Received: 4 May 2018 | Revised: 14 March 2019 | Accepted: 14 March 2019

DOI: 10.1002/nla.2241

RESEARCH ARTICLE

WILEY

Convergence analysis of Anderson-type acceleration of Richardson's iteration

Massimiliano Lupo Pasini[1,2]

[1]Department of Mathematics and Computer Science, Emory University, Atlanta, Georgia

[2]National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, Tennessee

Correspondence
Massimiliano Lupo Pasini was with the Department of Mathematics and Computer Science, Emory University, Atlanta, GA 30322. He is now at the National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, TN 37830.
Email: massimiliano.lupo.pasini@emory.edu; lupopasinim@ornl.gov

Present Address
Massimiliano Lupo Pasini, 1 Bethel Valley Road, PO 2008, MS6008, Oak Ridge, TN 37830

Funding information
United States Department of Energy (Office of Science), Grant/Award Number: ERKJ247; U.S. Department of Energy at Oak Ridge National Laboratory, Grant/Award Number: DE-AC05-00OR22725

Summary

We consider Anderson extrapolation to accelerate the (stationary) Richardson iterative method for sparse linear systems. Using an Anderson mixing at periodic intervals, we assess how this benefits convergence to a prescribed accuracy. The method, named alternating Anderson–Richardson, has appealing properties for high-performance computing, such as the potential to reduce communication and storage in comparison to more conventional linear solvers. We establish sufficient conditions for convergence, and we evaluate the performance of this technique in combination with various preconditioners through numerical examples. Furthermore, we propose an augmented version of this technique.

KEYWORDS

Anderson acceleration, fixed-point scheme, projection method, Richardson iteration

1 | INTRODUCTION

In this work, we analyze recently proposed linear solvers that aim to leverage the performance of new computing architectures. In fact, standard linear solvers face many challenges to achieve efficiency in this new computational context. On the one hand, there are Krylov subspace methods[1] that are widely employed in scientific computing to solve large-scale linear systems. These methods require global operations such as inner products at each iteration. However, as global operations

Alternating Anderson-Richardson

hinder the parallelization, these techniques struggle to efficiently exploit the parallel computing potential of new architectures. On the other hand, there are standard Richardson iterations that are better suited for massive parallelization because of a strong locality of the operations required (e.g., matrix–vector multiplications). However, their convergence is generally too slow. Moreover, conditions to guarantee convergent splittings may be too demanding. Another category of linear solvers is represented by Chebyshev iterations.[2,3] Differently from Krylov methods, Chebyshev iterations do not need to compute any inner products. However, the effectiveness of these techniques depends on knowledge about the spectrum of the coefficient matrix A. In particular, if A is symmetric, then the algorithm needs the extreme eigenvalues of A. Otherwise, if A is nonsymmetric, then the iteration requires knowledge of an ellipse in the complex plane that contains the spectrum and excludes the origin. Such knowledge is difficult or often impossible to obtain, and this makes Chebyshev iterations impractical as well.

While several authors have explored the opportunity to enhance the performance of Krylov methods based on the new requirements,[4–6] less attention has been paid to the advantages that stationary Richardson can provide to next-generation computers. The possibility of leveraging the performance of fixed-point schemes has been recently reconsidered.[7] However, adapting acceleration techniques for Richardson schemes suitable for the new hardware/software issues is still an open question. The main goal is to maintain the locality and simplicity of standard Richardson as much as possible, pursuing a trade-off between parallelizability and efficiency. One recent contribution in this direction has been made by Pratapa et al.[8,9] In these papers, the authors proposed a new strategy called *alternating Anderson–Richardson* (AAR), which applies Anderson extrapolation[10,11] at periodic intervals within the classical Richardson iteration. Promising numerical results have been shown with AAR outperforming the generalized minimum residual (GMRES)[12] for some classes of problems. However, no theoretical analysis of this scheme has been conducted so far.

In the present work, we study the mathematical properties of AAR in comparison with more standard techniques such as full GMRES,[1,12] truncated GMRES,[1] restarted GMRES,[1] and Anderson–Richardson.[13–17] All the convergence analyses and comparisons between algorithms conducted in this work assume exact arithmetic.

The remainder of this paper is organized as follows. In Section 2, we briefly recall the (one-level) Richardson method. In Section 3, we introduce the AAR algorithm, and we establish convergence results, followed by numerical experiments in Section 4, the goal of which is to validate the analyses conducted in the previous sections. We conclude this paper with remarks on the state of the art and comments about future developments.

2 | STATIONARY RICHARDSON

Consider a nonsingular linear system of the form

$$A\mathbf{x} = \mathbf{b}, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ and $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$. We assume that left preconditioning has already been incorporated. The equation can be recast as a fixed-point problem that iteratively updates the approximation to $\mathbf{x}$, that is,

$$\mathbf{x}^{k+1} = H\mathbf{x}^k + \mathbf{b}, \quad k = 0, 1, \ldots, \tag{2}$$

where $H = I - A$ is the iteration matrix and $\mathbf{x}^0$ is an initial guess. Another equivalent representation, called correction form, is commonly adopted as follows:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{r}^k, \quad k = 0, 1, \ldots, \tag{3}$$

where $\mathbf{r}^k = \mathbf{b} - A\mathbf{x}^k$ is the residual at the kth iteration. A necessary and sufficient condition for the scheme in (3) to converge to the solution in exact arithmetic is that the spectral radius $\rho(H) < 1$. Scheme (2) can be generalized by introducing a positive weighing parameter ω, as follows:

$$\mathbf{x}^{k+1} = (1-\omega)\mathbf{x}^k + \omega(H\mathbf{x}^k + \mathbf{b}), \tag{4}$$

which, in correction form, is equivalently represented as

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \omega\mathbf{r}^k. \tag{5}$$

Equations (4) and (5) are known as the *stationary Richardson scheme*, and the iteration matrix associated with this fixed-point scheme is

$$H_\omega = I - \omega A.$$

2.1 | Anderson mixing

A possible approach to accelerate an iterative method was proposed in the work of Anderson.[10] The technique presented in the aforementioned work aims to compute a vector

$$\mathbf{g}^k = \left(g_1^k, \ldots, g_m^k\right)^T \in \mathbb{R}^m$$

to correct the approximation $\mathbf{x}^k$ as follows:

$$\overline{\mathbf{x}}^k = \mathbf{x}^k - \sum_{j=1}^{m} g_j^k (\mathbf{x}^{k-m+j} - \mathbf{x}^{k-m+j-1}), \tag{6}$$

where the vector $\mathbf{g}^k$ is computed in order to minimize the ℓ^2-norm of the updated residual

$$\overline{\mathbf{r}}^k = \mathbf{b} - A\overline{\mathbf{x}}^k. \tag{7}$$

The procedure proposed in Equation (6) to compute $\overline{\mathbf{x}}^k$ is called *Anderson mixing*. Define

$$X_k = \left[(\mathbf{x}^{k-m+1} - \mathbf{x}^{k-m}), (\mathbf{x}^{k-m+2} - \mathbf{x}^{k-m+1}), \ldots, (\mathbf{x}^k - \mathbf{x}^{k-1})\right] \in \mathbb{R}^{n\times m} \tag{8}$$

and

$$R_k = \left[(\mathbf{r}^{k-m+1} - \mathbf{r}^{k-m}), (\mathbf{r}^{k-m+2} - \mathbf{r}^{k-m+1}), \ldots, (\mathbf{r}^k - \mathbf{r}^{k-1})\right] \in \mathbb{R}^{n\times m}. \tag{9}$$

Note that $R_k = -AX_k$. We can recast Equation (6) as

$$\overline{\mathbf{x}}^k = \mathbf{x}^k - X_k\mathbf{g}^k,$$

where the optimal vector $\mathbf{g}^k$ is computed as

$$\mathbf{g}^k = \underset{\mathbf{g}\in\mathbb{R}^m}{\operatorname{argmin}} \left\|\mathbf{b} - A\left(\mathbf{x}^k - X_k\mathbf{g}\right)\right\|_2^2.$$

Under the assumption that R_k has linearly independent columns, we can express $\mathbf{g}^k$ as

$$\mathbf{g}^k = \left(R_k^T R_k\right)^{-1} R_k^T \mathbf{r}^k.$$

The eventuality of linearly dependent columns of R_k is discussed in Remark 4. Another way to write Equation (6) is

$$\begin{aligned}\overline{\mathbf{x}}^k &= \mathbf{x}^k - X_k\mathbf{g}^k \\ &= \mathbf{x}^k - X_k\left(R_k^T R_k\right)^{-1} R_k^T \mathbf{r}^k \\ &= \mathbf{x}^k + X_k\left(R_k^T A X_k\right)^{-1} R_k^T \mathbf{r}^k.\end{aligned} \tag{10}$$

Therefore, Anderson mixing can be interpreted as a step of a *projection method* (see chapter 5 in the work of Saad[1]) with the subspace of corrections chosen as

$$\mathcal{V}_k = \mathcal{R}(X_k)$$

and the subspace of constraints as

$$\mathcal{W}_k = \mathcal{R}(R_k) = A\mathcal{R}(X_k) = A\mathcal{V}_k.$$

After the Anderson mixing is applied, $\overline{\mathbf{x}}^k$ is used to compute a new update via a standard Richardson's step, that is,

$$\mathbf{x}^{k+1} = \overline{\mathbf{x}}^k + \omega\overline{\mathbf{r}}^k.$$

Remark 1. The Anderson mixing is sometimes described in the literature in a slightly different form than Equation (6). In fact, defining

$$\hat{\mathbf{g}}^k = \left(\hat{g}_1^k, \ldots, \hat{g}_m^k\right)^T \in \mathbb{R}^m,$$

an alternative definition is

$$\overline{\mathbf{x}}^k = \mathbf{x}^{k-m} - \sum_{j=1}^{m} \hat{g}_j^k (\mathbf{x}^{k-m+j} - \mathbf{x}^{k-m}). \tag{11}$$

By defining the matrices $\hat{X}_k \in \mathbb{R}^{n\times m}$ and $\hat{R}_k \in \mathbb{R}^{n\times m}$ as follows:

$$\hat{X}_k = \left[(\mathbf{x}^{k-m+1} - \mathbf{x}^{k-m}), (\mathbf{x}^{k-m+2} - \mathbf{x}^{k-m}), \ \dots, (\mathbf{x}^{k} - \mathbf{x}^{k-m})\right] \in \mathbb{R}^{n\times m}, \tag{12}$$

$$\hat{R}_k = \left[(\mathbf{r}^{k-m+1} - \mathbf{r}^{k-m}), (\mathbf{r}^{k-m+2} - \mathbf{r}^{k-m}), \ \dots, (\mathbf{r}^{k} - \mathbf{r}^{k-m})\right] \in \mathbb{R}^{n\times m}, \tag{13}$$

Equation (11) is recast as

$$\overline{\mathbf{x}}^k = \mathbf{x}^{k-m} - \hat{X}_k \hat{\mathbf{g}}^k,$$

where the optimal vector $\hat{\mathbf{g}}^k$ is computed as

$$\begin{aligned} \hat{\mathbf{g}}^k &= \underset{\mathbf{g}\in\mathbb{R}^m}{\operatorname{argmin}} \left\| \mathbf{b} - A\left(\mathbf{x}^{k-m} - \hat{X}_k \mathbf{g}\right)\right\|_2^2 \\ &= \left(\hat{R}_k^T \hat{R}_k\right)^{-1} \hat{R}_k^T \mathbf{r}^{k-m}. \end{aligned}$$

Anderson mixing is thus well suited to potentially accelerate Richardson schemes. Contributions in the mathematical literature in this respect have resulted in an algorithm called the *Anderson–Richardson* method (AR for short).[10,15,17] Studies about AR have highlighted similarities between this method and GMRES. However, AR is seldom used to solve linear systems. In fact, having to solve a least squares problem at each iteration increases communication, which may affect the performance on next-generation computers. Pratapa et al. have recently proposed a new way to combine Richardson schemes and Anderson mixing, which aims to reduce the communication by relaxing the number of least squares problem to solve. The method they propose is called the AAR method.[8,9]

3 | AAR METHOD

The new feature characterizing the AAR method consists in computing an Anderson mixing after multiple Richardson steps, so that the cost of solving successive least squares problems is mitigated by several cheap Richardson sweeps in between. A pseudocode is provided in Algorithm 1. We present the formulation of AAR assuming that the Anderson acceleration is implemented based on Equation (6). The iteration of the AAR method has the form of a nonstationary Richardson scheme, as follows:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + C_k \mathbf{r}^k,$$

where, now, matrix C_k becomes

$$C_k = \begin{cases} \omega I, & k/p \notin \mathbb{N} \\ \beta_k I - (X_k + \beta_k R_k)\left(R_k^T R_k\right)^{-1} R_k^T, & k/p \in \mathbb{N}. \end{cases}$$

The matrices X_k and R_k are defined as in (8) and (9), respectively. The parameter p represents the number of Richardson sweeps separating two consecutive Anderson mixing accelerations. The other parameters are ω, $\{\beta_k\}$, and m. The first two are relaxation parameters for the Richardson sweeps, whereas m represents the number of columns in the $n \times m$ least squares problem to compute the Anderson acceleration. It should be noted that the least squares problems used in this scheme are not structured. This is a disadvantage in comparison with other standard linear solvers like GMRES, where the least squares problems are cheaper to solve because the matrices are structured as upper Hessenberg matrices. Letting $p \to \infty$ would reduce AAR to stationary Richardson with ω as the relaxation parameter, whereas $p = 1$ would make it coincide with Anderson–Richardson where $\{\beta_k\}$ would be the sequence of relaxation parameters. We stress the fact that an alternative would be to adopt the definition of Anderson correction as in Equation (11) along with the definition of matrices $\hat{X}_k$ and $\hat{R}_k$ as in (12) and (13). However, this implementation of the Anderson acceleration does not allow us to recast the iteration of AAR so that $\mathbf{x}^{k+1}$ directly relates to $\mathbf{x}^k$ whenever k is a multiple of p.

