $k = 3$ in the above problem and use $u_0 = (1, 1, \ldots, 1)^T$ as an initial approximate solution in all cases. As in solving the Bratu problem, the same preconditioning strategy is used here so that the basic Picard iteration still works.

The results are shown in Figs. 6 to 9. In Fig. 6, we plot the results of applying $AAoptD(10)$ and $AA(10)$ with different constant damping factors and adaptive damping factors to accelerate Picard iteration on a grid of 64 × 64. $AAoptD(10)$ are better than other methods. In this case, the constant damping strategies and adaptive damping strategy are even worse than $AA(10)$ without damping.

Problem 5.3 (*2D Lid-Driven Cavity*)**.** The 2D lid-driven cavity uses a domain $\Omega = (-1, 1)^2$ with no-slip boundary conditions on the sides and bottom and a "moving lid" on the top which is implemented by enforcing the Dirichlet boundary condition:

$$\{y = 1; -1 \leq x \leq 1 | u_x = 1 - x^4\} \quad a\ regularized cavity.$$

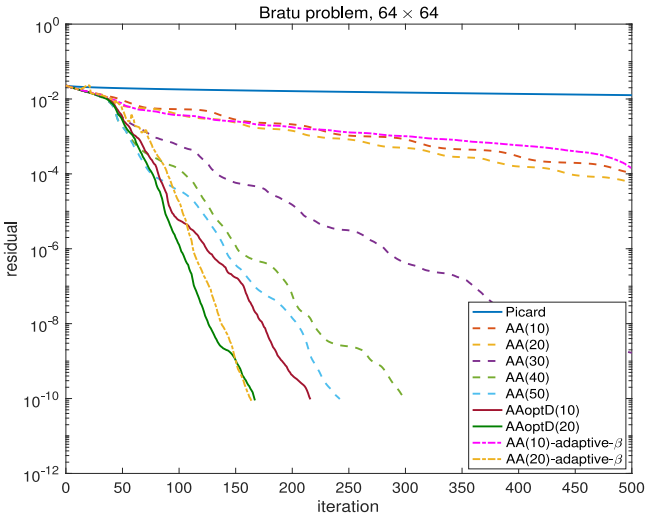


Fig. 5. Solving nonlinear Bratu problem on a 64 × 64 grid with larger window size.

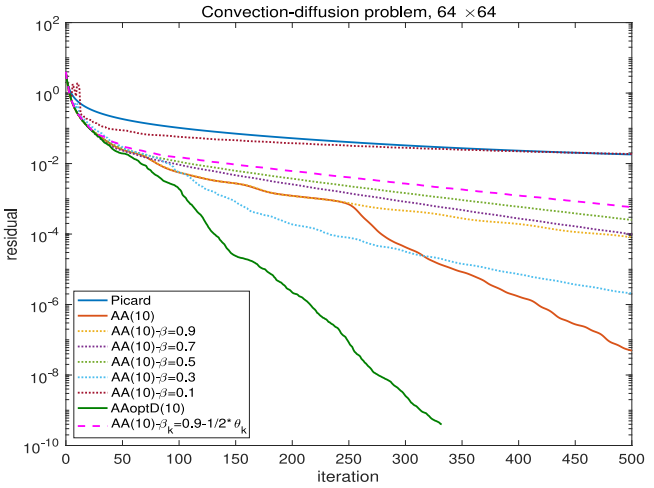


Fig. 6. Solving the convection–diffusion equation on a 64 × 64 grid with $m = 10$.

There is no forcing and the kinematic viscosity ν is set to be $\nu = \frac{1}{Re}$. We discretize with $(Q2 - Q1)$ Taylor–Hood elements on a 64 × 64 non-uniform mesh and use the corresponding discrete Stokes solution as the initial guess.

We test it with Reynolds numbers of $Re = 1000$, $Re = 2500$ and $Re = 5000$, respectively. In Fig. A.15 as in the appendix, we plot the streamlines of the velocity and the pressure field for different Reynolds numbers. In Fig. 10(a), we plot the convergence rate for the case where $Re = 1000$ and use the window size $m = 2$. Our proposed method $AAoptD(m)$ performs better than other damping strategies. We also observe that the constant damping $\beta = 0.5$, $\beta = 0.1$ and the adaptive damping $\beta_k = 0.9 - 1/2 * \theta_k$ are all doing even worse than $AA(2)$ without damping. Thus choosing the wrong damping factors may slow down the speed of convergence rate. We also plot the β_k values used in our $AAoptD(2)$ and the adaptive damping method in Fig. 10(b). We can see from this figure that those adaptive damping factors are around 0.4. This explains why the adaptive strategy behaves like the constant damping $\beta_k = 0.5$. Besides, although around half of the β_k values used in $AAoptD(2)$ equal 0.5, the performance is much better than the constant damping with $\beta_k = 0.5$. Similar results are obtained for $Re = 2500$ as in Fig. 10(c) and (d). In this case, $AAoptD(2)$ is much better than other methods since most "local optimal" β_k are achieved in the interval $(0, 1]$. When we keep increasing the Reynolds number to $Re = 5000$, our proposed method $AAoptD(2)$ and other damping strategies do not gain very much over $AA(2)$ without damping.

