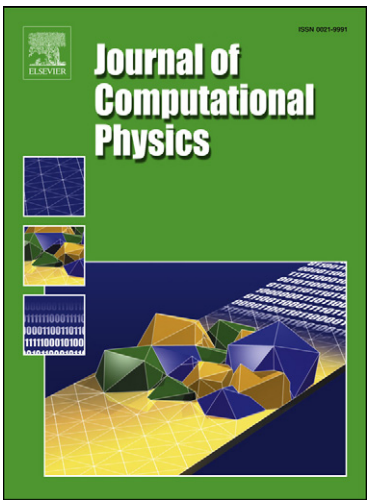
Accepted Manuscript



Anderson acceleration of the Jacobi iterative method: An efficient alternative to Krylov methods for large, sparse linear systems

Phanisri P. Pratapa, Phanish Suryanarayana, John E. Pask

PII: S0021-9991(15)00758-5
DOI: http://dx.doi.org/10.1016/j.jcp.2015.11.018
Reference: YJCPH 6235

To appear in: *Journal of Computational Physics*

Received date: 18 February 2015
Revised date: 10 August 2015
Accepted date: 11 November 2015

Please cite this article in press as: P.P. Pratapa et al., Anderson acceleration of the Jacobi iterative method: An efficient alternative to Krylov methods for large, sparse linear systems, *J. Comput. Phys.* (2015), http://dx.doi.org/10.1016/j.jcp.2015.11.018

Anderson acceleration of the Jacobi iterative method: an efficient alternative to Krylov methods for large, sparse linear systems

Phanisri P. Pratapa[a], Phanish Suryanarayana[*,a], John E. Pask[b]

[a] *College of Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA*
[b] *Physics Division, Lawrence Livermore National Laboratory, Livermore, CA 94550, USA*

Abstract

We employ Anderson extrapolation to accelerate the classical Jacobi iterative method for large, sparse linear systems. Specifically, we utilize extrapolation at periodic intervals within the Jacobi iteration to develop the Alternating Anderson-Jacobi (AAJ) method. We verify the accuracy and efficacy of AAJ in a range of test cases, including nonsymmetric systems of equations. We demonstrate that AAJ possesses a favorable scaling with system size that is accompanied by a small prefactor, even in the absence of a preconditioner. In particular, we show that AAJ is able to accelerate the classical Jacobi iteration by over four orders of magnitude, with speed-ups that increase as the system gets larger. Moreover, we find that AAJ significantly outperforms the Generalized Minimal Residual (GMRES) method in the range of problems considered here, with the relative performance again improving with size of the system. Overall, the proposed method represents a simple yet efficient technique that is particularly attractive for large-scale parallel solutions of linear systems of equations.

Key words: Linear systems of equations, Fixed-point iteration, Jacobi method, Anderson extrapolation, Nonsymmetric matrix, Poisson equation, Helmholtz equation, Parallel computing

1. Introduction

In nearly all areas of computational physics, it is common to encounter linear systems of equations of the form

$$\begin{aligned} &\mathbf{A}\mathbf{x} = \mathbf{b}\,, \\ &\mathbf{A} \in \mathbb{C}^{N\times N}\,,\ \mathbf{x} \in \mathbb{C}^{N\times 1} \text{ and } \mathbf{b} \in \mathbb{C}^{N\times 1}\,, \end{aligned} \tag{1}$$

where $\mathbb{C}$ is the set of all complex numbers. For small systems, solution strategies based on direct methods are typically the preferred choice. However, as the size of the system increases, it becomes necessary to employ iterative approaches in order to efficiently determine the solution. The basic fixed-point techniques that have been developed for this purpose include the Richardson, Jacobi, Gauss-Seidel, and Successive over-relaxation (SOR) methods [1]. However, these approaches suffer from relatively large prefactors and poor scaling with system size. This makes them unattractive for solving large systems of equations compared to Krylov subspace approaches such as the conjugate gradient [2] and Generalized Minimal Residual (GMRES) [3] methods.

[*] Corresponding Author (*phanish.suryanarayana@ce.gatech.edu)*

In spite of the aforementioned limitations of basic fixed-point methods, the Jacobi iteration stands out because of its tremendous simplicity and potential for massive parallelization. This motivates the development of strategies that are able to significantly accelerate the convergence of the Jacobi method, while maintaining its underlying locality and simplicity to the maximum extent possible. Examples of such approaches include the Chebyshev acceleration technique [1] and the recently proposed Scheduled Relaxation Jacobi (SRJ) method [4]. However, Chebyshev acceleration requires knowledge of the extremal eigenvalues of the matrix $\mathbf{A}$. Furthermore, the SRJ method as currently formulated is restricted to linear systems arising from second-order finite-difference discretization of elliptic equations. For such reasons, Krylov subspace techniques remain yet the methods of choice for the solution of large, sparse linear systems.

Anderson's extrapolation [5] is a widely used approach for accelerating the convergence of non-linear fixed-point iterations. In particular, it has found great success in the field of electronic structure calculations to speed-up the convergence of the Self-Consistent Field (SCF) iteration [6, 7, 8]. In this context, Anderson's technique is also referred to as Pulay's Direct Inversion in the Iterative Subspace (DIIS) method [9]. This approach has also recently been adopted to accelerate the convergence of non-linear fixed-point iterations in coupled fluid-structure transient thermal problems [10], as well as neutronics and plasma physics [11]. As discussed in Section 5.2.2, in the context of linear systems of equations, Anderson's technique bears a close connection to the GMRES method [12, 13, 14].

In the present work, we treat the Jacobi method as a fixed-point iteration and employ Anderson's extrapolation to accelerate its convergence. However, rather than applying the extrapolation in every step, we employ it at periodic intervals within the Jacobi iteration. We refer to this approach as the Alternating Anderson Jacobi (AAJ) method. We verify the accuracy, efficiency, and generality of AAJ in a range of test cases, including nonsymmetric, 3D Poisson, and complex-valued Helmholtz problems. In particular, we demonstrate that AAJ is able to accelerate the classical Jacobi method by factors exceeding $10,000$, and substantially outperform GMRES in the process. The remainder of this paper is organized as follows. In Sections 2 and 3, we discuss the Jacobi method and the Anderson-Jacobi (AJ) method, respectively. Next, we present the AAJ method in Section 4, which we validate through a range of test cases in Section 5. Finally, we conclude in Section 6.