Algorithm 1 Alternating Anderson–Richardson (AAR)

Data: $\mathbf{x}^0$, ω, $\{\beta_k\}_{k=0}^{k=\text{maxit}}$, p, m, tol
Compute $\mathbf{r}^0 = \mathbf{b} - A\mathbf{x}^0$
Compute $\mathbf{x}^1 = (I - \beta_0 A)\mathbf{x}^0 + \mathbf{b}$
Set $k = 1$
while $\frac{\|\mathbf{r}^{k-1}\|_2}{\|\mathbf{b}\|_2} > tol$ **do**
  Compute $\ell = \min\{k, m\}$
  Compute $\mathbf{r}^k = \mathbf{b} - A\mathbf{x}^k$
  Set $X_k = [(\mathbf{x}^{k-\ell+1} - \mathbf{x}^{k-\ell}), \dots, (\mathbf{x}^k - \mathbf{x}^{k-1})] \in \mathbb{R}^{n\times\ell}$
  Set $R_k = [(\mathbf{r}^{k-\ell+1} - \mathbf{r}^{k-\ell}), \dots, (\mathbf{r}^k - \mathbf{r}^{k-1})] \in \mathbb{R}^{n\times\ell}$
  if $k \pmod p \neq 0$ **then**
    Set $\mathbf{x}^{k+1} = \mathbf{x}^k + \omega\mathbf{r}^k$
  else
    Determine $\mathbf{g}^k = [g_1^k, \dots, g_\ell^k]^T$ such that $\mathbf{g}^k = \underset{\mathbf{g}\in\mathbb{R}^\ell}{\text{argmin}}\|\mathbf{r}^k - R_k\mathbf{g}\|_2$
    Set $\overline{\mathbf{x}}^k = \mathbf{x}^k - X_k\mathbf{g}^k$
    Set $\mathbf{x}^{k+1} = \overline{\mathbf{x}}^k + \beta_k(\mathbf{b} - A\overline{\mathbf{x}}^k)$
  end if
  $k = k + 1$
end while
return $\mathbf{x}^{k+1}$

3.1 | Comparison between GMRES and AAR

Similarly to other works[15,17] on AR, it is possible to identify a connection between AAR and GMRES.[1] Let us assume again that $m = k$ at first. Hence, the projection subspace for the Anderson mixing is constructed by using all the previous approximations. From now on, we refer to AAR with $m = k$ as full AAR.

For the purpose of comparing the way full AAR and full GMRES work, we need to introduce some quantities that are employed to describe the behavior of the methods. Following the work of Potra and Engler,[15] we introduce the *stagnation index* as

$$\eta_G = \min\left\{\ell \in \mathbb{N} : \ \mathbf{x}_G^\ell = \mathbf{x}_G^{\ell+1}\right\},$$

where $\mathbf{x}_G^\ell$ is the approximate solution computed by full GMRES at the ℓth iteration. In the work of Wilkinson,[18] the *grade* of $\mathbf{r}^0 \neq 0$ with respect to A is defined as

$$\nu(A, \mathbf{r}^0) = \max\left\{\ell \in \mathbb{N} : \ \dim\left(\mathcal{K}_\ell(A, \mathbf{r}^0)\right) = \ell\right\},$$

with $\mathcal{K}_p(A, \mathbf{r}^0)$ being the p-dimensional Krylov subspace defined on matrix A and vector $\mathbf{r}^0$. Alternatively, the grade of $\mathbf{r}^0 \neq 0$ with respect to A can also be equivalently defined as the smallest integer ν for which there is a nonzero monic polynomial q of degree ν such that

$$q(A)\mathbf{r}^0 = 0,$$

that is,

$$\nu(A, \mathbf{r}^0) = \min\{\ell \in \mathbb{N} : \ \mathbf{r}^0, A\mathbf{r}^0, \dots, A^\ell\mathbf{r}^0 \text{ are linearly dependent}\}.$$

The upcoming discussion considers Anderson mixing as defined in Equation (11) for clarity of exposition. However, the same conclusions hold also for Anderson mixing as defined in Equation (6). In fact, the study of full AAR is mathematically equivalent with respect to the specific choice of the Anderson mixing.

Theorem 1. *Consider full AAR with a periodic updating interval equal to p between two consecutive Anderson mixing steps. Denote* $\overline{\mathbf{x}}_{\text{AAR}}^{\ell p}$ *as the approximate solution to (1) provided by full AAR after the Anderson mixing at the* (ℓp)*th*

iteration, and denote $\mathbf{x}_G^{\ell p}$ *as the solution computed via full GMRES at the* (ℓp)*th iteration. Refer to* $\nu(A, \mathbf{r}^0)$ *as the grade of* $\mathbf{r}^0$ *with respect to* A *and* η_G *as the stagnation index of full GMRES. Then, the following relation holds in exact arithmetic:*

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{\ell p} = \mathbf{x}_G^{\ell p}, \quad \ell = 0, 1, \ldots \quad \text{s.t.} \quad \ell p \leq \eta_G \quad \text{or} \quad \text{s.t.} \quad \ell p \leq \nu(A, \mathbf{r}^0). \tag{14}$$

Moreover, if the stagnation does not occur, then

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{tp} = \mathbf{x}_G^{\nu(A,\mathbf{r}^0)} = \mathbf{x}, \quad t \in \mathbb{N}, \quad \text{s.t.} \quad (t-1)p < \nu(A, \mathbf{r}^0) \leq tp. \tag{15}$$

Proof. Without loss of generality, we set $\omega = 1$ and $\beta_k = 1, \forall k \geq 0$. Moreover, we remind the reader of the following relation:

$$\eta_G < \nu(A, \mathbf{r}^0).$$

Indeed, if the iteration index hit $\nu(A, \mathbf{r}^0)$, then full GMRES would converge to the exact solution without stagnating. We start considering the first p Richardson sweeps, and we express the approximations to the solution of (1) in terms of powers of A that multiply the initial residual $\mathbf{r}^0 = \mathbf{b} - A\mathbf{x}^0$. Our initial claim is that

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{p} = \mathbf{x}_G^{p}. \tag{16}$$

The proof of this initial claim proceeds in an inductive fashion. The first Richardson iteration yields

$$\mathbf{x}_{\mathrm{AAR}}^{1} = \mathbf{x}^0 + \mathbf{r}^0,$$

which proves the base case. For the induction step, we assume that

$$\mathbf{x}_{\mathrm{AAR}}^{k} = \mathbf{x}^0 + \sum_{i=0}^{k-1} c_i A^i \mathbf{r}^0, \quad k \leq p-1.$$

The successive Richardson step generates a new approximation

$$\begin{aligned}
\mathbf{x}_{\mathrm{AAR}}^{k+1} &= \mathbf{x}_{\mathrm{AAR}}^{k} + \mathbf{r}^k \\
&= \mathbf{x}^0 + \sum_{i=0}^{k-1} c_i A^i \mathbf{r}^0 + \mathbf{b} - A\left(\mathbf{x}^0 + \sum_{i=0}^{k-1} c_i A^i \mathbf{r}^0\right) \\
&= \mathbf{x}^0 + \mathbf{r}^0 + \left(\sum_{i=0}^{k-1} c_i A^i \mathbf{r}^0\right) - A\left(\sum_{i=0}^{k-1} c_i A^i \mathbf{r}^0\right) \\
&= \mathbf{x}^0 + \mathbf{r}^0 + (I - A)\left(\sum_{i=0}^{k-1} c_i A^i \mathbf{r}^0\right).
\end{aligned}$$

This leads to

$$\mathbf{x}_{\mathrm{AAR}}^{k+1} = \mathbf{x}^0 + \mathbf{z}^{k+1}, \quad k \leq p-1,$$

where $\mathbf{z}^{k+1} \in \mathcal{K}_{k+1}(A, \mathbf{r}^0)$. After the first p Richardson sweeps, the algorithm performs an Anderson mixing as defined in Equation (11), yielding

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{p} = \mathbf{x}^0 - \sum_{i=1}^{p} \hat{g}_i^p \left(\mathbf{x}_{\mathrm{AAR}}^{i} - \mathbf{x}^0\right),$$

where

$$\hat{\mathbf{g}}^p = \left(\hat{g}_1^p, \ldots, \hat{g}_p^p\right)^T = \underset{\mathbf{g}\in\mathbb{R}^p}{\operatorname{argmin}} \left\| \mathbf{b} - A\left[\mathbf{x}^0 - \sum_{i=1}^{p} g_i \left(\mathbf{x}_{\mathrm{AAR}}^{i} - \mathbf{x}^0\right)\right] \right\|_2.$$

The columns of the matrix $\hat{X}_k$ (defined in Equation (12)) are expressed in terms of increasing powers of A. Since we are assuming that $\ell p < \nu(A, \mathbf{r}^0)$, this guarantees that matrices $\hat{X}_k$ and $\hat{R}_k$ (defined in Equation (13)) have full column rank and

$$\mathcal{R}(\hat{X}_k) = \mathcal{K}_k(A, \mathbf{r}^0), \quad k \leq p.$$

Since the least squares problem minimizes the residual on the pth Krylov subspace $\mathcal{K}_p(A, \mathbf{r}^0)$, we have proved that

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{p} = \mathbf{x}_G^{p}.$$

This completes the induction proof for the first p iterations of AAR. Repeating the same induction procedure for successive Richardson and Anderson mixing steps, one can prove the original statement in formula (14). In fact, let us assume that the Anderson mixing at the $(\ell - 1)p$th iteration computes

$$\overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} = \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p}, \quad \overline{\mathbf{z}}^{(\ell-1)p} \in \mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0)$$

and that $\hat{X}_{(\ell-1)p}$ and $\hat{R}_{(\ell-1)p}$ have full column rank. Then, we have

$$\mathcal{R}\left(\hat{X}_{(\ell-1)p}\right) = \mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0).$$

The approximation $\mathbf{x}_{\text{AAR}}^{(\ell-1)p+1}$ is computed via a standard Richardson sweep, that is,

$$\begin{aligned}\mathbf{x}_{\text{AAR}}^{(\ell-1)p+1} &= \overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} + \mathbf{b} - A\overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} \\ &= \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p} + \mathbf{b} - A\left(\mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p}\right) \\ &= \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p} + \mathbf{r}^0 - A\overline{\mathbf{z}}^{(\ell-1)p}.\end{aligned}$$

Therefore, we obtain

$$\mathbf{x}_{\text{AAR}}^{(\ell-1)p+1} \in \mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0) + A\mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0) = \mathcal{K}_{(\ell-1)p+1}(A, \mathbf{r}^0).$$

In other words, each Richardson sweep expands the Krylov subspace from which the approximation is computed, leading eventually to

$$\mathbf{x}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 + \mathbf{z}^{\ell p}, \quad \mathbf{z}^{\ell p} \in \mathcal{K}_{\ell p}(A, \mathbf{r}^0).$$