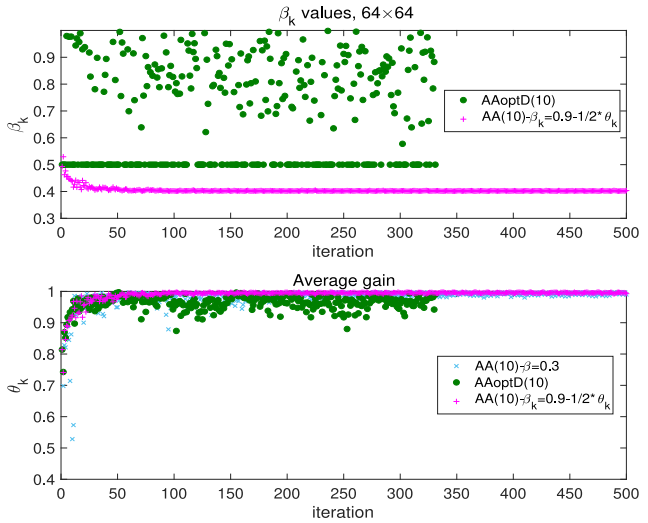


Fig. 7. Solving the convection–diffusion equation: β_k and θ_k values.

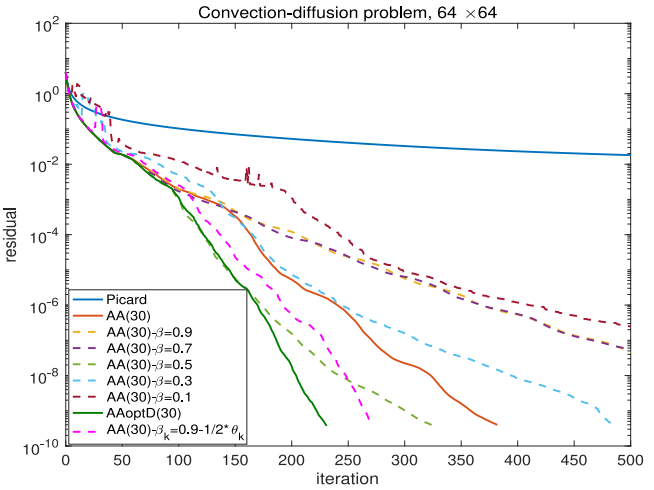


Fig. 8. Solving the convection–diffusion equation on a 64 × 64 grid with $m = 30$.

Problem 5.4 (*Flow Over a Backward Facing Step*)**.** This example represents slow flow in a rectangular duct with a sudden expansion on a L-shaped region generated by taking the complement in $(-1, L) \times (-1, 1)$ of the quadrant $(-1, 0) \times (-1, 0)$. A Poiseuille flow profile is imposed on the inflow boundary $(x = -1; 0 \leq y \leq 1)$, and a zero velocity condition is imposed on the walls. At the outflow boundary, a natural outflow boundary condition is imposed and the mean outflow pressure is set to be zero. We discretize with $(Q2 - Q1)$ Taylor–Hood elements on a 64 × 192 non-uniform mesh and use the corresponding discrete Stokes solution as the initial guess.

We test it with Reynolds numbers of $Re = 150$ and $Re = 250$, respectively. In Fig. A.16 as in the appendix, we plot the streamlines and pressure field for $Re = 150$ and $Re = 250$. The convergence results are shown in Fig. 11. For both cases, our proposed method $AAopt D$ performs better than the constant and adaptive damping strategies. The β_k values for each case are shown in Fig. 11(b) and (d). From Fig. 11(b), we observe again that even if only few "local optimal" β_k values are achieved in $AAopt D(2)$ (most values are set to equal 0.5 since it does in the interval $(0, 1]$), it can much better than the constant damping $\beta_k = 0.5$.

Problem 5.5 (*Flow Over an Obstacle*)**.** This is another classical problem. The domain is a disconnected rectangular region $(0, 8)\times(-1, 1)$ generated by deleting the square $(7/4, 9/4) \times (-1/4, 1/4)$. And it is associated with modeling flow in a rectangular channel with a square cylindrical obstruction. A Poiseuille profile is imposed on the inflow boundary $(x = 0; -1 \leq y \leq 1)$, and a zero velocity