2. Jacobi method

Consider the linear system of equations described by Eqn. 1. The matrix $\mathbf{A}$ can be split as

$$\mathbf{A} = \mathbf{D} + \mathbf{R}\,, \tag{2}$$

where all the off-diagonal components of the matrix $\mathbf{D} \in \mathbb{C}^{N\times N}$ and the diagonal components of $\mathbf{R} \in \mathbb{C}^{N\times N}$ are identically zero. Using this decomposition, Eqn. 1 can be rewritten as the fixed-point problem

$$\mathbf{x} = \mathbf{g}(\mathbf{x})\,, \tag{3}$$

where the mapping

$$\mathbf{g}(\mathbf{x}) = \mathbf{D}^{-1}(\mathbf{b} - \mathbf{R}\mathbf{x})\,. \tag{4}$$

In this setting, the residual can be defined to be

$$\mathbf{f}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) - \mathbf{x}\,. \tag{5}$$

Further, the error

$$\mathbf{e}(\mathbf{x}) = \mathbf{x} - \mathbf{x}^*\,, \tag{6}$$

where $\mathbf{x}^*$ denotes the solution of the linear system in Eqn. 1. This Jacobi-type reformulation is predicated on the assumption that there are no zeros on the diagonal of $\mathbf{D}$, and therefore by extension the diagonal of $\mathbf{A}$.

The Jacobi method [1, 15] proposes to solve the fixed-point problem in Eqn. 3 using an iteration of the form

$$\mathbf{x}_{k+1} = \mathbf{g}(\mathbf{x}_k)\,, \tag{7}$$

where the subscript k is used to represent the iteration number. In this approach, the relationship between the error/residual in any two consecutive iterates can be shown to be

$$\mathbf{e}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \mathbf{D}^{-1}\mathbf{A}\right)\mathbf{e}(\mathbf{x}_k)\,, \quad \mathbf{f}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \mathbf{D}^{-1}\mathbf{A}\right)\mathbf{f}(\mathbf{x}_k)\,. \tag{8}$$

It follows that the Jacobi method is effective at nullifying the error/residual components corresponding to eigenvalues of $(\mathbf{I} - \mathbf{D}^{-1}\mathbf{A})$ whose magnitudes are close to zero, and relatively ineffective at nullifying components corresponding to eigenvalues with magnitudes near but less than unity. In particular, convergence of the Jacobi iteration requires

$$\|\mathbf{I} - \mathbf{D}^{-1}\mathbf{A}\| < 1\,, \tag{9}$$

where $\|.\|$ refers to the 2-norm. Such a constraint limits the applicability of the Jacobi method, which motivates suitable modification of the underlying iteration.

The Weighted Jacobi (WJ) method [1, 15] represents a generalization of the aforedescribed Jacobi technique, wherein the fixed-point iteration in Eqn. 7 takes the form

$$\mathbf{x}_{k+1} = (1-\omega)\mathbf{x}_k + \omega\mathbf{g}(\mathbf{x}_k)\,. \tag{10}$$

In terms of the residual, the above equation reduces to

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \omega\mathbf{f}(\mathbf{x}_k)\,. \tag{11}$$

The scalar $\omega \in \mathbb{C}$ is referred to as the relaxation parameter, with the specific choice of $\omega = 1$ yielding the standard Jacobi iteration. Analogous to the Jacobi method, the progression of error/residual in the WJ iteration can be expressed as

$$\mathbf{e}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{A}\right)\mathbf{e}(\mathbf{x}_k)\,, \quad \mathbf{f}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{A}\right)\mathbf{f}(\mathbf{x}_k)\,. \tag{12}$$

It follows that the WJ approach is efficient for error/residual components corresponding to eigenvalues of $(\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{A})$ whose magnitudes are near zero, while relatively inefficient at reducing components corresponding to eigenvalues with magnitudes close to but less than one. Furthermore, convergence of the WJ method requires

$$\|\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{A}\| < 1\,. \tag{13}$$

Overall, when $\mathbf{D}^{-1}\mathbf{A}$ has eigenvalues with positive real part, an appropriately small relaxation parameter ω can be chosen to enable convergence when the standard Jacobi method diverges. However, doing so negatively impacts the performance of the WJ method in neutralizing error/residual components corresponding to small-magnitude eigenvalues of $\mathbf{D}^{-1}\mathbf{A}$. A detailed description and analysis of the classical Jacobi method and its weighted counterpart can be found in standard texts [1, 15, 16, 17].

3. Anderson-Jacobi method

The extrapolation method of Anderson [5] is a widely used approach for accelerating the convergence of non-linear fixed-point iterations [7, 10, 11]. It has been particularly successful at enhancing the convergence of the Self-Consistent Field (SCF) method in electronic structure calculations [8, 12]. In the linear setting, it has been shown that Anderson's approach bears a close connection to the Generalized Minimal Residual (GMRES) method [12, 13, 14]. Notably, Anderson demonstrated in his original work [5] that the proposed extrapolation technique can be employed to significantly improve the performance of the Jacobi method, among others. In spite of this, such an approach for solving linear systems of equations—which we shall refer to as the Anderson-Jacobi (AJ) method—has received little attention subsequently. In this work, we demonstrate that AJ is an efficient method for solving large systems of equations. In fact, as shown in Section 5, it is able to consistently outperform GMRES for all the cases considered here.

In the AJ method, the fixed-point iteration in Eqn. 11 is generalized to

$$\mathbf{x}_{k+1} = \bar{\mathbf{x}}_k + \beta \mathbf{f}(\bar{\mathbf{x}}_k)\,, \tag{14}$$

where $\bar{\mathbf{x}}_k$ denotes the weighted average of the previous iterates and $\beta \in \mathbb{C}$ is a parameter. Specifically,

$$\bar{\mathbf{x}}_k = \mathbf{x}_k - \sum_{j=1}^{m} \gamma_j \left(\mathbf{x}_{k-m+j} - \mathbf{x}_{k-m+j-1}\right)\,, \tag{15}$$

where $m+1$ is the number of iterates used for extrapolation. The scalars $\Gamma_k = \begin{bmatrix}\gamma_1 & \gamma_2 & \dots & \gamma_m\end{bmatrix}^T \in \mathbb{C}^{m\times 1}$ are chosen so as to minimize the l_2 norm of the residual, i.e.,

$$\Gamma_k = \arg\min_{\Gamma_k} \|\mathbf{f}(\bar{\mathbf{x}}_k)\|^2\,. \tag{16}$$