The next step requires an Anderson mixing so that

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 - \sum_{i=1}^{\ell p} \hat{g}_i^{\ell p} \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right),$$

where

$$\hat{\mathbf{g}}^{\ell p} = \left(\hat{g}_1^{\ell p}, \ldots, \hat{g}_{\ell p}^{\ell p}\right)^T = \underset{\mathbf{g}\in\mathbb{R}^{\ell p}}{\operatorname{argmin}} \left\| \mathbf{b} - A\left[\mathbf{x}^0 - \sum_{i=1}^{\ell p} g_i \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right)\right]\right\|_2.$$

Since $\ell p \leq \nu(A, \mathbf{r}^0)$, this still guarantees that matrices $\hat{X}_k$ and $\hat{R}_k$ have full column rank and

$$\mathcal{R}(\hat{X}_k) = \mathcal{K}_k(A, \mathbf{r}^0), \quad k \leq \ell p.$$

Since the residual is minimized on the Krylov subspace $\mathcal{K}_{\ell p}(A, \mathbf{r}^0)$, we have that

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}_G^{\ell p}.$$

Now, we analyze what would happen if the stagnation did not occur. If this were the case, then the index $\nu(A, \mathbf{r}^0)$ would be associated either with a simple Richardson's step or with an iteration where both a Richardson's step and an Anderson mixing are performed. However, the Richardson's step after the iteration with index $\nu(A, \mathbf{r}^0)$ would no longer expand the dimension of the column space of $\hat{X}_k$, which means that

$$\mathcal{R}(\hat{X}_k) = \mathcal{K}_{\nu(A,\mathbf{r}^0)}(A, \mathbf{r}^0), \quad k \geq \nu(A, \mathbf{r}^0). \tag{17}$$

Denoting by $tp \geq \nu(A, \mathbf{r}^0)$ the first iteration performing an Anderson mixing following the iteration with index equal to $\nu(A, \mathbf{r}^0)$, we have

$$\overline{\mathbf{x}}_{\text{AAR}}^{tp} = \mathbf{x}^0 - \sum_{i=1}^{tp} \hat{g}_i^{tp} \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right),$$

where

$$\hat{\mathbf{g}}^{tp} = \left(\hat{g}_1^{tp}, \ldots, \hat{g}_{tp}^{tp}\right)^T = \underset{\mathbf{g}\in\mathbb{R}^{tp}}{\operatorname{argmin}} \left\| \mathbf{b} - A\left[\mathbf{x}^0 - \sum_{i=1}^{tp} g_i \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right)\right]\right\|_2.$$

Because of the relation in (17), the least squares problem minimizes the residual on the Krylov subspace $\mathcal{K}_{\nu(A,\mathbf{r}^0)}(A, \mathbf{r}^0)$. Therefore, we have

$$\overline{\mathbf{x}}_{\text{AAR}}^{tp} = \mathbf{x}_G^{\nu(A,\mathbf{r}^0)} = \mathbf{x},$$

which means that both full GMRES and full AAR converge to the exact solution of the linear system.

Note that the statement of the theorem and the proof are valid for any nonzero value of ω and β_k's. □

Remark 2. Theorem 1 guarantees periodic equivalence (the equivalence occurs only after every Anderson mixing) between full AAR and full GMRES if there is no stagnation in the history of approximations computed by full GMRES. This means that in the absence of stagnations, full AAR is periodically an oblique projection method (for more details, see p. 129 in the work of Saad[1]). Therefore, at the iteration $k = \ell p \leq \eta_G$, we obtain

$$\mathcal{V}_{\ell p} = \mathcal{K}_{\ell p}(A, \mathbf{r}^0) = \mathcal{R}(X_{\ell p}) = \mathcal{R}(\hat{X}_{\ell p}), \quad \dim(\mathcal{V}_{\ell p}) = m = \ell p$$

and

$$\mathcal{W}_{\ell p} = A\mathcal{K}_{\ell p}(A, \mathbf{r}^0) = A\mathcal{R}(X_{\ell p}) = A\mathcal{R}(\hat{X}_{\ell p}) = \mathcal{R}(R_{\ell p}) = \mathcal{R}(\hat{R}_{\ell p}),$$

with $\dim(\mathcal{W}_{\ell p}) = m = \ell p$.

Let us denote by full AR the version of Anderson–Richardson, which uses every previous approximation for an Anderson mixing at each iteration. If full GMRES stagnates, it is known that this breaks the equivalence between full GMRES and full AR, since the latter would never recover from the stagnation (see the work of Potra and Engler[15] for more details). As concerns full AAR, the situation is more complex. Next, we are going to show that full AAR is more robust to stagnations than full AR, which means that under certain circumstances, the stagnation does not prevent full AAR from converging to the solution of (1).

Theorem 2. *Consider AAR with a periodic updating interval between two consecutive Anderson mixing steps of length p. For an integer s, denote by $\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp}$ the approximate solution to (1) obtained by full AAR after the Anderson mixing at the (sp)th iteration, and denote by $\mathbf{x}_G^{sp}$ the solution computed via full GMRES at the (sp)th iteration. One of the following three cases holds:*

1. *if* $(\ell - 1)p = \eta_G$ *and* $\mathbf{x}_G^{(\ell-1)p} \neq \mathbf{x}_G^{\ell p}$, *then*

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \mathbf{x}_G^{sp}, \quad \textit{when} \quad s \geq \ell \quad \textit{and} \quad sp < \nu(A, \mathbf{r}^0),$$

   or

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \mathbf{x}_G^{\nu(A,\mathbf{r}^0)} = \mathbf{x}, \quad \textit{when} \quad s \geq \ell \quad \textit{and} \quad sp \geq \nu(A, \mathbf{r}^0);$$

2. *if* $(\ell - 1)p < \eta_G < \ell p$ *and* $\mathbf{x}_G^{\ell p} \neq \mathbf{x}_G^{(\ell+1)p}$, *then*

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \mathbf{x}_G^{sp}, \quad \textit{when} \quad s \geq \ell \quad \textit{and} \quad sp < \nu(A, \mathbf{r}^0),$$

   or

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \mathbf{x}_G^{\nu(A,\mathbf{r}^0)} = \mathbf{x}, \quad \textit{when} \quad s \geq \ell \quad \textit{and} \quad sp \geq \nu(A, \mathbf{r}^0);$$

3. *if* $\eta_G = (\ell - 1)p$ *and* $\mathbf{x}_G^{(\ell-1)p} = \mathbf{x}_G^{\ell p}$, *then*

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \mathbf{x}_G^{(\ell-1)p}, \quad \textit{when} \quad s \geq (\ell - 1).$$

Proof. The three items in the statement of the theorem deal with three possible scenarios of a stagnation in full GMRES. Theorem 1 has already shown that in the absence of stagnation, full AAR and full GMRES provide the same solution at the end of each periodic interval. The equivalence between full GMRES and full AAR holds after the stagnation only if the number of consecutive Richardson sweeps in full AAR exceeds the extension of stagnation in full GMRES. If full GMRES stagnates for a number of iterations at least equal to the number of consecutive Richardson sweeps in full AAR, then the equivalence between the algorithms is lost.

1. Let us assume at first that no stagnation occurs before the iteration with index $k = (\ell - 1)p$. By Theorem 1, we know that every time an Anderson acceleration is computed by full AAR, the following holds:

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \mathbf{x}_G^{sp}, \quad sp \leq \eta_G.$$

   If full GMRES stagnates between the iterations with index $k = (\ell - 1)p$ and the successive one, then

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{(\ell-1)p} = \mathbf{x}_G^{(\ell-1)p}$$

   and

$$\mathbf{x}_G^{(\ell-1)p} = \mathbf{x}_G^{(\ell-1)p+1}.$$

   Let us assume that the stagnation of full GMRES lasts only for the successive q iterations, where $q < p$. Therefore, we have

$$\mathbf{x}_G^{(\ell-1)p} = \mathbf{x}_G^{(\ell-1)p+1} = \cdots = \mathbf{x}_G^{(\ell-1)p+q} \neq \cdots \neq \mathbf{x}_G^{\ell p}.$$

Full AAR performs standard Richardson sweeps at the iterations indexed by

$$k = (\ell - 1)p + 1, \ \dots, \ell p.$$

In fact, the approximation $\mathbf{x}_{\text{AAR}}^{(\ell-1)p+1}$ is

$$\begin{aligned}\mathbf{x}_{\text{AAR}}^{(\ell-1)p+1} &= \overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} + \mathbf{b} - A\overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} \\ &= \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p} + \mathbf{b} - A\left(\mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p}\right) \\ &= \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p} + \mathbf{r}^0 - A\overline{\mathbf{z}}^{(\ell-1)p},\end{aligned}$$

implying that

$$\mathbf{x}_{\text{AAR}}^{(\ell-1)p+1} \in \mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0) + A\mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0) = \mathcal{K}_{(\ell-1)p+1}(A, \mathbf{r}^0).$$

In other words, each Richardson sweep expands the Krylov subspace from which the approximation is computed, leading eventually to

$$\mathbf{x}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 + \mathbf{z}^{\ell p}, \quad \mathbf{z}^{\ell p} \in \mathcal{K}_{\ell p}(A, \mathbf{r}^0).$$

The next step requires an Anderson mixing so that

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 - \sum_{i=1}^{\ell p} \hat{g}_i^{\ell p} \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right),$$

where

$$\hat{\mathbf{g}}^{\ell p} = \left(\hat{g}_1^{\ell p}, \ \dots, \hat{g}_{\ell p}^{\ell p}\right)^T = \underset{\mathbf{g}\in\mathbb{R}^{\ell p}}{\text{argmin}} \left\| \mathbf{b} - A\left[\mathbf{x}^0 - \sum_{i=1}^{\ell p} g_i \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right)\right]\right\|_2.$$

If $\ell p < \nu(A, \mathbf{r}^0)$, then $\hat{X}_{\ell p}$ has full column rank, and in particular,

$$\mathcal{R}(\hat{X}_k) = \mathcal{K}_k(A, \mathbf{r}^0), \quad k \leq \ell p.$$

Since the least squares problem minimizes the residual on the Krylov subspace $\mathcal{K}_{\ell p}(A, \mathbf{r}^0)$, we have that

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}_G^{\ell p}.$$

If $\ell p \geq \nu(A, \mathbf{r}^0)$, then

$$\mathcal{R}(\hat{X}_{\ell p}) = \mathcal{K}_{\nu(A,\mathbf{r}^0)}(A, \mathbf{r}^0).$$

This means that the least squares problem solved at iteration ℓp minimizes the residual on the Krylov subspace $\mathcal{K}_{\nu(A,\mathbf{r}^0)}(A, \mathbf{r}^0)$, leading to

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}_G^{\nu(A,\mathbf{r}^0)} = \mathbf{x}.$$

In the situation where $\ell p < \nu(A, \mathbf{r}^0)$, the performance of full AAR after the (ℓp)th iteration is equivalent to the performance of full AAR using directly $\overline{\mathbf{x}}_{\text{AAR}}^{\ell p}$ as an initial guess. Therefore, from the iteration index $k = \ell p$ on, Theorem 1 can be applied again using $\overline{\mathbf{x}}_{\text{AAR}}^{\ell p}$ as an initial guess, eventually proving the conclusion for Case 1. This means that the type of stagnation considered in Case 1 does not impact the performance of full AAR.