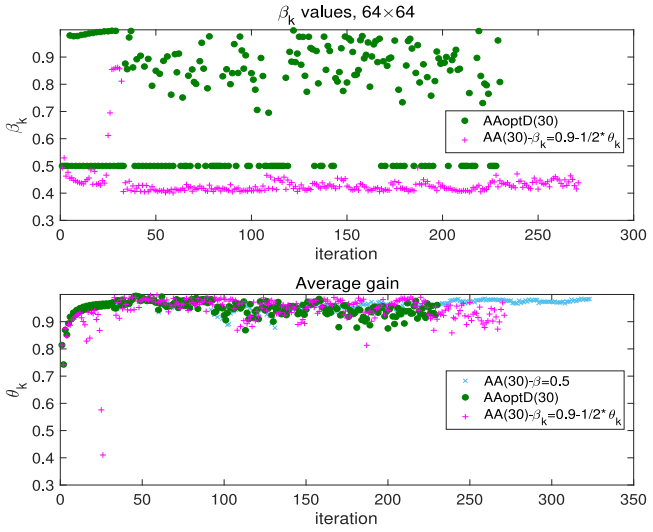


Fig. 9. Solving the convection–diffusion equation: β_k and θ_k values.

condition is imposed on the obstruction and on the top and bottom walls. A natural Neumann outflow condition is applied at the outflow boundary which automatically sets the mean outflow pressure to zero. We discretize with $(Q2-Q1)$ Taylor–Hood elements on a 32×80 non-uniform mesh and use the corresponding discrete Stokes solution as the initial guess.

We test it with Reynolds numbers of $Re = 300$ and $Re = 600$, respectively. In Fig. A.17 as in the appendix, we plot the streamlines and pressure field for $Re = 300$ and $Re = 600$. The convergence results are shown in Fig. 12. From Fig. 12(a), we see that the performance of $AAoptD(2)$ is very close to $AA(2)$. Other damping strategies work even worse. For $Re = 600$, we see that $AAoptD(2)$ is much better than other methods, as shown in Fig. 12(c).

6. Conclusions

We proposed and analyzed a non-stationary Anderson acceleration algorithm with an optimized damping factor in each iteration to further speed up linear and nonlinear iterations by applying one extra optimization. By taking advantage of the QR decomposition in the first optimization problem, the calculation of optimized β_k at each iteration is cheap if two extra function evaluations are relatively inexpensive. This procedure has a strong connection to another perspective of generating non-stationary AA (i.e. varying the window size m at different iterations). Our numerical results show that the gain of doing this extra optimized step on β_k sometimes could be very beneficial. Therefore, when the stationary AA is not working well or a larger size of the window is needed in AA, we recommend using AAoptD proposed in the present work. Besides, we also observed from our numerical results that damping can be good but choosing the wrong damping factors may slow down the convergence rate.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China [grant number 12001287]; the Startup Foundation for Introducing Talent of Nanjing University of Information Science and Technology, China [grant number 2019r106]; The first author Kewang Chen also gratefully acknowledge the financial support for his doctoral study provided by the China Scholarship Council (No. 202008320191). Moreover, the authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

Appendix. Figures

See Figs. A.13–A.17.