Introducing the notation

$$\mathbf{X}_k = \begin{bmatrix}(\mathbf{x}_{k-m+1} - \mathbf{x}_{k-m}) & (\mathbf{x}_{k-m+2} - \mathbf{x}_{k-m+1}) & \dots & (\mathbf{x}_k - \mathbf{x}_{k-1})\end{bmatrix} \in \mathbb{C}^{N\times m}\,, \tag{17}$$

$$\mathbf{F}_k = \begin{bmatrix}(\mathbf{f}(\mathbf{x}_{k-m+1}) - \mathbf{f}(\mathbf{x}_{k-m})) & (\mathbf{f}(\mathbf{x}_{k-m+2}) - \mathbf{f}(\mathbf{x}_{k-m+1})) & \dots & (\mathbf{f}(\mathbf{x}_k) - \mathbf{f}(\mathbf{x}_{k-1}))\end{bmatrix} \in \mathbb{C}^{N\times m}\,, \tag{18}$$

the optimized Γ_k satisfy the relation

$$\left(\mathbf{F}_k^T \mathbf{F}_k\right) \Gamma_k = \mathbf{F}_k^T \mathbf{f}(\mathbf{x}_k)\,. \tag{19}$$

Thereafter, the update formula in Eqn. 14 can be written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \beta \mathbf{f}(\mathbf{x}_k) - (\mathbf{X}_k + \beta \mathbf{F}_k)(\mathbf{F}_k^T \mathbf{F}_k)^{-1} \mathbf{F}_k^T \mathbf{f}(\mathbf{x}_k)\,. \tag{20}$$

The aforedescribed AJ approach can also be interpreted as a multi-secant type method [7, 18]. In this context, the AJ iteration generalizes Eqn. 11 to take the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{C}_k \mathbf{f}(\mathbf{x}_k)\,. \tag{21}$$

The matrix $\mathbf{C}_k \in \mathbb{C}^{N\times N}$ is set to the solution of the constrained minimization [7, 8]

$$\begin{aligned} &\min_{\mathbf{C}_k} \|\mathbf{C}_k + \beta \mathbf{I}\| \\ &\text{s.t.} \quad \mathbf{C}_k \mathbf{F}_k = \mathbf{X}_k\,, \end{aligned} \tag{22}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix, with $\mathbb{R}$ denoting the set of all real numbers. The solution to this variational problem is

$$\mathbf{C}_k \quad = \quad \beta\mathbf{I} - (\mathbf{X}_k + \beta\mathbf{F}_k)(\mathbf{F}_k^T\mathbf{F}_k)^{-1}\mathbf{F}_k^T \,. \tag{23}$$

On substituting this expression for $\mathbf{C}_k$ into Eqn. 21, the AJ fixed-point iteration in Eqn. 20 is recovered. It is worth noting that the constraint in Eqn. 22 can be expressed as

$$(\mathbf{C}_k\mathbf{D}^{-1}\mathbf{A})\mathbf{X}_k = \mathbf{X}_k \,, \tag{24}$$

from which it can be inferred that $\mathbf{C}_k$ is designed to approximate $\mathbf{A}^{-1}\mathbf{D}$ in Anderson's extrapolation.

In the AJ method, the relation between the error/residual at consecutive iterations can be written as

$$\mathbf{e}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \mathbf{C}_k\mathbf{D}^{-1}\mathbf{A}\right)\mathbf{e}(\mathbf{x}_k) \,, \quad \mathbf{f}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \mathbf{C}_k\mathbf{D}^{-1}\mathbf{A}\right)\mathbf{f}(\mathbf{x}_k) \,. \tag{25}$$

From these equations, it can be deduced that the AJ update is a contraction provided

$$\|\mathbf{I} - \mathbf{C}_k\mathbf{D}^{-1}\mathbf{A}\| < 1 \,. \tag{26}$$

Furthermore, the fixed-point iteration will converge faster when $\mathbf{C}_k$ is able to better approximate $\mathbf{A}^{-1}\mathbf{D}$. As a result, the AJ method significantly accelerates the convergence of the basic Jacobi iteration, as verified in Section 5. Overall, the AJ method can be viewed as a generalization of the Weighted Jacobi (WJ) method, since it replaces the constant matrix $\omega\mathbf{I}$ with a dynamically updated matrix $\mathbf{C}_k$. Furthermore, AJ reduces to WJ on setting $\beta = \omega$ and $m = 0$.

4. Alternating Anderson-Jacobi method

The weighted Jacobi method typically suffers from slow convergence due to its inability to efficiently reduce the 'low frequency' components of the error/residual. Here and henceforth, 'low frequency' and 'high frequency' error/residual components denote those corresponding to the eigenvalues of $(\mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{A})$ with magnitude close to unity and zero, respectively. In this work, we aim to develop an accelerated variant of the Jacobi method, while seeking to retain its tremendous simplicity, locality, and potential for scalability on massively parallel architectures. We shall refer to this generalization, which incorporates both Weighted Jacobi (WJ) and Anderson-Jacobi (AJ) updates, as the Alternating Anderson-Jacobi (AAJ) method.

In Fig. 1, we outline the algorithm of the proposed AAJ method. We have used $\mathbf{x}_0$ to represent the initial guess, r to denote the normalized l_2 norm of the residual, and ϵ to signify the tolerance specified for convergence. In the AAJ approach, the fixed-point iteration takes the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{B}_k\mathbf{f}(\mathbf{x}_k) \,, \tag{27}$$

where the matrix

$$\mathbf{B}_k = \begin{cases} \omega\mathbf{I} & \text{if } (k+1)/p \notin \mathbb{N} \,, \\ \beta\mathbf{I} - (\mathbf{X}_k + \beta\mathbf{F}_k)(\mathbf{F}_k^T\mathbf{F}_k)^{-1}\mathbf{F}_k^T & \text{if } (k+1)/p \in \mathbb{N} \,. \end{cases} \tag{28}$$

In this setting, the relationship between the error/residual at consecutive iterations can be written as

$$\mathbf{e}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \mathbf{B}_k\mathbf{D}^{-1}\mathbf{A}\right)\mathbf{e}(\mathbf{x}_k) \,, \quad \mathbf{f}(\mathbf{x}_{k+1}) = \left(\mathbf{I} - \mathbf{B}_k\mathbf{D}^{-1}\mathbf{A}\right)\mathbf{f}(\mathbf{x}_k) \,. \tag{29}$$

It follows that the AAJ update is a contraction provided

$$\|\mathbf{I} - \mathbf{B}_k\mathbf{D}^{-1}\mathbf{A}\| < 1\,. \tag{30}$$

Overall, AAJ represents a generalization of the WJ method wherein the WJ update in Eqn. 11 is replaced with an AJ update described by Eqn. 20 every p^{th} iteration. It can also be viewed as a generalization of the AJ method in which $\mathbf{C}_k = \omega\mathbf{I}$ if $(k+1)/p \notin \mathbb{N}$. Indeed, the AJ method is recovered for $p = 1$ and the WJ method is recovered in the limit $p \to \infty$.