2. Let us assume now that the stagnation happens in the history of full GMRES at an iteration with index $k = (\ell - 1)p + q$, where $q < p$. Full AAR performs Richardson steps from iterations $k = (\ell - 1)p + q$ through $k = \ell p$, leading thus to

$$\mathbf{x}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 + \mathbf{z}^{\ell p},$$

where $\mathbf{z}^{\ell p} \in \mathcal{K}_{\ell p}(A, \mathbf{r}^0)$. If $\ell p < \nu(A, \mathbf{r}^0)$, then $\hat{X}_{\ell p}$ has full column rank, and in particular,

$$\mathcal{R}(\hat{X}_k) = \mathcal{K}_k(A, \mathbf{r}^0), \quad k \leq \ell p.$$

Since the least squares problem at the (ℓp)th iteration minimizes the residual on the Krylov subspace $\mathcal{K}_{\ell p}(A, \mathbf{r}^0)$, we have that

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}_G^{\ell p}.$$

If $\ell p \geq \nu(A, \mathbf{r}^0)$, then

$$\mathcal{R}(\hat{X}_{\ell p}) = \mathcal{K}_{\nu(A,\mathbf{r}^0)}(A, \mathbf{r}^0).$$

This means that the least squares problem solved at iteration ℓp minimizes the residual on the Krylov subspace $\mathcal{K}_{\nu(A,\mathbf{r}^0)}(A, \mathbf{r}^0)$, leading to

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}_G^{\nu(A,\mathbf{r}^0)} = \mathbf{x}.$$

If $\ell p < \nu(A, \mathbf{r}^0)$, the situation is similar to Case 1 from iteration ℓp on. The only difference is that the iteration index is shifted forward by p. In fact, here, we have a stagnation at iteration ℓp, and we are assuming that $\mathbf{x}_G^{(\ell+1)p} \neq \mathbf{x}_G^{\ell p}$, whereas in Case 1, we were assuming that we had a stagnation at the iteration $(\ell - 1)p$ and $\mathbf{x}_G^{\ell p} \neq \mathbf{x}_G^{(\ell-1)p}$. Using the same reasoning as in Case 1 but with the indices shifted forward by p, we can prove the statement for Case 2. Therefore, full AAR converges to the solution of (1) even for the type of stagnation considered in Case 2.

3. We now consider the most challenging case of stagnation for full AAR. This occurs when the stagnation in the history of full GMRES lasts for a number of iterations equal to $q \geq p$. For ease of exposition, we are going to adopt the definition of Anderson mixing as described in Equation (11). Since full AAR involves all the previous updates in the Anderson mixing, we know that formulations (6) and (11) are equivalent.

   If there is no stagnation up to the iteration $k = (\ell - 1)p$, we have that

$$\overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} = \mathbf{x}_G^{(\ell-1)p}.$$

Therefore, from the definition of Anderson mixing in Equation (11), it follows that

$$\overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} = \mathbf{x}^0 - \sum_{i=1}^{(\ell-1)p} \hat{g}_i^{(\ell-1)p} \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right) = \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p},$$

where the vector

$$\overline{\mathbf{z}}^{(\ell-1)p} = -\sum_{i=1}^{(\ell-1)p} \hat{g}_i^{(\ell-1)p} \left(\mathbf{x}_{\text{AAR}}^i - \mathbf{x}^0\right) \in \mathcal{K}_{(\ell-1)p}(A, \mathbf{r}^0)$$

is computed so as to minimize the ℓ^2-norm of the residual. After computing the solution to the least squares problem, full AAR performs Richardson steps from iterations $k = (\ell - 1)p + 1$ through $k = \ell p$. We thus obtain

$$\mathbf{x}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 + \mathbf{z}^{\ell p},$$

where $\mathbf{z}^{\ell p} \in \mathcal{K}_{\ell p}(A, \mathbf{r}^0)$. Under the hypotheses that $\mathbf{x}_G^{(\ell-1)p} = \mathbf{x}_G^{\ell p}$, it follows that

$$\ell p < \nu(A, \mathbf{r}^0).$$

Therefore, the matrix $\hat{X}_{\ell p}$ has full column rank, and in particular,

$$\mathcal{R}(\hat{X}_k) = \mathcal{K}_k(A, \mathbf{r}^0), \quad k \leq \ell p.$$

Therefore, the least squares problem at iteration ℓp computes $\overline{\mathbf{z}}^{\ell p} \in \mathcal{K}_{\ell p}(A, \mathbf{r}^0)$ such that

$$\overline{\mathbf{z}}^{\ell p} = \underset{\mathbf{z} \in \mathcal{K}_{\ell p}(A, \mathbf{r}^0)}{\operatorname{argmin}} \|\mathbf{b} - A(\mathbf{x}^0 + \mathbf{z})\|_2.$$

However, we are assuming that full GMRES is still stagnating at the iteration $k = \ell p$. Therefore, we have

$$\overline{\mathbf{x}}_{\text{AAR}}^{\ell p} = \mathbf{x}^0 + \overline{\mathbf{z}}^{\ell p} = \mathbf{x}_G^{\ell p} = \mathbf{x}_G^{(\ell-1)p} = \overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p}$$

and

$$\begin{aligned}
\mathbf{x}_{\text{AAR}}^{\ell p+1} &= \overline{\mathbf{x}}_{\text{AAR}}^{\ell p} + \beta_{\ell p+1} \left(\mathbf{b} - A\overline{\mathbf{x}}_{\text{AAR}}^{\ell p}\right) \\
&= \overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} + \beta_{\ell p+1} \left(\mathbf{b} - A\overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p}\right) \\
&= \overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} + \beta_{\ell p+1} \left(\mathbf{b} - A\left(\mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p}\right)\right) \\
&= \overline{\mathbf{x}}_{\text{AAR}}^{(\ell-1)p} + \beta_{\ell p+1}\mathbf{r}^0 - \beta_{\ell p+1}A\overline{\mathbf{z}}^{(\ell-1)p} \\
&= \mathbf{x}^0 + \overline{\mathbf{z}}^{(\ell-1)p} + \beta_{\ell p+1}\mathbf{r}^0 - \beta_{\ell p+1}A\overline{\mathbf{z}}^{(\ell-1)p} \\
&= \mathbf{x}^0 + \beta_{\ell p+1}\mathbf{r}^0 + (I - \beta_{\ell p+1}A)\overline{\mathbf{z}}^{(\ell-1)p}.
\end{aligned} \tag{18}$$

This means that

$$\mathbf{x}_{\text{AAR}}^{\ell p+1} - \mathbf{x}^0 \in \mathcal{K}_{(\ell-1)p+1}(A, \mathbf{r}^0).$$

At each iteration with index

$$k = \ell p + j, \quad j = 2, \ldots, p,$$

a Richardson sweep is computed so that

$$\mathbf{x}_{\text{AAR}}^{\ell p+j} = \mathbf{x}_{\text{AAR}}^{\ell p+j-1} + \omega \left(\mathbf{b} - A\mathbf{x}_{\text{AAR}}^{\ell p+j-1}\right),$$

which leads to

$$\mathbf{x}_{\mathrm{AAR}}^{\ell p+j} - \mathbf{x}^0 \in \mathcal{K}_{(\ell-1)p+j}, \quad 1 \le j \le p.$$

Given the subspace

$$\mathcal{L}_k = \mathrm{span}\left\{\left(\mathbf{x}_{\mathrm{AAR}}^1 - \mathbf{x}^0\right), \left(\mathbf{x}_{\mathrm{AAR}}^2 - \mathbf{x}^0\right), \ldots, \left(\mathbf{x}_{\mathrm{AAR}}^k - \mathbf{x}^0\right)\right\},$$

we have that

$$\mathcal{L}_{\ell p+j} = \mathcal{K}_{\ell p}(A, \mathbf{r}^0), \qquad 1 \le j \le p.$$

Because of this, the Anderson mixing at $k = (\ell + 1)p$ computes an approximation $\overline{\mathbf{x}}_{\mathrm{AAR}}^{(\ell+1)p}$ that minimizes the residual onto $\mathcal{L}_{(\ell+1)p} = \mathcal{K}_{\ell p}(A, \mathbf{r}^0)$. Therefore, we have

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{(\ell+1)p} = \overline{\mathbf{x}}_{\mathrm{AAR}}^{\ell p} = \overline{\mathbf{x}}_{\mathrm{AAR}}^{(\ell-1)p}.$$

By repeating the process for a generic periodic interval like the one described by

$$k = sp + 1, \ldots, (s+1)p,$$

we obtain

$$\mathbf{x}_{\mathrm{AAR}}^{sp+j} - \mathbf{x}^0 \in \mathcal{K}_{(\ell-1)p+j}(A, \mathbf{r}^0), \quad s \ge \ell - 1, \quad 1 \le j \le p.$$

Therefore, full AAR stagnates from the iteration $k = \ell p$ on, without ever recovering because

$$\mathcal{L}_k = \mathcal{L}_{\ell p} \ne \mathcal{K}_{\ell p+1}(A, \mathbf{r}^0), \quad k \ge \ell p.$$

Hence, we obtain

$$\overline{\mathbf{x}}_{\mathrm{AAR}}^{sp} = \overline{\mathbf{x}}_{\mathrm{AAR}}^{(\ell-1)p} = \mathbf{x}_G^{(\ell-1)p}, \quad s \ge \ell - 1,$$

whereas full GMRES keeps on increasing the Krylov subspace $\mathcal{K}_k(A, \mathbf{r}^0)$ used to compute $\mathbf{x}_G^k$, and it eventually converges to the solution of (1).

□

Remark 3. Theorem 2 shows that the convergence of full AAR is not affected by a stagnation that occurs in the equivalent full GMRES as long as the stagnation lasts less than p iterations. In terms of robustness against stagnation, this represents an improvement over full AR. In fact, a theoretical discussion in the work of Potra and Engler[15] shows that full AR cannot recover from any stagnations. This happens because full AR solves a least squares problem to minimize the residual at each iteration, whereas full AAR solves a least squares problem only at periodic intervals of length p. This suggests that there are two advantages in solving the least squares problem less frequently through the insertion of multiple relaxation steps in between. On the one hand, it alleviates the computational burden because there is no need to solve a least squares problem at each iteration but only after a batch of p Richardson sweeps, which reduces computational complexity and increases locality. On the other hand, it also enhances the robustness against stagnations by making their occurrence less likely. This property of full AAR is likely to be inherited by truncated or restarted versions of AAR as well.

Although full AAR and full GMRES are mathematically equivalent at the end of each periodic interval of length p, the bases constructed by the two algorithms for the projection steps are not the same. Only the first vector is the same up to a scaling factor. In fact, the first vector of the basis built by full GMRES is

$$\mathbf{v}_1 = \frac{\mathbf{r}^0}{\|\mathbf{r}^0\|_2},$$

and the first vector of the basis built by full AAR is

$$\mathbf{x}_{\mathrm{AAR}}^1 - \mathbf{x}^0 = \omega \mathbf{r}^0.$$

We aim now to compare the second vector constructed by each algorithm. We recall that the non-normalized second vector of the full GMRES basis is

$$\mathbf{w}_1 = \left[I - \frac{\mathbf{r}^0}{\|\mathbf{r}^0\|_2}\left(\frac{\mathbf{r}^0}{\|\mathbf{r}^0\|_2}\right)^T\right]\frac{A\mathbf{r}^0}{\|\mathbf{r}^0\|_2}.$$

As concerns full AAR, we need to carry out two different computations, depending on which implementation of Anderson mixing is adopted. The approximation computed by full AAR at the second iteration is

$$\begin{aligned}\mathbf{x}_{\mathrm{AAR}}^2 &= \mathbf{x}_{\mathrm{AAR}}^1 + \omega(\mathbf{b} - A\mathbf{x}^1)\\ &= \mathbf{x}^0 + \omega\mathbf{r}^0 + \omega\left(\mathbf{b} - A(\mathbf{x}^0 + \omega\mathbf{r}^0)\right)\\ &= \mathbf{x}^0 + 2\omega\mathbf{r}^0 - \omega^2 A\mathbf{r}^0.\end{aligned}$$

If the Anderson mixing were defined as in Equation (6), then the second vector of the basis built by full AAR would be

$$\mathbf{x}^2_{\mathrm{AAR}} - \mathbf{x}^1_{\mathrm{AAR}} = \omega(I - \omega A)\mathbf{r}^0.$$

If the Anderson mixing were defined as in Equation (11) instead, this would lead to an expansion of the basis through the vector

$$\mathbf{x}^2_{\mathrm{AAR}} - \mathbf{x}^0 = \omega(2I - \omega A)\mathbf{r}^0.$$

It is possible to see that $\mathbf{w}_1, \mathbf{x}^2_{\mathrm{AAR}} - \mathbf{x}^1_{\mathrm{AAR}}$ and $\mathbf{x}^2_{\mathrm{AAR}} - \mathbf{x}^0$ identify different directions. In general, this difference propagates also to the other vectors used to expand the basis, leading to

$$\begin{aligned} \mathrm{span}\{\mathbf{v}_k\} &\neq \mathrm{span}\left\{\mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^{k-1}_{\mathrm{AAR}}\right\}, \\ \mathrm{span}\{\mathbf{v}_k\} &\neq \mathrm{span}\left\{\mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^0\right\}, \\ \mathrm{span}\left\{\mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^{k-1}_{\mathrm{AAR}}\right\} &\neq \mathrm{span}\left\{\mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^0\right\}, \quad k \geq 2. \end{aligned}$$

Therefore, in general, we have

$$\begin{aligned} \mathrm{span}\{\mathbf{v}_\ell, \ldots, \mathbf{v}_k\} &\neq \mathrm{span}\left\{\mathbf{x}^\ell_{\mathrm{AAR}} - \mathbf{x}^{\ell-1}_{\mathrm{AAR}}, \ldots, \mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^{k-1}_{\mathrm{AAR}}\right\}, \\ \mathrm{span}\{\mathbf{v}_\ell, \ldots, \mathbf{v}_k\} &\neq \mathrm{span}\left\{\mathbf{x}^\ell_{\mathrm{AAR}} - \mathbf{x}^0, \ldots, \mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^0\right\}, \\ \mathrm{span}\{\mathbf{x}^\ell_{\mathrm{AAR}} - \mathbf{x}^{\ell-1}_{\mathrm{AAR}}, \ldots, \mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^{k-1}_{\mathrm{AAR}}\} &\neq \mathrm{span}\left\{\mathbf{x}^\ell_{\mathrm{AAR}} - \mathbf{x}^0, \ldots, \mathbf{x}^k_{\mathrm{AAR}} - \mathbf{x}^0\right\}, \end{aligned} \tag{19}$$

with $1 < \ell \leq k$.