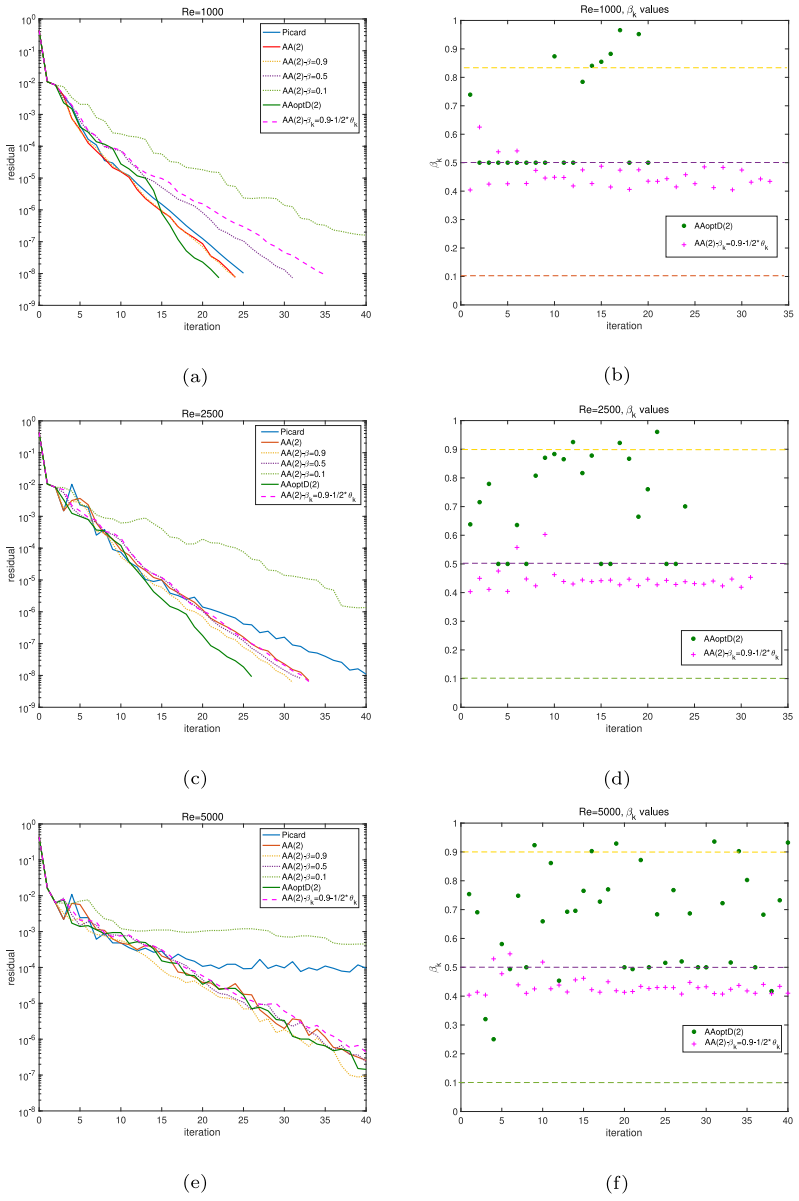


Fig. 10. Lid-driven cavity: (a) $Re = 1000$, convergence results; (b) $Re = 1000$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping; (c) $Re = 2500$, convergence results; (d) $Re = 2500$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping; (e) $Re = 5000$, convergence results; (f) $Re = 5000$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping.

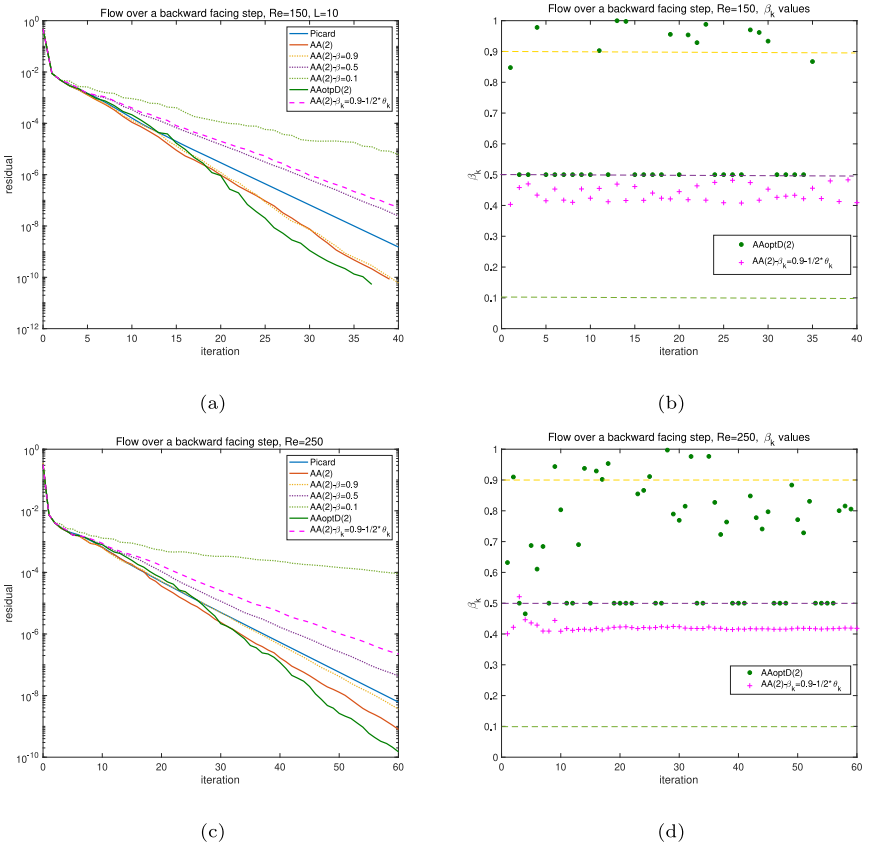


Fig. 11. Flow over a backward-facing step: (a) $Re = 150$, convergence results; (b) $Re = 150$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping; (c) $Re = 250$, convergence results; (d) $Re = 250$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping.