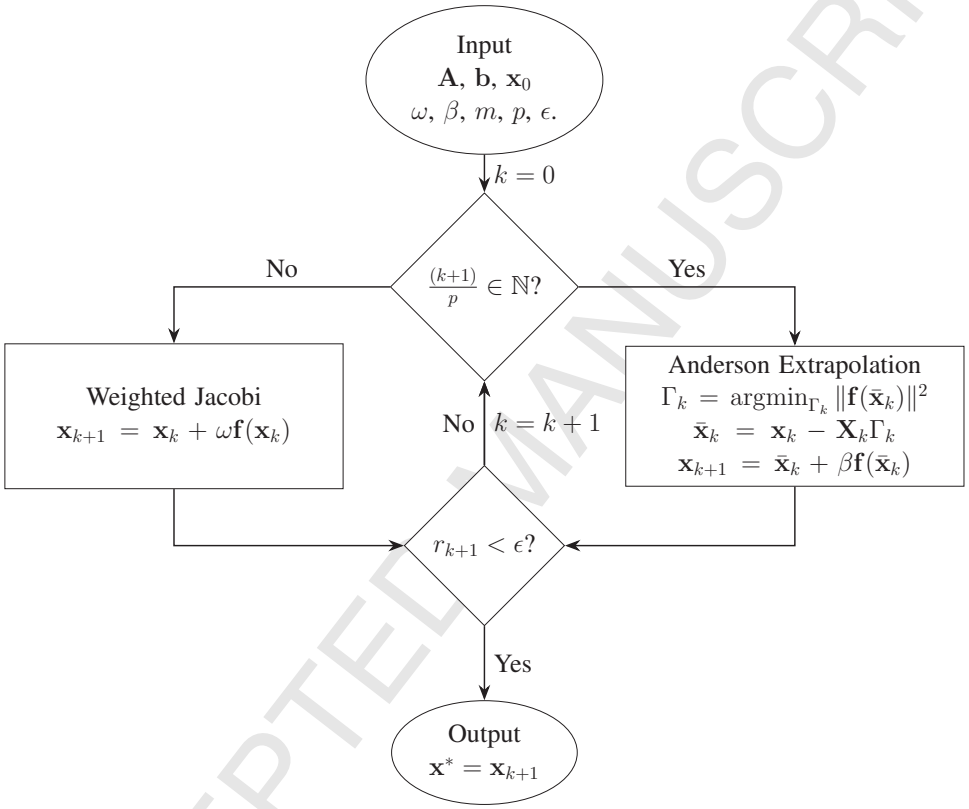


Figure 1: The Alternating Anderson-Jacobi (AAJ) method.

In this work, we have employed Anderson's extrapolation to accelerate the convergence of the Jacobi method. Indeed, we expect such an approach to be effective in the context of other stationary iterative methods, e.g., Richardson iteration, Gauss-Seidel, and Successive Over Relaxation (SOR). Notably, when the proposed approach is developed in the context of the Richardson iteration, the resulting technique—which we shall refer to as the Alternating Anderson-Richardson (AAR) method—represents a generalization of the approach proposed by Khabaza [19]. In particular, the AAR method will reduce to Khabaza's approach on setting $\omega = 1$, $\beta = 0$, and $p = m + 1$, with the coefficients Γ_k calculated only when the reduction in the residual is smaller than a specified threshold. Most significantly for parallel computing, the AAJ method is more amenable to efficient massively parallel implementation than AJ, due to the reduction in global communication associated with the evaluation of Γ_k in every AJ update.

5. Results and discussion

In this section, we validate the accuracy and effectiveness of the proposed Alternating Anderson-Jacobi (AAJ) approach in a series of test cases, including nonsymmetric, Poisson, and complex-valued Helmholtz problems. Using finite differences, we discretize the partial differential equations in a domain Ω having boundary $\partial\Omega$ with outward unit normal $\mathbf{n}$. We denote the mesh-size by h and the number of nodes in each direction by n_d. For the resulting linear systems of equations, we employ the following nomenclature for presenting results and ensuing discussion. We denote each partial differential equation and its associated boundary conditions by 'Problem #', where '#' represents a number. Next, we associate with every linear system a four character label, in which we abbreviate 'Problem #' in the first two characters as 'P#'. We set the third character as either 'a' or 'b', where 'a' denotes a collection of 'P#' systems having the same Ω with varying h, whereas 'b' denotes same h with varying Ω. We append a number as the fourth character to indicate the value of n_d, with values in ascending order. For example, if $n_d = n_1$, n_2, and n_3 ($n_1 < n_2 < n_3$) are used for discretization, fourth characters of '1', '2', and '3' signify systems with $n_d = n_1$, n_2, and n_3, respectively.

We compare the performance of the AAJ method with the Weighted Jacobi (WJ), Anderson-Jacobi (AJ), and Scheduled Relaxation Jacobi (SRJ) [4] fixed-point approaches. In the AJ and AAJ iterations, we employ the Moore-Penrose pseudoinverse [20] for the calculation of $(\mathbf{F}_k^T\mathbf{F}_k)^{-1}$ since $\mathbf{F}_k^T\mathbf{F}_k$ can become ill-conditioned as the iteration proceeds, for large m in particular. We also compare with the Krylov subspace method GMRES [3], whose efficiency can be significantly enhanced with sophisticated preconditioning schemes such as multigrid [21]. However, such schemes increase the cost per iteration, and pose significant challenges for large-scale parallelization [4]. The aim of the present work is to retain as far as possible the simplicity and computational locality of the classical Jacobi method while substantially accelerating it without need of such advanced preconditioning, thus providing a method well-suited to large-scale parallel implementation. Hence, for the present purposes, we shall compare to GMRES with simple Jacobi preconditioning, using the same inverse diagonal as in the Jacobi iteration. It is worth noting however that, given the relation of Anderson and GMRES iterations [12], if more sophisticated preconditioners are available, they may be expected to benefit AJ and AAJ iterations as well as GMRES.

We perform all calculations using MATLAB [22] on a workstation with the following configuration: Intel Xeon Processor E3-1220 v3 (Quad Core, 3.10GHz Turbo, 8MB), 16GB (2x8GB) 1600MHz DDR3 ECC UDIMM.