From a practical viewpoint, a fixed value of m across the iterations is recommended, in order to alleviate the computational cost of each least squares problem. In case of a truncation of the basis, AAR leads to a variant that resembles truncated GMRES. We refer to this variant as truncated AAR. However, the inequalities in (19) show that, in general, truncated AAR and truncated GMRES are not equivalent in exact arithmetic, since the subspaces adopted for the projection differ.

Remark 4. If R_ℓ is not a full-rank matrix, then $\mathcal{R}(R_\ell)$ is A-invariant. However, using R_ℓ to compute an Anderson mixing as in Equation (6) is still legitimate by solving least squares problems. Indeed, albeit some numerical examples presented in Section 4 generated a matrix R_ℓ with linearly dependent columns at some iterations, this did not hinder convergence.

3.2 | Convergence analysis for AAR

Next, we prove a convergence result for AAR analogous to the one found theorem 2.1 in the work of Toth and Kelley[16] for AR.

Theorem 3. *Consider a nonsingular linear system as in Equation (1). If the iteration matrix $H = I - A$ is such that $\|H\|_2 = c < 1$, then AAR converges to the solution of (1) in exact arithmetic for any choice of p and m, regardless of the initial guess. The residual norm converges to zero q-linearly, and the q-factor is c.*

Proof. Denoting by $\mathbf{x}^k_{\mathrm{AAR}}$ the Richardson approximation before the Anderson acceleration, the associated residual is

$$\mathbf{r}^k_{\mathrm{AAR}} = H\mathbf{r}^{k-1}_{\mathrm{AAR}};$$

hence,

$$\left\|\mathbf{r}^k_{\mathrm{AAR}}\right\|_2 \leq \|H\|_2 \left\|\mathbf{r}^{k-1}_{\mathrm{AAR}}\right\|_2 < c\left\|\mathbf{r}^{k-1}_{\mathrm{AAR}}\right\|_2.$$

The Anderson mixing is computed to minimize the residual. By recalling the definition of Anderson mixing as in Equation (6) or (11) to compute $\bar{\mathbf{x}}^k_{\mathrm{AAR}}$, we have $\bar{\mathbf{r}}^k_{\mathrm{AAR}} = \mathbf{b} - A\bar{\mathbf{x}}^k_{\mathrm{AAR}}$, and it holds that

$$\left\|\bar{\mathbf{r}}^k_{\mathrm{AAR}}\right\|_2 \leq \left\|\mathbf{r}^k_{\mathrm{AAR}}\right\|_2 < c\left\|\mathbf{r}^{k-1}_{\mathrm{AAR}}\right\|_2,$$

regardless of the choice of m. This proves that the residual ℓ^2-norm converges to zero q-linearly, and the q-factor is c. □

Similarly to what was already discussed in the work of Toth and Kelley[16] for AR, requiring $\|H\|_2 < 1$ for AAR to converge is too restrictive and impractical in many cases. If the condition $\|H\|_2 < 1$ does not hold, truncated AAR cannot be guaranteed to converge in general. This motivates the possible employment of a variant of the algorithm that we call *augmented AAR*.

3.3 | Augmented AAR

In this section, we discuss a way to expand the subspace of projection used by AAR so that convergence results can be obtained for specific classes of matrices. The basic idea behind the augmentation of the subspace that we are going to present is to split a vector $\mathbf{z} \in \mathbb{R}^n$ into two components

$$\mathbf{z} = \mathbf{z}^1 + \mathbf{z}^2, \quad \mathbf{z}^1, \mathbf{z}^2 \in \mathbb{R}^n$$

so that

$$\text{span}\{\mathbf{z}^1, \mathbf{z}^2\} \supseteq \text{span}\{\mathbf{z}\}.$$

The criterion adopted to decompose the vector of interest is guided by the search of an expanded subspace where convergence of truncated AAR can be guaranteed under certain hypotheses. In this section, we consider the algorithm of truncated AAR with $m > p$, and we focus on the implementation of Anderson mixing as in Equation (6). We denote with $k = \ell$ the iteration index where a Richardson sweep is performed to compute $\mathbf{x}^{\ell}_{\text{AAR}}$ followed by an Anderson mixing to compute $\overline{\mathbf{x}}^{\ell}_{\text{AAR}}$. From Equation (6), we obtain

$$\overline{\mathbf{x}}^{\ell}_{\text{AAR}} = \mathbf{x}^{\ell}_{\text{AAR}} - X_{\ell}\mathbf{g}^{\ell}.$$

Then, a successive Richardson step computes

$$\begin{aligned}\mathbf{x}^{\ell+1}_{\text{AAR}} &= \overline{\mathbf{x}}^{\ell}_{\text{AAR}} + \beta_{\ell}\left(\mathbf{b} - A\overline{\mathbf{x}}^{\ell}_{\text{AAR}}\right)\\ &= \mathbf{x}^{\ell}_{\text{AAR}} - X_{\ell}\mathbf{g}^{\ell} + \beta_{\ell}\left(\mathbf{b} - A\overline{\mathbf{x}}^{\ell}_{\text{AAR}}\right)\\ &= \mathbf{x}^{\ell}_{\text{AAR}} - X_{\ell}\mathbf{g}^{\ell} + \beta_{\ell}\overline{\mathbf{r}}^{\ell}_{\text{AAR}},\end{aligned}$$

where $\overline{\mathbf{r}}^{\ell}_{\text{AAR}} = \mathbf{b} - A\overline{\mathbf{x}}^{\ell}_{\text{AAR}}$. Recalling the definition of $X_{\ell+1} \in \mathbb{R}^{n\times m}$ in (8), the vector used in truncated AAR to update $X_{\ell+1}$ is $\mathbf{x}^{\ell+1}_{\text{AAR}} - \mathbf{x}^{\ell}_{\text{AAR}}$ so that

$$X_{\ell+1} = \left[X_{\ell}(:, 2:m), \mathbf{x}^{\ell+1}_{\text{AAR}} - \mathbf{x}^{\ell}_{\text{AAR}}\right] = \left[X_{\ell}(:, 2:m), -X_{\ell}\mathbf{g}^{\ell} + \beta_{\ell}\overline{\mathbf{r}}^{\ell}_{\text{AAR}}\right].$$

A desirable property to produce convergence results would be that the direction identified by the minimal residual $\overline{\mathbf{r}}^{\ell}_{\text{AAR}}$ be included in the subspace of projection. One way to obtain this is to replace the vector

$$-X_{\ell}\mathbf{g}^{\ell} + \beta_{\ell}\overline{\mathbf{r}}^{\ell}_{\text{AAR}}$$

in the last column of $X_{\ell+1}$ with the separate vectors $X_{\ell}\mathbf{g}^{\ell}$ and $\overline{\mathbf{r}}^{\ell}_{\text{AAR}}$. Obviously,

$$\text{span}\left\{X_{\ell}\mathbf{g}^{\ell}, \overline{\mathbf{r}}^{\ell}_{\text{AAR}}\right\} \supseteq \text{span}\left\{-X_{\ell}\mathbf{g}^{\ell} + \beta_{\ell}\overline{\mathbf{r}}^{\ell}_{\text{AAR}}\right\},$$

which guarantees that no information is lost due to the separation of the original vector in two components. This allows us to replace $X_{\ell+1}$ with a new matrix $\tilde{X}_{\ell+1}$ defined as follows:

$$\tilde{X}_{\ell+1} = \begin{cases}\left[X_{\ell}(:, 2:m), X_{\ell}\mathbf{g}^{\ell}, \beta_{\ell}\overline{\mathbf{r}}^{\ell}_{\text{AAR}}\right] \in \mathbb{R}^{n\times(m+1)}, & \text{if} \quad g^{\ell}_1 \neq 0\\ \left[X_{\ell}(:, 2:m), \beta_{\ell}\overline{\mathbf{r}}^{\ell}_{\text{AAR}}\right] \in \mathbb{R}^{n\times m}, & \text{if} \quad g^{\ell}_1 = 0.\end{cases} \tag{20}$$

Note that we use MATLAB notation for columns. The distinction of the two cases in (20) is driven by numerical considerations. In fact, it is important to guarantee that $\tilde{X}_{\ell+1}$ have full column rank. Obviously,

$$X_{\ell}\mathbf{g}^{\ell} \in \mathcal{R}(X_{\ell}).$$

However,

$$X_{\ell}\mathbf{g}^{\ell} \notin \mathcal{R}(X_{\ell}(:, 2:m)), \quad \text{if} \quad g^{\ell}_1 \neq 0.$$

Therefore, the inclusion of the vector $X_{\ell}\mathbf{g}^{\ell}$ as a column of $\tilde{X}_{\ell+1}$ does not preclude the full column rank property if $g^{\ell}_1 \neq 0$. Moreover,

$$\overline{\mathbf{r}}^{\ell}_{\text{AAR}} \notin \mathcal{R}(X_{\ell}(:, 2:m)) + \text{span}\left\{X_{\ell}\mathbf{g}^{\ell}\right\},$$

which always legitimizes the inclusion of the vector $\overline{\mathbf{r}}^{\ell}_{\text{AAR}}$ as a column of $\tilde{X}_{\ell+1}$. Notice also that, from (20), we have

$$\mathcal{R}(\tilde{X}_{\ell+1}) \supseteq \mathcal{R}(X_{\ell+1}).$$

If $\tilde{X}_{\ell+1}$ has $p+1$ columns, one may shrink it to restore the initial dimension of the projected problem. In our numerical examples, we keep $\tilde{X}_{\ell+1}$ with $p+1$ columns if $g_1^\ell \neq 0$. This choice of expanding the number of columns from m to $m+1$ introduces an additional computational cost, due to the increase in the size of the least squares problem to solve. Regardless of the arbitrary way to reduce from $m+1$ to m columns, the only thing that matters is that

$$\bar{\mathbf{r}}_{\mathrm{AAR}}^{\ell} \in \mathcal{R}(\tilde{X}_{\ell+1}).$$

In fact, the direction identified by $\bar{\mathbf{r}}_{\mathrm{AAR}}^{\ell}$ is going to play a central role in the following convergence result of augmented AAR for positive definite matrices. We remind the reader that a real matrix A is *positive definite* if its symmetric part $\frac{1}{2}(A+A^T)$ is symmetric positive definite.