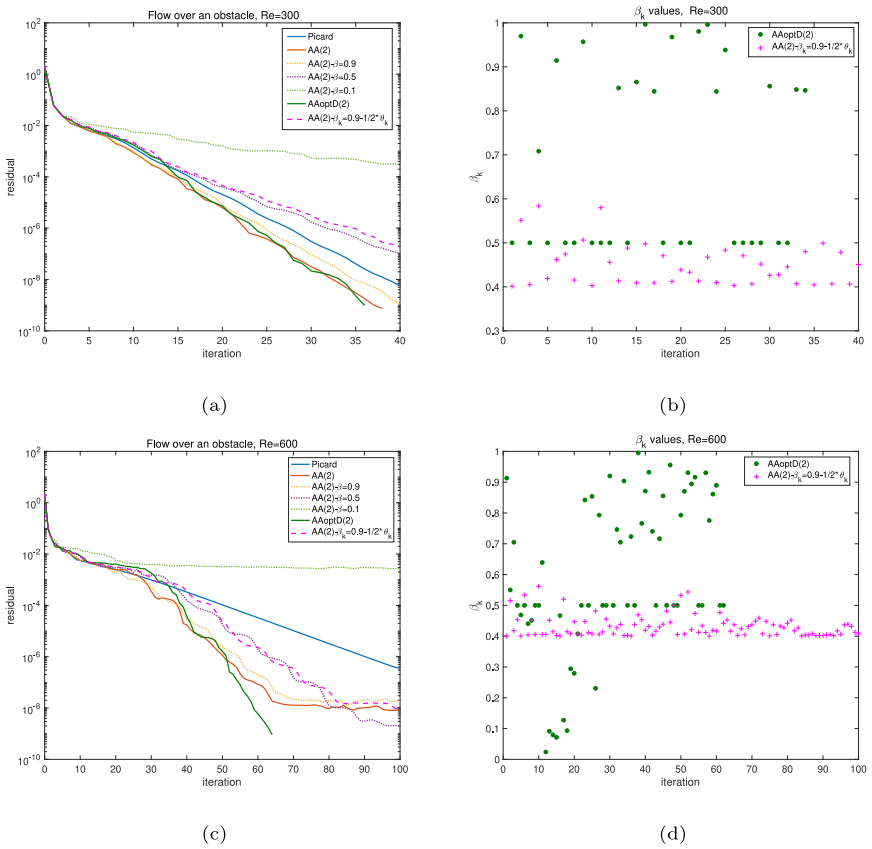


Fig. 12. Flow over an obstacle: (a) $Re = 300$, convergence results; (b)$Re = 300$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping; (c) $Re = 600$, convergence results; (d) $Re = 600$, β_k values of $AAoptD(2)$ and $AA(2)$ with adaptive damping.

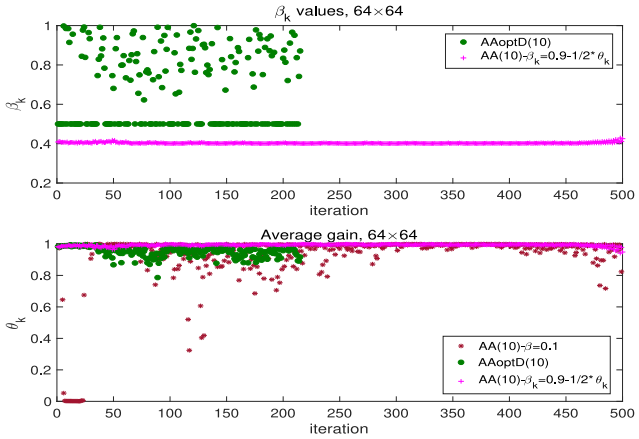


Fig. A.13. Solving nonlinear Bratu problem: β_k and θ_k values, $m = 10$.

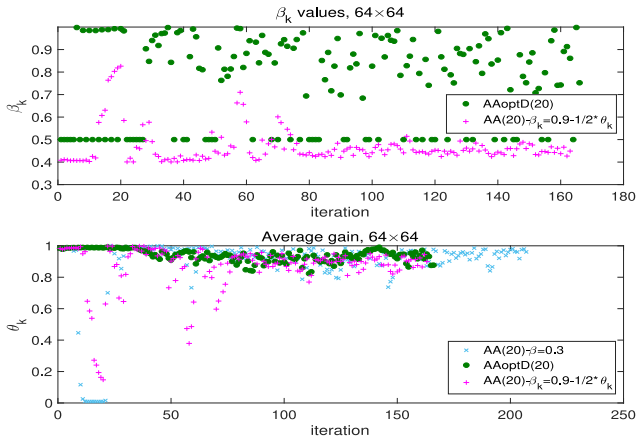


Fig. A.14. Solving nonlinear Bratu problem: β_k and θ_k values, $m = 20$.