5.1. Model problem: Laplace equation

The Laplace equation is among the most well studied partial differential equations, making it an excellent test case. We generate the corresponding linear systems using second-order finite-differences. Since the matrix $\mathbf{D}^{-1}\mathbf{A}$ is independent of the mesh-size h, we only consider systems resulting from varying h with fixed domain Ω. In the Weighted Jacobi (WJ) method, we utilize the optimal relaxation parameter [1] of $\omega = \omega^* = 2/(\lambda_1 + \lambda_N) = 1$, where λ_1 and λ_N are the minimum and maximum eigenvalues of $\mathbf{D}^{-1}\mathbf{A}$. In situations where ω^* results in a non-convergent iteration due to finite precision, we reduce ω^* by 0.01. In the Anderson-Jacobi (AJ) approach, we choose $\{\beta, m\} = \{0.2, 10\}$, which we have found to be efficient after an initial traversal of the two-dimensional parameter space. Since $\mathbf{b} = \mathbf{0}$ for the Laplace equation, it is not possible to use the relative residual for the stopping criterion. Instead, we define the normalized residual in this case as

$$r_{k+1} = \frac{\|\mathbf{f}(\mathbf{x}_k)\|}{\|\mathbf{f}(\mathbf{x}_0)\|}\,, \tag{31}$$

and set $\epsilon = 1 \times 10^{-8}$ as the tolerance for convergence. We employ the same random starting guess while studying the relative performance of different approaches.

5.1.1. One-dimensional Laplace equation

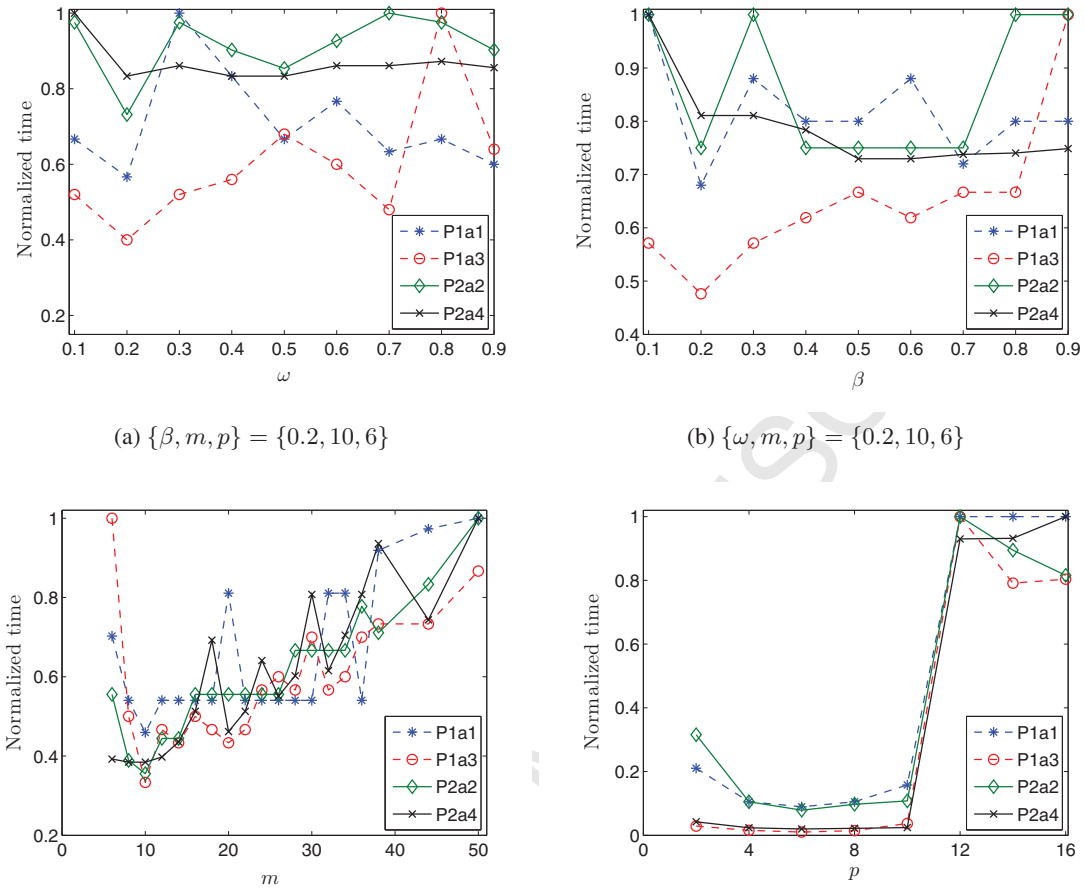
We first consider the one-dimensional Laplace equation with zero Dirichlet and Neumann boundary conditions:

$$\text{Problem 1:} \qquad -V_{xx}(x) = 0 \ \text{ in } \ \Omega\,, \quad V(x) = 0 \ \text{ on } \ \partial\Omega\,, \tag{32}$$

$$\text{Problem 2:} \qquad -V_{xx}(x) = 0 \ \text{ in } \ \Omega\,, \quad V_x(x) = 0 \ \text{ on } \ \partial\Omega\,, \tag{33}$$

where $\Omega = (0, L)$. We choose a domain of size $L = 100$, and discretize it using $n_d = 101$, 301, 1001, 3001 and $10,001$ finite-difference nodes. The matrices $\mathbf{A}$ resulting from the discretization of Problems 1 and 2 are positive-definite and positive-semidefinite respectively. Further, their respective solutions are $\mathbf{x}^* = 0$ and $\mathbf{x}^* = \mathbf{c}$, where $\mathbf{c}$ is any constant vector.

The parameters within the AAJ method are $\{\omega, \beta, m, p\}$, where ω and β are the relaxation parameters in the WJ and AJ updates respectively, $m+1$ is the number of iterates in the Anderson extrapolation history, and p is the frequency of the AJ update. We choose 'P1a1', 'P1a3', 'P2a2', and 'P2a4' as representative systems to perform a parametric study. After a preliminary traversal of the four-dimensional space of parameters, we have found $\{\omega, \beta, m, p\} = \{0.2, 0.2, 10, 6\}$ to be an efficient set. In Fig. 2, we present the normalized computational time taken when three of these parameters are fixed and the fourth one is varied. We observe that the performance of AAJ is relatively insensitive to the choice of ω and β. We find $m \sim 10$ to be optimal, with a steep increase in time for smaller values. We also notice a drastic reduction in performance for $p > 10$, with $p \sim 6$ being optimal. We have made similar observations for the Poisson and Helmholtz equations, with AAJ again relatively insensitive to the choice of parameters. Overall, we find $\{\omega, \beta, m, p\} = \{0.2, 0.2, 10, 6\}$ to perform appreciably, with solution times of 0.02, 0.10, 0.03, and 0.30 seconds for 'P1a1', 'P1a3', 'P2a2', and 'P2a4' systems, respectively. We shall employ this set of parameters within AAJ for the Laplace, Poisson, and Helmholtz equations.