Theorem 4. *Consider a linear system as in Equation (1). If A is positive definite and the parameters of augmented AAR satisfy $m \geq p$, then augmented AAR converges to the solution of (1) in exact arithmetic, regardless of the initial guess.*

Proof. We denote by $k = \ell$ the iteration index where a Richardson sweep is performed to compute $\mathbf{x}_{\mathrm{AAR}}^{\ell}$ followed by an Anderson mixing that produces $\bar{\mathbf{x}}_{\mathrm{AAR}}^{\ell}$. We thus obtain

$$\bar{\mathbf{x}}_{\mathrm{AAR}}^{\ell} = \mathbf{x}_{\mathrm{AAR}}^{\ell} - X_\ell \mathbf{g}^\ell$$

and

$$\bar{\mathbf{r}}_{\mathrm{AAR}}^{\ell} = \mathbf{b} - A\bar{\mathbf{x}}_{\mathrm{AAR}}^{\ell}.$$

The proof mimics the reasoning presented in the work of Saad[1] (see p. 205) to prove the convergence of restarted GMRES(p) on positive definite matrices. In particular, if $m \geq p$, then

$$\bar{\mathbf{r}}_{\mathrm{AAR}}^{\ell} \in \mathcal{R}(\tilde{X}_{\ell+p}).$$

Therefore, $\mathcal{R}(\tilde{X}_{\ell+p})$ includes the direction of the initial residual vector at each cycle. This means that $\mathcal{R}(\tilde{X}_{\ell+p})$ identifies a hyperplane that contains the direction used by the minimal residual method (see p. 140 in the work of Saad[1]) to update the solution. Since the minimal residual method is guaranteed to converge on positive definite matrices, so is augmented AAR, regardless of the values of m and p. □

4 | NUMERICAL EXPERIMENTS

In this section, we present some experiments to illustrate our theoretical analysis and to assess the performance of the schemes described. First we show a test case to illustrate Theorem 2, followed by two different sets of numerical examples: the first set of tests is performed in MATLAB, whereas the second set of examples uses a code written in `C` with a distributed memory parallelization via an `MPI` environment.

4.1 | Numerical experiments in MATLAB

We initially describe an experiment to validate the statement of Theorem 2, which consists of a linear system where the coefficient matrix is block diagonal with b blocks. Each block has increasing size so that the kth block is an $\ell k \times \ell k$ circulant matrix, where ℓ represents an integer. Each circulant block is constructed as a permutation matrix by taking the first column of the identity matrix and placing it as last. The right-hand side is constructed so that, with zero initial guess, full GMRES stagnates for ℓ consecutive iterations. This is obtained by setting to 1 all the entries of the right-hand side whose index coincides with the row index of the matrix where a new circulant block starts. All the other entries of the right-hand side are set to zero. In this situation, full AR does not converge for any choice of $\ell \geq 1$, whereas the behavior of full AAR depends on the value of p, that is, the number of consecutive Richardson's steps. Indeed, full AAR does not converge when $p \leq \ell - 1$, and it converges in a finite number of iterations for any value of $p > \ell - 1$. This behavior has been detected for different values of b (i.e., total number of diagonal blocks). In Figure 1, we present the history of the residual norm for full AR, full AAR($p=2$), and full AAR($p=3$) when $\ell = 3$ and $b = 5$, which leads to a linear system with 45 unknowns. As shown in Figure 1a, the residual never decreases when full AR is used to solve the linear system. A similar trend is detected for full AAR($p=2$), where the residual norm keeps on oscillating between two values. The lowest value of the residual norm occurs when the least squares problem is solved at every other iteration, as shown in

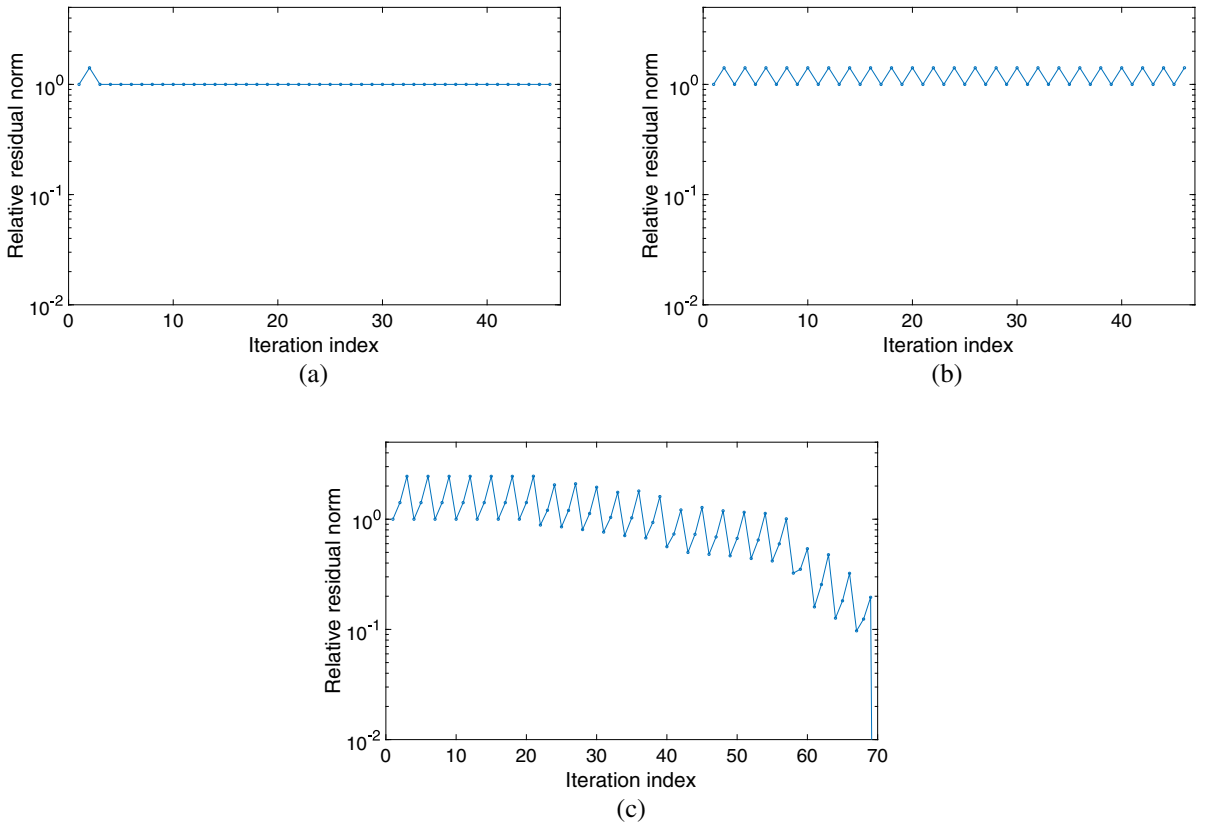

FIGURE 1 History of the relative residual norm for (a) full AR, (b) full AAR($p = 2$), and (c) full AAR($p = 3$) for the block circulant matrix test case with $l = 2$ and $b = 3$

Figure 1b. In conclusion, Figure 1c displays the residual norm for full AAR($p = 3$) that eventually attains the prescribed accuracy in computing the solution.

For the numerical examples performed in MATLAB, we employ a set of matrices selected from the SuiteSparse Matrix Collection (formerly known as the University of Florida Sparse Matrix Collection),[19] the Matrix Market Collection,[20] and a few others obtained from collaborators at Oak Ridge National Laboratory (matrices `sp1`, `sp3`, and `sp5` that arise from the numerical solution of radiation transport problems). In Table 1, we report the matrices and their most significant properties. The sources used to retrieve the matrices are specified in Table 1 as well. The notation MM is used to refer to the Matrix Market Collection, SS is used for the SuiteSparse Matrix Collection, and ORNL is used for those test cases provided by collaborators from Oak Ridge National Laboratory. Both real and complex matrices are considered. The theoretical discussion conducted in the previous sections of this work holds also for the complex field by replacing the transposition of a vector or of a matrix with its conjugate transpose.

The numerical results have been produced using the mathematical software MATLAB version R2016B with serial execution and without multithreading. The computer used is an Intel Xeon E5-4627.

The experiments aim to compare the performance of truncated AR, truncated AAR, augmented AAR, and restarted GMRES without preconditioning, with a diagonal preconditioner, with zero fill-in incomplete LU factorization (ILU(0) for short) and incomplete LU factorization with threshold (ILUT(τ) for short). With regard to the diagonal preconditioner, the diagonal entry of the preconditioner is set to 1 if the corresponding entry of the coefficient matrix A is zero. When ILUT(τ) is used, the tolerance τ controls the level of sparsity in the incomplete LU factors. We set $\tau = 10^{-4}$, and any zeros on the diagonal of the upper triangular factor are replaced by the local drop tolerance. The ILUT(τ) preconditioner is computed with a column pivoting, and the pivot is chosen as the maximum magnitude entry in the column. For more details about ILU(0) and ILUT(τ), we refer to the works of Benzi[21] and Saad[1] (see pp. 287–307). A symmetric reverse Cuthill–McKee reordering[22] has been applied to some matrices to allow a stable construction of the ILU factors. The matrices that needed a reordering for the ILU preconditioner are marked with an asterisk close to their name in Table 1. Since the matrices `xenon1` and `xenon2` were poorly scaled, the ILU factors have been computed for these matrices only after a diagonal scaling. The diagonal scaling has been applied via a diagonal preconditioner on these matrices. Those

TABLE 1 List of matrices used for numerical experiments performed in MATLAB

Matrix	Type	Size	Structure	Pos. def.	Source
fidap029	real	2,870	nonsymmetric	yes	MM
fidapm37*	real	9,152	nonsymmetric	yes	MM
raefsky5	real	6,316	nonsymmetric	yes	SS
sp1	real	18,207	nonsymmetric	yes	ORNL
sp3	real	36,414	nonsymmetric	yes	ORNL
sp5	real	54,621	nonsymmetric	yes	ORNL
bcsstk29*	real	13,992	symmetric	no	SS
sherman3	real	5,005	nonsymmetric	no	MM
sherman5	real	3,312	nonsymmetric	no	MM
fidap008*	real	3,096	nonsymmetric	no	MM
chipcool0	real	20,082	nonsymmetric	no	SS
e20r0000*	real	4,241	nonsymmetric	no	MM
spmsrtls	real	29,995	nonsymmetric	no	SS
III_Stokes*	real	20,896	nonsymmetric	no	SS
garon1*	real	3,175	nonsymmetric	no	SS
garon2*	real	13,535	nonsymmetric	no	SS
memplus	real	17,758	nonsymmetric	no	SS
saylr4	real	3,564	nonsymmetric	no	MM
xenon1*	real	48,600	nonsymmetric	no	SS
xenon2*	real	157,464	nonsymmetric	no	SS
venkat01	real	62,424	nonsymmetric	no	SS
QC2534	complex	2,534	non-Hermitian	no	SS
mplate	complex	5,962	non-Hermitian	no	SS
waveguide3	complex	21,306	non-Hermitian	no	SS
ABACUS_shell_hd	complex	23,412	non-Hermitian	no	SS
light_in_tissue	complex	29,282	non-Hermitian	no	SS
kim1	complex	38,415	non-Hermitian	no	SS
chevron2	complex	90,249	non-Hermitian	no	SS

situations where the matrices had zeros on the main diagonal were treated by setting to 1 the associated entries in the diagonal preconditioner. The Richardson relaxation parameter is set to $\omega = \frac{2}{\|A\|_\infty}$ if no preconditioner is employed and to $\omega = 0.2$ when an actual preconditioner is applied. This choice for the values of ω is motivated by the need to dampen the exponential growth associated with successive powers of the coefficient matrix A. Neglecting the ω, which means setting $\omega = 1$, generally doubles the number of iterations needed to converge. The weights β_k are all set to 1 (different values smaller than 1 have been tested as well, but results did not seem to significantly vary), and the size of the least squares problems is set to $m = 12$. Furthermore, the number of Richardson iterations between two Anderson updates for the AAR is $p = 6$. Setting $m = p$ allows an Anderson acceleration to use the information coming from the two last Anderson updates for the construction of the projection subspace. Through an empirical study of the tuning of the parameters, we found choosing $m = 2p$ to be a good compromise between convergence rate and computational complexity of the Anderson mixing. For ease of exposition with this setting, we refer to the truncated version of AR as truncated AR(12), to the truncated version of AAR as truncated AAR(6,12), and to the augmented version of AAR as augmented AAR(6,12). The least squares problems related to Anderson mixing steps are solved via QR factorization with column pivoting of the rectangular matrix R_ℓ. The length p of the cycle in restarted GMRES is set to $p = 10$ and $p = 30$. All the results are averaged over 10 runs with a randomly generated solution vector for each run (rounded to the nearest integer in the case of iteration counts). The right-hand side of the linear systems is obtained by multiplying the solution vector by the coefficient matrix A. The variation across single runs never exceeded 10% in terms of computational time. A threshold of 10^{-8} is used as a stopping criterion on the relative residual ℓ^2-norm.