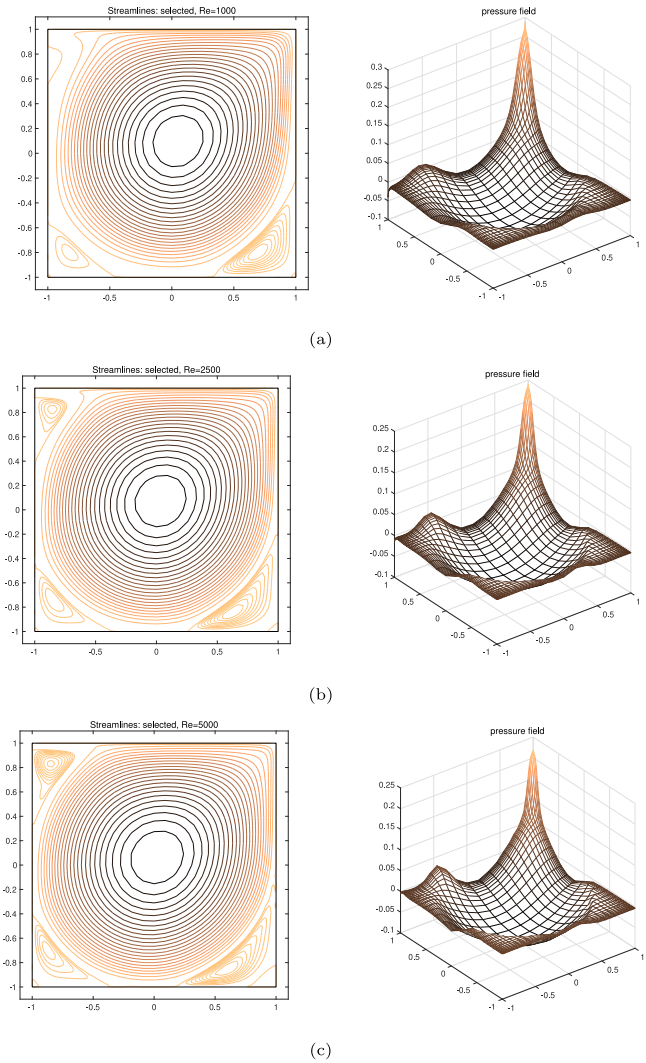
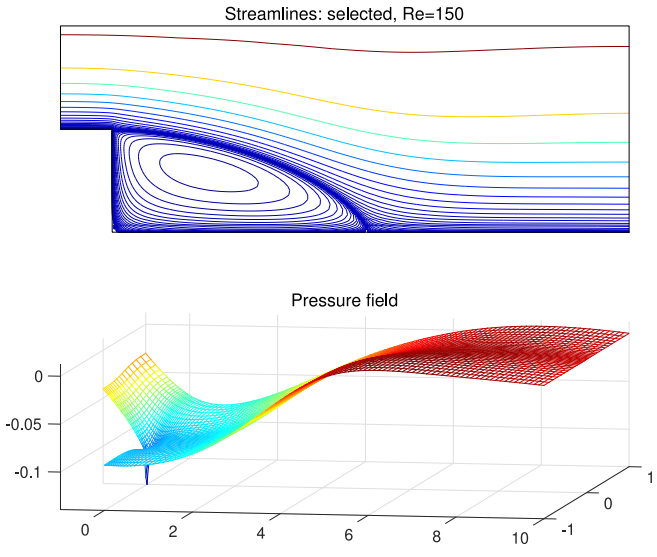
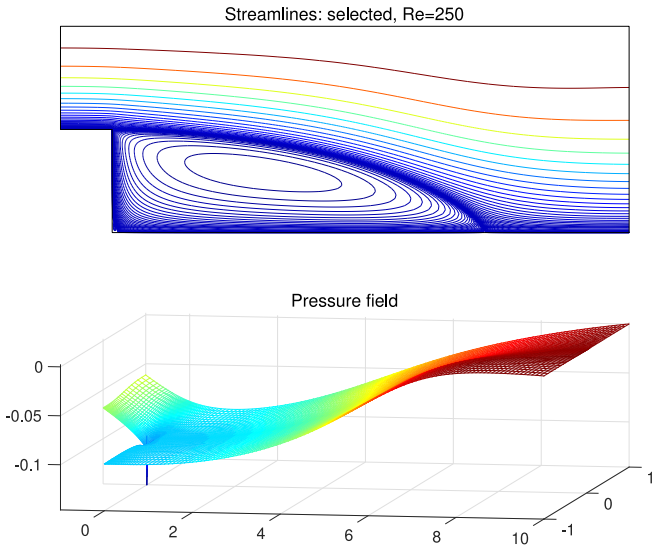


Fig. A.15. Lid-driven cavity: (a) $Re = 1000$, solution; (b) $Re = 2500$, solution; (c) $Re = 5000$, solution.

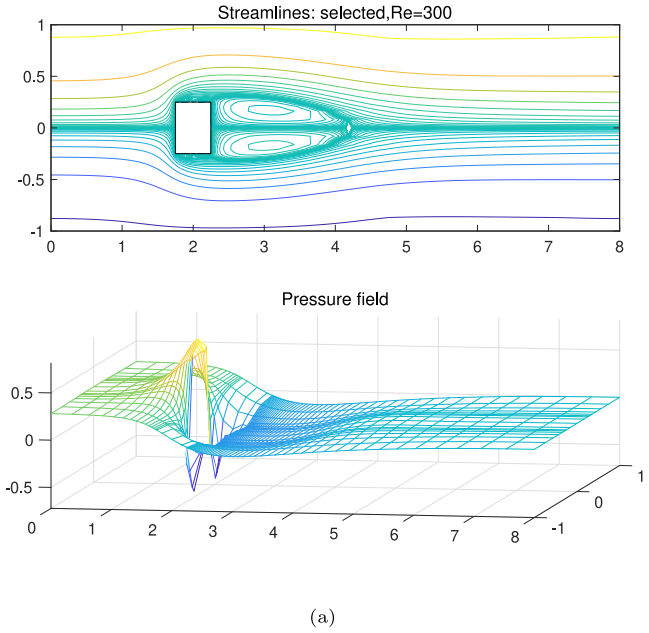


(a)