(a) $\{\beta, m, p\} = \{0.2, 10, 6\}$

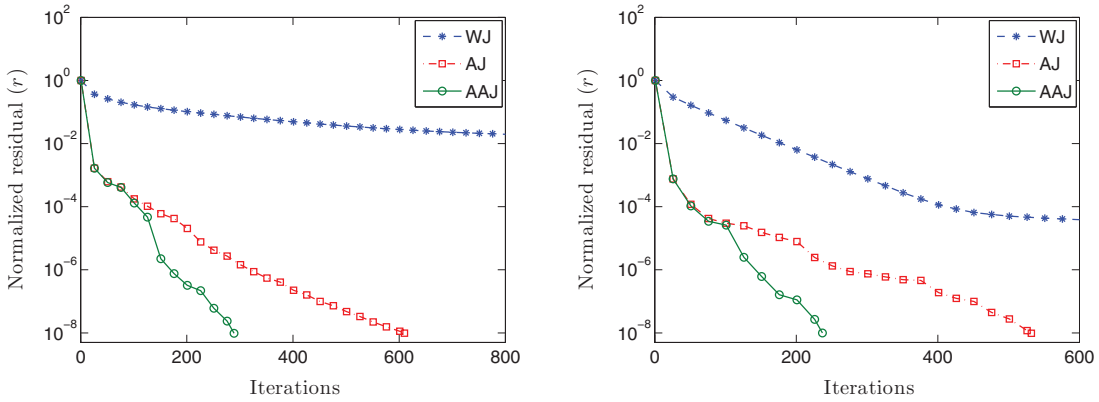
(b) $\{\omega, m, p\} = \{0.2, 10, 6\}$

(c) $\{\omega, \beta, p\} = \{0.2, 0.2, 6\}$

(d) $\{\omega, \beta, m\} = \{0.2, 0.2, 10\}$

Figure 2: Performance of AAJ for different choices of parameters. The computational times within any curve are normalized with respect to the maximum value in that curve. The linear systems are obtained from the discretization of the one-dimensional Laplace equation with zero Dirichlet and Neumann boundary conditions.

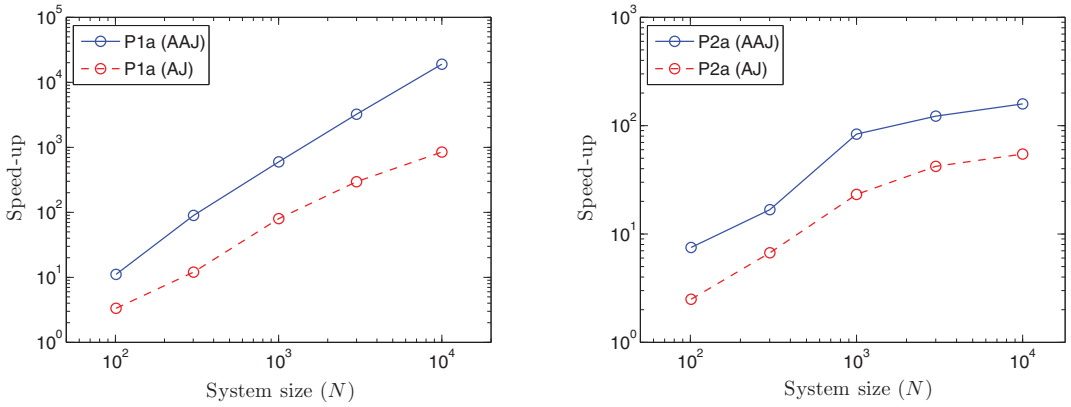
Next, we compare in Fig. 3 the progression of the relative residual during the WJ, AJ, and AAJ fixed-point iterations. We observe that the AAJ method converges extremely rapidly, while maintaining a relatively high rate of convergence throughout the iteration. In fact, AAJ is able to reduce the normalized residual to 1×10^{-8} in 107 and 72 times fewer iterations compared to WJ for the systems 'P1a1' and 'P2a1', respectively. Remarkably, the AAJ technique also requires fewer iterations to achieve a specified tolerance compared to the AJ method. This suggests that iterates produced by WJ updates are better suited for Anderson extrapolation than those produced by AJ updates. In practice, we find that the WJ iterations effectively reduce 'higher-frequency' components of the error while Anderson extrapolations effectively reduce 'lower-frequency' components, yielding a combined method effective at reducing both.



(a) P1a1 (b) P2a1

Figure 3: Comparison of the convergence of the WJ, AJ, and AAJ methods. The linear systems are obtained from the discretization of the one-dimensional Laplace equation with zero Dirichlet and Neumann boundary conditions.

Finally, we present in Fig. 4 the speed-ups of the AJ and AAJ methods relative to WJ as a function of system size. It is clear that both AJ and AAJ are able to significantly accelerate the convergence of the WJ method. In fact, AAJ is able to achieve staggering speed-ups in excess of $19,000$ and 100 for the largest systems in the Dirichlet and Neumann problems, respectively. At the same time, AAJ is able to accelerate AJ by up to an order of magnitude. Notably, the trends in the plots indicate that even larger speed-ups of AJ and AAJ over WJ are expected as n_d is increased. Overall, we conclude that AAJ is not only able to tremendously accelerate the WJ method, but also able to noticeably outperform AJ as well.



(a) Problem 1 (b) Problem 2

Figure 4: Speed-up of AJ and AAJ methods relative to WJ. The linear systems are obtained from the discretization of the one-dimensional Laplace equation with zero Dirichlet and Neumann boundary conditions.

5.1.2. Two-dimensional Laplace equation

The WJ method presented in Section 2 employs a constant relaxation parameter ω. However, this condition can be relaxed to accelerate the WJ method, as demonstrated by the recently developed Scheduled

Relaxation Jacobi (SRJ) method [4]. In order to facilitate comparison with SRJ, we consider the two-dimensional Laplace equation with zero Neumann boundary conditions:

$$\text{Problem 3:} \qquad -V_{xx}(x,y) - V_{yy}(x,y) = 0 \ \text{ in } \ \Omega\,, \quad \frac{\partial V(x,y)}{\partial \mathbf{n}} = 0 \ \text{ on } \ \partial\Omega\,, \tag{34}$$

where $\Omega \in \mathbb{R}^2$ is a square with side L. Specifically, we choose $L = 100$ and $n_d = 32, 64, 128,$ and 256 finite-difference nodes in each direction. The resulting systems are symmetric positive-semidefinite with solution $\mathbf{x}^* = \mathbf{c}$.