The performance of the linear solvers is evaluated through *performance profiles*.[23] These graphic tools provide an immediate visual approach to compare the performance of multiple algorithms tested on a set of benchmark problems. In order to explain how this performance profiles are generated, let us refer to $\mathcal{S}$ as the set of solvers and to $\mathcal{P}$ as the test

set. We assume that we have n_s solvers and n_p problems. In this paper, performance profiles are used to compare the computational times. To this end, we introduce

$$t_{p,s} = \text{computing time to solve problem } p \text{ with solver } s.$$

The comparison between the times taken by each solver is based on the performance ratio defined as

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}}.$$

The performance ratio allows one to compare the performance of solver s on problem p with the best performance by any solver to address the same problem p. In case a specific solver s does not succeed in solving problem p, then a convention is adopted to set $r_{p,s} = r_M$, where r_M is a maximal value. In our case, we set $r_m = 10,000$. The performance of one solver compared to the others' on the whole benchmark set is displayed by the cumulative distribution function $\rho_s(\tau)$ that is defined as follows:

$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\}.$$

The value $\rho_s(\tau)$ represents the probability that solvers $s \in \mathcal{S}$ have a performance ratio $r_{p,s}$ less than or equal to the best possible ratio up to a scaling factor τ. The cumulative distribution function is nondecreasing, piecewise constant, and continuous from the right at each discontinuity. A particular interpretation is associated with the value $\rho_s(1)$. In fact, this value represents the probability that solver s outperforms every other solver from set $\mathcal{S}$ in solving a generic problem from set $\mathcal{P}$. The convention adopted that prescribes $r_{p,s} = r_M$ if solver s does not solve problem p leads to the reasonable assumption that

$$r_{p,s} \in [1, r_M].$$

Therefore, $\rho_s(r_M) = 1$, and

$$\rho_s^* = \lim_{\tau \to r_M^-} \rho_s(\tau)$$

represents the probability that solver s succeeds in solving a generic problem from set $\mathcal{P}$. In general, solvers with larger values of $\rho_s(\tau)$ are to be preferred. It may happen that some solvers from set $\mathcal{S}$ take a considerable amount of time in solving specific problems from set $\mathcal{P}$. This consequently requires choosing a sufficiently high value for r_M and extending the range of values of τ displayed in the performance profiles. This may lead to graphs that are difficult to interpret, since most of the main features of the curves may be shrunk on the far left of the graph window. Moreover, most of the window may be occupied to describe the trend of the curves for high values of τ where nothing relevant happens. This motivates the replacement of $\rho_s(\tau)$ with

$$\tau \mapsto \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : \log_2(r_{p,s}) \leq \tau\}.$$

Although using the log scale complicates the interpretation of the graph, it dedicates most of the space to values of τ where significant trends are captured and worth being discussed. From now on, we use this quantity for each performance profile displayed.

In Figure 2a, we report the results without preconditioning. We assessed that augmented AAR(6,12) overall performs better than any other linear solver both in terms of robustness and computational time. In particular, truncated AR(12), truncated AAR(6,12), and augmented AAR(6,12) are the only linear solvers to converge on matrices `fidapm37`, `fidap008`, `e20r0000`, `III_Stokes`, `garon1`, and `garon2`. The iteration count for AAR is usually higher than for AR or restarted GMRES. However, it should be remembered that the computational cost is not the same for all AAR iterations. Indeed, most of the iterations are simply relaxation sweeps that are far cheaper than least squares solves. We notice also that the positive definiteness does impact the convergence of the scheme. Indeed, augmented AAR(6,12) does not converge on the indefinite matrices `bcsstk29` and `spmsrtls`. As concerns the positive definite matrix `fidapm37`, restarted GMRES(10) and restarted GMRES(30) do not converge within the maximum number of iterations imposed because of the slow decrease of the relative residual norm.

The situation is similar when a diagonal preconditioner is used. The results in Figure 2b show augmented AAR(6,12) outperforming truncated AR(12), truncated AAR(6,12), restarted GMRES(10), and restarted GMRES(30).

As concerns the use of ILU(0) as a preconditioner, results are reported in Figure 2c. Truncated AR(12) and augmented AAR(6,12) reach convergence on the matrix `mplate`, whereas truncated AAR(6,12), restarted GMRES(10), and restarted GMRES(30) cannot. Augmented AAR(6,12) still generally converges faster than any other linear solver. However, the difference between the linear solvers is not as pronounced as for the other preconditioning configurations. This is reasonable to expect, since a more effective preconditioner is expected to cope with possible inefficiencies of the linear solver.

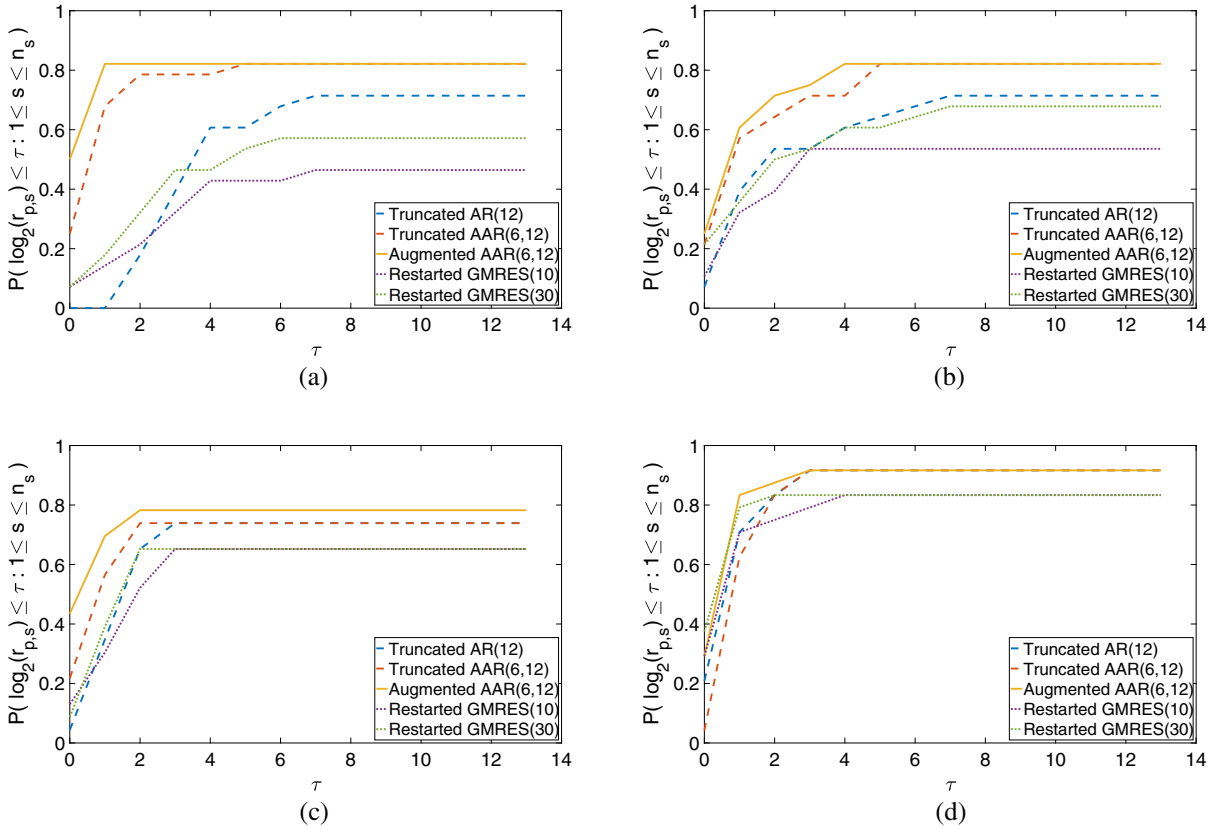

FIGURE 2 Performance profiles in a $\log_2$ scale for augmented AAR(6,12), restarted GMRES(10), and restarted GMRES(30). (a) Without preconditioning. (b) Diagonal preconditioner. (c) ILU(0) preconditioner. (d) ILUT(10^{-4}) preconditioner

TABLE 2 MATLAB performance expressed in CPU seconds for truncated AR(12), truncated AAR(6,12), augmented AAR(6,12), restarted GMRES(10), and restarted GMRES(30) on the matrix `mnplate` with no preconditioning or a diagonal, ILU(0), or ILUT(10^{-4}) preconditioner

	Tr. AR	Tr. AAR	AAAR	GMRES(10)	GMRES(30)
no prec	–	–	–	–	–
diag	–	–	–	–	–
ILU(0)	349.36	–	179.32	–	–
ILUT(10^{-4})	56.62	16.41	13.50	191.99	33.51

Results associated with ILUT(τ) are displayed in Figure 2d. In this case, truncated AR(12), truncated AAR(6,12), and augmented AAR(6,12) are the only linear solvers to reach convergence on the matrix `ABACUS_shell_hd`. In general, the performances of truncated AR(12), truncated AAR(6,12), augmented AAR(6,12), restarted GMRES(10), and restarted GMRES(30) are very similar. In this case as well, this can be explained by the quality of the preconditioner employed. The matrix R_ℓ happened not to be of full column rank for some iterations of AAR for `fidap029`, `raefsky5`, `sherman3`, and `sherman5`. However, as already mentioned in Remark 4, this fact did not hinder convergence.

To better understand the results obtained in MATLAB, we show in Table 2 the performance expressed in CPU seconds for truncated AR(12), truncated AAR(6,12), augmented AAR(6,12), restarted GMRES(10), and restarted GMRES(30) on the matrix `mplate` with no preconditioning or a diagonal, ILU(0), or ILUT(10^{-4}) preconditioner. When no preconditioner or a diagonal preconditioner is used, none of the linear solvers converge. When ILU(0) is employed, only truncated AR(12) and augmented AAR(6,12) converge. All the linear solvers succeed in attaining the prescribed accuracy only when the preconditioner ILUT(10^{-4}) is used. For this test case, augmented AAR(6,12) performs better than the other linear solvers either because it reaches convergence when the others do not or because it converges faster.

4.2 | Numerical experiments in MPI

The code used to produce the last set of numerical experiments is written in `C` language. The cluster used for the experiments includes 16×64-core (4 sockets, 16 cores per processor) AMD Opteron 6200 "Interlagos" nodes. Each of these nodes has 128-GB RAM and is connected via Infiniband (IB) QDR connection. Each of these nodes is running RHEL6.7 using the 2.6.32-573.12.1.el6.x86_64 kernel. Therefore, the total number of cores available is 1,024. Furthermore, in this case, the set of matrices is selected from the SuiteSparse Matrix Collection. Details about the features of the problem are provided in Table 3. The experiments compare the performance of augmented AAR(6,12) with restarted GMRES(10) and restarted GMRES(30).

The library used to handle numerical linear algebra operations is the `PETSc` library,[24] version 3.5.3. The solution to the linear system is generated as a random vector, and the right-hand side vector is constructed by multiplying the solution vector by the coefficient matrix. The stopping criterion is still the relative residual in the ℓ^2-norm with a threshold set to 10^{-8}. The class defining augmented AAR has been written by the authors of this paper, whereas the version of restarted GMRES used is the one provided by the `KSP` class in `PETSc`. The preconditioner adopted is Block-ILU(0) with a total number of blocks equal to 1,024. This preconditioner is used in the MPI tests since it is the intermediate quality preconditioner from the MATLAB tests. The number of blocks has been set to 1,024 because this is the total number of processors available in the cluster used for MPI experiments. The version of PETSc installed on the cluster allows an MPI process to handle multiple blocks but not the opposite, that is, a single block cannot be shared by multiple MPI processes. Because of this, 1,024 has been chosen as the total number of blocks.