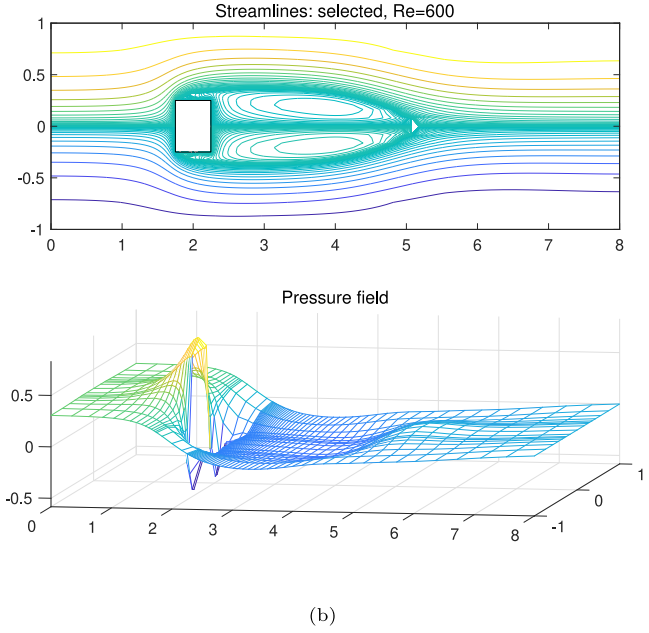


(b)

Fig. A.16. Flow over a backward facing step: (a) $Re = 150$, solution; (b) $Re = 250$, solution.



(a)



(b)

Fig. A.17. Flow over an obstacle: (a) $Re = 300$, solution; (b) $Re = 600$, solution.

References

[1] D.G. Anderson, Iterative procedures for nonlinear integral equations, J. ACM 12 (1965) 547–560, http://dx.doi.org/10.1145/321296.321305.

[2] D.G.M. Anderson, Comments on "Anderson acceleration, mixing and extrapolation", Numer. Algorithms 80 (1) (2019) 135–234, http://dx.doi.org/10.1007/s11075-018-0549-4.

[3] A. Toth, C.T. Kelley, Convergence analysis for Anderson acceleration, SIAM J. Numer. Anal. 53 (2) (2015) 805–819, http://dx.doi.org/10.1137/130919398.

[4] H.F. Walker, Anderson acceleration: Algorithms and implementations, in: WPI Math. Sciences Dept. Report MS-6-15-50, 2011, URL https://users.wpi.edu/~walker/Papers/anderson_accn_algs_imps.pdf.

[5] N.N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code. I. In one dimension, SIAM J. Sci. Comput. 19 (3) (1998) 728–765, http://dx.doi.org/10.1137/S106482759426955X.

[6] K. Miller, Nonlinear krylov and moving nodes in the method of lines, J. Comput. Appl. Math. 183 (2) (2005) 275–287, http://dx.doi.org/10.1016/j.cam.2004.12.032.