In Fig. 5, we compare the residual as a function of iteration number for the WJ, AJ, AAJ, and SRJ methods. On one hand, SRJ demonstrates significantly larger asymptotic convergence rates compared to WJ. Therefore, in situations where high accuracies are desired, the SRJ method may be expected to significantly outperform the WJ method. On the other hand, WJ quickly reduces the initial residual compared to SRJ, which follows from its ability to rapidly nullify the 'high frequency' components of the error/residual. We find both AJ and AAJ methods require fewer iterations than SRJ, with AAJ demonstrating the most rapid convergence of all.

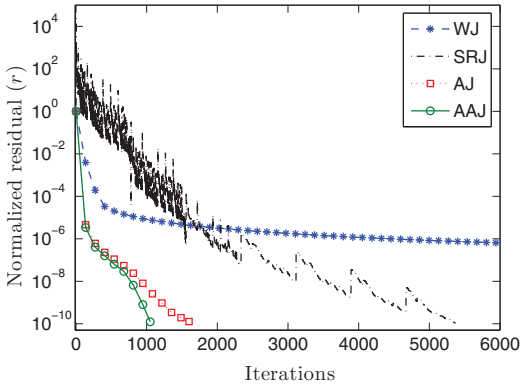


Figure 5: Convergence of WJ, AJ, AAJ, and SRJ methods for the 'P3a4' system. The linear systems are obtained from discretization of the two-dimensional Laplace equation with zero Neumann boundary conditions.

In Table 1, we compare the ability of the AAJ and SRJ methods to accelerate the WJ method. We observe that AAJ demonstrates larger speed-ups compared to SRJ. Most notably, the performance of AAJ relative to SRJ improves with size of the system.

	P3a1	P3a2	P3a3	P3a4
AAJ	4.0	14.3	44.1	61.0
SRJ	5.0	10.8	21.1	26.3

Table 1: Speed-up of AAJ and SRJ methods relative to WJ. The linear systems are obtained by discretization of the two-dimensional Laplace equation with zero Neumann boundary conditions.

5.2. Electronic structure calculations: Poisson and Helmholtz equations

We now focus on linear systems arising from the discretization of partial differential equations arising in electronic structure calculations. Specifically, we consider the Poisson and complex-valued Helmholtz equations discretized using sixth-order finite differences. As for the Laplace problems, we employ the

optimal relaxation parameter ω^* for WJ, $\{\beta, m\} = \{0.2, 10\}$ for AJ, and $\{\omega, \beta, m, p\} = \{0.2, 0.2, 10, 6\}$ for AAJ. When comparing with the Krylov subspace method GMRES (restarted every 30 iterations), we calculate the relative residual within the WJ, AJ, and AAJ methods using the relation

$$r_{k+1} = \frac{\|\mathbf{A}\mathbf{x}_k - \mathbf{b}\|}{\|\mathbf{b}\|} = \frac{\|\mathbf{D}^{-1}\mathbf{A}\,\mathbf{x}_k - \mathbf{D}^{-1}\mathbf{b}\|}{\|\mathbf{D}^{-1}\mathbf{b}\|} = \frac{\|\mathbf{f}(\mathbf{x}_k)\|}{\|\mathbf{D}^{-1}\mathbf{b}\|}\,. \tag{35}$$

Above, the second equality follows from the use of the finite-difference approximation with a uniform mesh, whereby the diagonal elements of $\mathbf{A}$ (and therefore $\mathbf{D}$) have the same value. Another implication of this property is that the performance of GMRES for the systems $\mathbf{D}^{-1}\mathbf{A}\mathbf{x} = \mathbf{D}^{-1}\mathbf{b}$ and $\mathbf{A}\mathbf{x} = \mathbf{b}$ is identical for the discretized problems considered in this section. Unless specified otherwise, we utilize a vector of all ones as the starting guess $\mathbf{x}_0$ and $\epsilon = 1 \times 10^{-8}$ as the tolerance for convergence.

5.2.1. Poisson equation

We now consider the three-dimensional non-periodic and periodic Poisson equations arising in real-space Density Functional Theory (DFT) [23, 24, 25, 26] and orbital-free Density Functional Theory (OF-DFT) [27, 28] simulations:

$$\text{Problem 4:}\quad -\frac{1}{4\pi}\nabla^2 V(\mathbf{r}) = \rho(\mathbf{r}) + b(\mathbf{r}) \;\text{ in }\; \Omega, \quad \Big\{ V(\mathbf{r}) = 0 \;\text{ on }\; \partial\Omega\,, \tag{36}$$

$$\text{Problem 5:}\quad -\frac{1}{4\pi}\nabla^2 V(\mathbf{r}) = \rho(\mathbf{r}) + b(\mathbf{r}) \;\text{ in }\; \Omega, \quad \begin{cases} V(\mathbf{r}) = V(\mathbf{r} + L\hat{\mathbf{e}}_i) \;\text{ on }\; \partial\Omega\,, \\ -\hat{\mathbf{e}}_i \cdot \nabla V(\mathbf{r}) = \hat{\mathbf{e}}_i \cdot \nabla V(\mathbf{r} + L\hat{\mathbf{e}}_i) \;\text{ on }\; \partial\Omega\,. \end{cases} \tag{37}$$

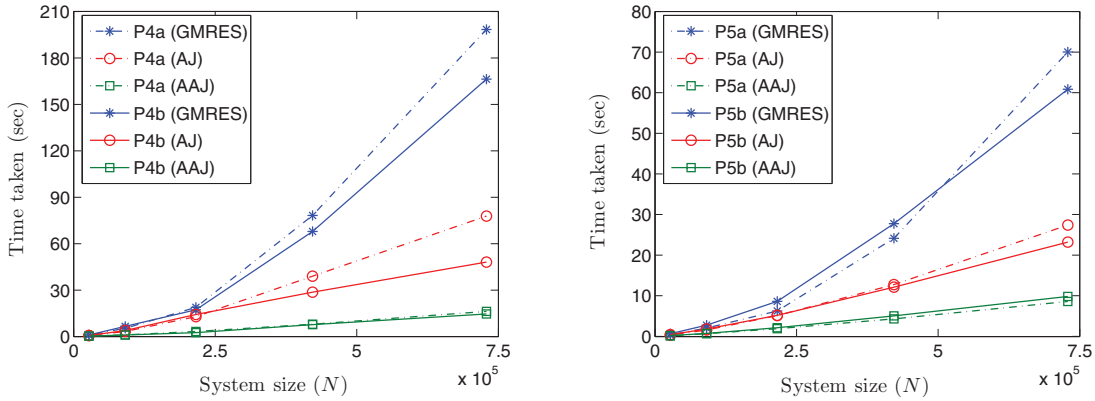
Above, $\Omega \in \mathbb{R}^3$ is a cubic domain of side L and $\hat{\mathbf{e}}_i$ are the unit vectors aligned with the edges of Ω. The fields $\rho(\mathbf{r})$ and $b(\mathbf{r})$ denote the electron density and nuclear density, respectively. $\rho(\mathbf{r})$ is calculated by superimposing isolated-atom electron densities. Similarly, $b(\mathbf{r})$ is evaluated by superimposing the charge densities calculated from the highest occupied angular momentum component of the Troullier-Martins pseudopotential [29] using the finite-difference approximation [30, 28]. In Table 2, we present the nomenclature and details for the various systems of equations corresponding to the aforementioned problems.