The least squares problems to compute the Anderson mixing are solved by explicitly building the normal equation

$$\left(R_k^T R_k\right) \mathbf{g}^k = R_k^T \mathbf{r}^k \tag{21}$$

and solving it on each processor with the singular value decomposition (SVD) of $R_k^T R_k$. The SVD is performed by the `LAPACKE_dgelsd` routine inside the `LAPACK` library.[25]

Although we are aware that the ill-conditioning of the least squares problem can be severely affected by explicitly building the normal equation (21), this does not seem to occur for the problems and preconditioner considered. Moreover, the explicit construction of the normal equation has appealing properties from a parallelism perspective. To explain why, let us temporarily assume that the number of rows is a multiple of the number of the `MPI` processes instantiated and that the number of rows owned by each process is equal to n_{loc}. Matrices and vectors are distributed row-wise across the `MPI` processes. Therefore, an MPI process with an ID equal to b owns rows from index $[(b-1)\cdot n_{\text{loc}} + 1]$ to $[b \cdot n_{\text{loc}}]$. This implies that each process can locally compute

$$\left[\left(R_{k,b}^{\text{loc}}\right)^T R_{k,b}^{\text{loc}}\right] \quad \text{and} \quad \left(R_{k,b}^{\text{loc}}\right)^T \mathbf{r}_{\text{loc},b}^k, \tag{22}$$

where $R_{k,b}^{\text{loc}}$ is the submatrix of R_k obtained by extracting the rows with indices from $[(b-1)\cdot n_{\text{loc}} + 1]$ to $[b \cdot n_{\text{loc}}]$ and $\mathbf{r}_{\text{loc},b}^k$ is the subvector of $\mathbf{r}^k$ with indices from $[(b-1)\cdot n_{\text{loc}} + 1]$ to $[b \cdot n_{\text{loc}}]$. The matrices $[(R_{k,b}^{\text{loc}})^T R_{k,b}^{\text{loc}}]$ and $(R_{k,b}^{\text{loc}})^T \mathbf{r}_{\text{loc},b}^k$ are computed independently by each process without any interprocessor communication needed. The only interaction among processes is needed to reconstruct the global quantities to obtain Equation (21). In particular, only an `MPI_Allreduce` operation is employed using the sum as global reduction operation to reconstruct $(R_k^T R_k)$ and $R_k^T \mathbf{r}^k$ as follows:

$$\left(R_k^T R_k\right) = \sum_{b=1}^{n_{\text{proc}}} \left[\left(R_{k,b}^{\text{loc}}\right)^T R_{k,b}^{\text{loc}}\right], \quad R_k^T \mathbf{r}^k = \sum_{b=1}^{n_{\text{proc}}} \left(R_{k,b}^{\text{loc}}\right)^T \mathbf{r}_{\text{loc},b}^k,$$

where n_{proc} is the total number of `MPI` processes instantiated. Once $(R_k^T R_k)$ and $R_k^T \mathbf{r}^k$ are reconstructed and locally stored on each processor, the solution to the least squares problem is locally computed by solving Equation (21). This allows

TABLE 3 List of matrices used for numerical experiments in `MPI`

Matrix	Type	Size	nnz	Field of application
atmosmodl	real	1,489,752	10,319,760	computational fluid dynamics
circuit5M_dc	real	3,523,317	59,524,291	circuit simulation
Freescale1	real	3,428,755	17,052,626	circuit simulation
CurlCurl_4	real	2,380,515	26,515,867	model reduction problem
Transport	real	1,602,111	23,487,281	structural engineering

TABLE 4 `MPI` experiments: Block-ILU(0) preconditioner with 1,024 diagonal blocks

	Number of processes				
atmosmodl	**32**	**64**	**128**	**256**	**512**
AAR(6,12)	5.81 (s)	2.31 (s)	1.18 (s)	0.80 (s)	0.72 (s)
Restarted GMRES(10)	4.99 (s)	2.52 (s)	1.24 (s)	0.78 (s)	0.70 (s)
Restarted GMRES(30)	4.98 (s)	2.29 (s)	1.27 (s)	0.70 (s)	0.59 (s)
	Number of processes				
circuit5M_dc	**32**	**64**	**128**	**256**	**512**
AAR(6,12)	4.94 (s)	2.33 (s)	1.51 (s)	1.35 (s)	1.30 (s)
Restarted GMRES(10)	3.11 (s)	1.83 (s)	1.31 (s)	1.34 (s)	1.37 (s)
Restarted GMRES(30)	3.12 (s)	1.82 (s)	1.40 (s)	1.38 (s)	1.39 (s)
	Number of processes				
Freescale1(RCM)	**32**	**64**	**128**	**256**	**512**
AAR(6,12)	753.29 (s)	375.20 (s)	243.05 (s)	253.01 (s)	270.59 (s)
Restarted GMRES(10)	–	–	–	–	–
Restarted GMRES(30)	–	–	–	–	–
	Number of processes				
CurlCurl4	**32**	**64**	**128**	**256**	**512**
AAR(6,12)	106.51	50.48 (s)	26.17 (s)	14.12 (s)	14.35 (s)
Restarted GMRES(10)	466.85	261.19 (s)	125.71 (s)	97.47 (s)	88.43 (s)
Restarted GMRES(30)	459.79	260.29 (s)	126.22 (s)	97.11 (s)	91.23 (s)
	Number of processes				
Transport	**32**	**64**	**128**	**256**	**512**
AAR(6,12)	120.11 (s)	55.48 (s)	35.60 (s)	20.26 (s)	40.70 (s)
Restarted GMRES(10)	590.42 (s)	270.52 (s)	158.32 (s)	121.90 (s)	171.39 (s)
Restarted GMRES(30)	591.14 (s)	269.04 (s)	163.27 (s)	121.09 (s)	167.50 (s)

one to cap the global communication and the memory storage requirement. In fact, only one global communication to transfer $m(m+1)$ values is needed to solve each least squares problem. Moreover, each process has to own only enough memory to compute the SVD of an $m \times m$ matrix.

The metric used to monitor the performance is the wall clock time expressed in seconds. All the results are averaged over 10 runs with a randomly generated solution vector for each run (rounded to the nearest integer in the case of iteration counts). The right-hand side of the linear systems is obtained by multiplying the solution vector by the coefficient matrix A.

Results using Block-ILU(0) as a preconditioner are displayed in Table 4, where augmented AAR exhibits promising results. Indeed, the computational locality of the preconditioner accommodates the strong scalability of the overall iterative procedure up to 256 cores. The performance deteriorates going up to 512 cores. This may be explained by a plausible communication overhead. The computational time for augmented AAR is slightly higher than restarted GMRES for the matrices `atmosmodl` and `circuit5M_dc`, whereas a significant performance improvement with respect to restarted GMRES is assessed for the other test cases. Indeed, augmented AAR is the only method to converge on the matrix `Freescale1`. Moreover, the times taken by augmented AAR to converge on the matrices `CurlCurl4` and `Transport` are significantly lower than the ones taken by restarted GMRES. Therefore, AAR exhibits the potential to outperform existing solvers in a high-performance computing environment.

5 | CONCLUSIONS AND FUTURE DEVELOPMENTS

In this work, we have explored different techniques that can be used to accelerate one-level standard relaxation algorithms. Although Anderson mixing is widely recognized as an efficient acceleration for nonlinear problems, studies have confirmed that it benefits also linear fixed-point iterations. Standard approaches in this respect alternate a Richardson sweep and an Anderson mixing at each iteration as in Anderson–Richardson. All the variants of this scheme are comparable with either full GMRES or truncated GMRES. However, a drawback of these techniques is that they do not address issues such as the avoidance of global communications to leverage the performance on next-generation computers. The AAR method has been recently proposed to address these shortcomings. Theoretical equivalence between

full AAR and full GMRES in exact arithmetic has been proved in this work. Moreover, a study of the basis generated by AAR shows that the algorithm is more robust than AR against stagnations. Furthermore, we introduced a new variant of AAR, called augmented AAR. This variant has the advantage of guaranteeing convergence on linear systems with positive definite matrices. Preliminary results with different preconditioners are promising and suggest that augmented AAR may be an appealing alternative to restarted GMRES as to reduction of both time and global communications. Furthermore, a preliminary parallel implementation of the algorithm has been developed in order to assess the impact of reduced communication (leading to enhanced concurrency) compared to standard preconditioned Krylov methods.

Future work should focus on exploring the properties of the subspaces used for the Anderson mixing and by studying the performance sensitivity of AAR to quantities like relaxation parameters, periodic interval length, and number of iterates involved in the mixing.

ACKNOWLEDGEMENTS

I would like to thank Michele Benzi (Emory University) and Phanish Suryanarayana (Georgia Institute of Technology) for the helpful suggestions and Steven Hamilton (Oak Ridge National Laboratory) for providing some of the test matrices used. This work was supported in part by the United States Department of Energy (Office of Science) under Grant ERKJ247. This research used resources of the Oak Ridge Leadership Computing Facility, which is a Department of Energy Office of Science User Facility supported under Contract DE-AC05-00OR22725.

CONFLICT OF INTEREST

There are no conflicts of interest to this work.

ORCID


Massimiliano Lupo Pasini https://orcid.org/0000-0002-4980-6924

REFERENCES

1. Saad Y. Iterative methods for sparse linear systems. 2nd ed. Philadelphia, PA: SIAM; 2003.
2. Golub GH, Varga RS. Chebyshev semi-iterative methods, successive over-relaxation iterative methods, and second-order Richardson iterative methods. Numer Math. 1961;3(1):157–168.
3. Manteuffel TA. The Tchebychev iteration for nonsymmetric linear systems. Numer Math. 1977;28(3):307–327.
4. Hoemmen M. Communication-avoiding Krylov subspace methods. [PhD dissertation]. Berkeley, CA: University of California, Berkeley; 2010.
5. McInnes LC, Smith B, Zhang H, Mills RT. Hierarchical Krylov and nested Krylov methods for extreme-scale computing. Parallel Comput. 2014;40(1):17–31.
6. Yamazaki I, Rajamanickam S, Boman EG, Hoemmen M, Heroux MA, Tomov S. Domain decomposition preconditioners for communication-avoiding Krylov methods on a hybrid CPU/GPU cluster. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis; 2014 Nov 16–21; New Orleans, LA. IEEE: Piscataway, NJ; 2014. p. 933–944.
7. Stoyanov M, Webster C. Numerical analysis of fixed point algorithms in the presence of hardware faults. SIAM J Sci Comput. 2015;37(5):C532–C553.
8. Pratapa PP, Suryanarayana P, Pask JE. Anderson acceleration of the Jacobi iterative method: an efficient alternative to Krylov methods for large, sparse linear systems. J Comput Phys. 2016;306:43–54.
9. Pratapa PP, Suryanarayana P, Pask JE. Alternating Anderson–Richardson method: an efficient alternative to preconditioned Krylov methods for large, sparse linear systems. Comput Phys Commun. 2019;234:278–285.
10. Anderson DG. Iterative procedures for nonlinear integral equations. J Assoc Comput Mach. 1965;12(4):547–560.
11. Householder AS. The theory of matrices in numerical analysis. New York, NY: Blaisdell Publishing; 1964.
12. Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput. 1986;7(3):856–869.
13. Fang H, Saad Y. Two classes of multisecant methods for nonlinear acceleration. Numer Linear Algebra Appl. 2009;16(3):197–221.
14. Loffeld J, Woodward CS. Considerations on the implementation and use of Anderson acceleration on distributed memory and GPU-based parallel computers. Berlin Germany: Springer; 2015. Association for women in mathematics research symposium.
15. Potra FA, Engler H. A characterization of the behavior of the Anderson acceleration on linear problems. Linear Algebra Appl. 2013;438(3):1002–1011.
16. Toth A, Kelley CT. Convergence analysis for Anderson acceleration. SIAM J Numer Anal. 2015;53(2):805–819.

17. Walker HF, Ni P. Anderson acceleration for fixed-point iterations. SIAM J Numer Anal. 2011;49(4):1715–1735.
18. Wilkinson JH. The algebraic eigenvalue problem. Oxford, UK: Clarendon Press; 1965.
19. Davis T. SuiteSparse Matrix Collection Formerly the University of Florida Sparse Matrix Collection. Available from: http://www.cise.ufl.edu/research/sparse/matrices/
20. Matrix Market Collection. Available from: http://math.nist.gov/MatrixMarket/
21. Benzi M. Preconditioning techniques for large linear systems: a survey. J Comput Phys. 2002;182(2):418–477.
22. Cuthill E, McKee J. Reducing the bandwidth of sparse symmetric matrices. Proceedings of the 1969 24th National Conference; 1969 Aug 26–28; New York, NY. New York, NY: ACM; 1969. p. 157–172.
23. Dolan ED, Moré JJ. Benchmarking optimization software with performance profiles. Math Program. 2002;91(2):201–213.
24. PETSc/Tao. Available from: https://www.mcs.anl.gov/petsc/
25. LAPACK – Linear Algebra PACKage. Available from: http://www.netlib.org/lapack/

How to cite this article: Lupo Pasini M. Convergence analysis of Anderson-type acceleration of Richardson's iteration. *Numer Linear Algebra Appl*. 2019;e2241. https://doi.org/10.1002/nla.2241