[7] C.W. Oosterlee, T. Washio, Krylov subspace acceleration of nonlinear multigrid with application to recirculating flows, SIAM J. Sci. Comput. 21 (5) (2000) 1670–1690, http://dx.doi.org/10.1137/S1064827598338093.
[8] T. Washio, C.W. Oosterlee, Krylov subspace acceleration for nonlinear multigrid schemes, Electron. Trans. Numer. Anal. 6 (Dec.) (1997) 271–290, URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.147.3799.
[9] L. Lin, C. Yang, Elliptic preconditioner for accelerating the self-consistent field iteration in Kohn-Sham density functional theory, SIAM J. Sci. Comput. 35 (5) (2013) S277–S298, http://dx.doi.org/10.1137/120880604.
[10] P. Pulay, Convergence acceleration of iterative sequences. the case of SCF iteration, Chem. Phys. Lett. 73 (2) (1980) 393–398, http://dx.doi.org/10.1016/0009-2614(80)80396-4.
[11] P. Pulay, Improved SCF convergence acceleration, J. Comput. Chem. 3 (4) (1982) 556–560, http://dx.doi.org/10.1002/jcc.540030413.
[12] T. Eirola, O. Nevanlinna, Accelerating with rank-one updates, Linear Algebra Appl. 121 (1989) 511–520, http://dx.doi.org/10.1016/0024-3795(89)90719-2.
[13] V. Eyert, A comparative study on methods for convergence acceleration of iterative vector sequences, J. Comput. Phys. 124 (2) (1996) 271–285, http://dx.doi.org/10.1006/jcph.1996.0059.
[14] H.-r. Fang, Y. Saad, Two classes of multisecant methods for nonlinear acceleration, Numer. Linear Algebra Appl. 16 (3) (2009) 197–221, http://dx.doi.org/10.1002/nla.617.
[15] R. Haelterman, J. Degroote, D. Van Heule, J. Vierendeels, On the similarities between the quasi-Newton inverse least squares method and GMRES, SIAM J. Numer. Anal. 47 (6) (2010) 4660–4679, http://dx.doi.org/10.1137/090750354.
[16] C. Yang, J.C. Meza, B. Lee, L.-W. Wang, KSSOLV—a MATLAB toolbox for solving the Kohn-Sham equations, ACM Trans. Math. Software 36 (2) (2009) http://dx.doi.org/10.1145/1499096.1499099, Art. 10, 35.
[17] H.F. Walker, P. Ni, Anderson acceleration for fixed-point iterations, SIAM J. Numer. Anal. 49 (4) (2011) 1715–1735, http://dx.doi.org/10.1137/10078356X.
[18] C. Evans, S. Pollock, L.G. Rebholz, M. Xiao, A proof that Anderson acceleration improves the convergence rate in linearly converging fixed-point methods (but not in those converging quadratically), SIAM J. Numer. Anal. 58 (1) (2020) 788–810, http://dx.doi.org/10.1137/19M1245384.
[19] S. Pollock, L.G. Rebholz, M. Xiao, Anderson-accelerated convergence of Picard iterations for incompressible Navier-Stokes equations, SIAM J. Numer. Anal. 57 (2) (2019) 615–637, http://dx.doi.org/10.1137/18M1206151.
[20] H. De Sterck, Y. He, On the asymptotic linear convergence speed of Anderson acceleration, Nesterov acceleration, and nonlinear GMRES, SIAM J. Sci. Comput. 43 (5) (2021) S21–S46, http://dx.doi.org/10.1137/20M1347139.
[21] D. Wang, Y. He, H. De Sterck, On the asymptotic linear convergence speed of Anderson acceleration applied to ADMM, J. Sci. Comput. 88 (2) (2021) http://dx.doi.org/10.1007/s10915-021-01548-2, Paper No. 38, 35.
[22] J. Zhang, B. O'Donoghue, S. Boyd, Globally convergent type-I Anderson acceleration for nonsmooth fixed-point iterations, SIAM J. Optim. 30 (4) (2020) 3170–3197, http://dx.doi.org/10.1137/18M1232772.
[23] W. Bian, X. Chen, C.T. Kelley, Anderson acceleration for a class of nonsmooth fixed-point problems, SIAM J. Sci. Comput. 43 (5) (2021) S1–S20, http://dx.doi.org/10.1137/20M132938X.
[24] P.R. Brune, M.G. Knepley, B.F. Smith, X. Tu, Composing scalable nonlinear algebraic solvers, SIAM Rev. 57 (4) (2015) 535–565, http://dx.doi.org/10.1137/130936725.
[25] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, L. Liu, Anderson acceleration for geometry optimization and physics simulation, ACM Trans. Graph. 37 (4) (2018) 1–14, http://dx.doi.org/10.1145/3197517.3201290.
[26] A. Toth, J.A. Ellis, T. Evans, S. Hamilton, C.T. Kelley, R. Pawlowski, S. Slattery, Local improvement results for Anderson acceleration with inaccurate function evaluations, SIAM J. Sci. Comput. 39 (5) (2017) S47–S65, http://dx.doi.org/10.1137/16M1080677.
[27] W. Shi, S. Song, H. Wu, Y.-C. Hsu, C. Wu, G. Huang, Regularized Anderson acceleration for off-policy deep reinforcement learning, 2019, arXiv preprint arXiv:1909.03245, URL https://arxiv.org/abs/1909.03245.
[28] Y. Yang, Anderson acceleration for seismic inversion, Geophysics 86 (1) (2021) R99–R108, http://dx.doi.org/10.1190/geo2020-0462.1.
[29] F.A. Potra, H. Engler, A characterization of the behavior of the Anderson acceleration on linear problems, linear Algebra Appl. 438 (3) (2013) 1002–1011, http://dx.doi.org/10.1016/j.laa.2012.09.008.
[30] H. De Sterck, A nonlinear GMRES optimization algorithm for canonical tensor decomposition, SIAM J. Sci. Comput. 34 (3) (2012) A1351–A1379, http://dx.doi.org/10.1137/110835530.
[31] C.T. Kelley, Iterative Methods for Linear and Nonlinear Equations, SIAM, 1995, http://dx.doi.org/10.1137/1.9781611970944.
[32] H.C. Elman, A. Ramage, D.J. Silvester, IFISS: A computational laboratory for investigating incompressible flow problems, SIAM Review 56 (2) (2014) 261–273, http://dx.doi.org/10.1137/120891393.
[33] R. Glowinski, H.B. Keller, L. Reinhart, Continuation-conjugate gradient methods for the least squares solution of nonlinear boundary value problems, SIAM J. Sci. Stat. Comput. 6 (4) (1985) 793–832, http://dx.doi.org/10.1137/0906055.
[34] M. Pernice, H.F. Walker, NITSOL: a Newton iterative solver for nonlinear systems, SIAM J. Sci. Comput. 19 (1) (1998) 302–318, http://dx.doi.org/10.1137/S1064827596303843.