Discretization parameters	Nodes in each direction (n_d)				
	30	45	60	75	90
$L = 28.50$ Bohr	P4a1	P4a2	P4a3	P4a4	P4a5
$h = 0.98$ Bohr	P4b1	P4b2	P4b3	P4b4	P4b5
$L = 10.26$ Bohr	P5a1	P5a2	P5a3	P5a4	P5a5
$h = 0.68$ Bohr	P5b1	P5b2	P5b3	P5b4	P5b5

Table 2: Nomenclature for the different linear systems arising from the discretization of the three-dimensional non-periodic and periodic Poisson equations. 'P4a' corresponds to a Si_5H_{12} cluster with varying h, whereas 'P4b1', 'P4b2', 'P4b3', 'P4b4', and 'P4b5' correspond to Si_5H_{12}, $Si_{17}H_{36}$, $Si_{87}H_{76}$, $Si_{275}H_{172}$, and $Si_{525}H_{276}$ clusters, respectively. 'P5a' denotes varying h for a single diamond cubic unit cell of Silicon, whereas 'P5b1', 'P5b2', 'P5b3', 'P5b4', and 'P5b5' correspond to 2, 3, 4, 5, and 6 diamond cubic unit cells of Silicon in each direction with a vacancy. The lattice constant of diamond cubic Silicon is chosen to be 10.26 Bohr.

In Fig. 6, we compare the performance of the AJ, AAJ, and GMRES methods by plotting the time taken as a function of system size. We observe that both AAJ and AJ are able to outperform GMRES, with AAJ comfortably demonstrating the best timings. In particular, AAJ exhibits close to linear scaling with system size, making it an attractive technique for solving large systems of equations. Notably, AAJ achieves a

speed-up of nearly an order of magnitude over GMRES for systems of size $N = 729,000$, with the speed-up increasing as the system gets larger. This is indeed verified by the results in Table 3. Significantly, for the 'P4b7' system, AAJ is faster than GMRES and WJ by factors in excess of 20 and 100, respectively.



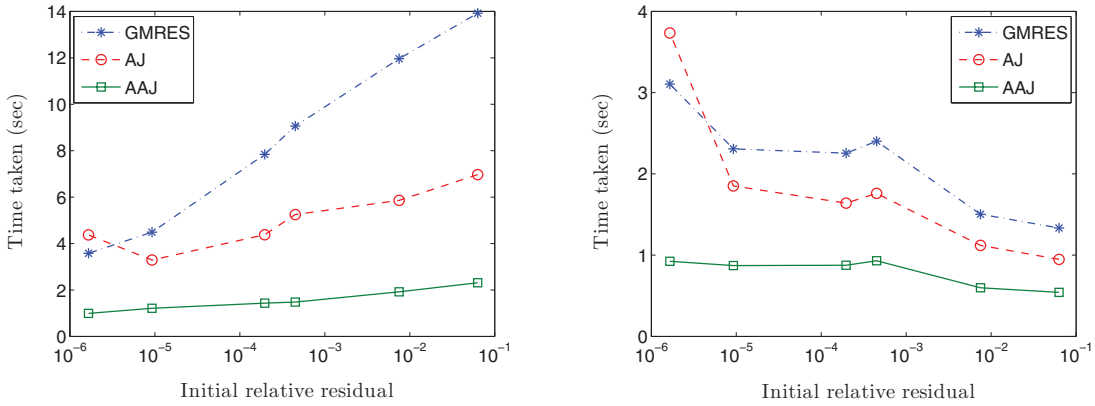
(a) Problem 4 (b) Problem 5

Figure 6: Performance of AJ, AAJ, and GMRES methods for linear systems obtained from the discretization of the three-dimensional non-periodic and periodic Poisson equations.

Method	Problem 4		Problem 5	
	P4b6	P4b7	P5b6	P5b7
WJ	3275.10	10030.40	1342.02	4157.14
AJ	331.63	766.45	87.61	227.52
AAJ	39.60	97.67	34.18	85.03
GMRES	682.93	2038.26	311.09	908.75

Table 3: Computational time in seconds taken by the WJ, AJ, AAJ, and GMRES approaches. 'P4b6' and 'P4b7' correspond to the $Si_{525}H_{276}$ cluster whereas 'P5b6' and 'P5b7' correspond to 6 diamond cubic unit cells of Silicon in each direction with a vacancy. In the label, the last characters of 6 and 7 correspond to $n_d = 120$ and $n_d = 150$, respectively. The linear systems are obtained from the discretization of the three-dimensional non-periodic and periodic Poisson equations.

Next, we study the influence of the quality of the initial guess on the performance of AAJ, for which we choose 'P4a3' as the representative example. Using an in-house code, we perform DFT calculations using the Anderson mixing accelerated SCF method, wherein Eqn. 36 is solved once every SCF iteration. As the iteration progresses towards convergence, the quality of the guess improves by virtue of using the previous step's solution. In Fig. 7, we compare the performance of the AAJ method with the AJ and GMRES techniques for different initial relative residuals. Specifically, we plot in Fig. 7a the time taken for the relative residual to reach $\epsilon = 1 \times 10^{-8}$. We also plot in Fig. 7b the time taken to reduce the relative residual by a factor of 1×10^{-2}. We find that AAJ significantly outperforms AJ and GMRES irrespective of the quality of the initial guess. In particular, the performance of AAJ is relatively independent of the nature of the initial guess. Interestingly, the time taken by GMRES and AJ to reduce the relative residual by two orders of magnitude increases as the initial guess gets closer to the converged solution. Based on these observations, we can surmise that the efficiency of AAJ can be partly attributed to its enhanced performance as the relative residual becomes smaller during the linear solve.



(a) Convergence to a tolerance of $\epsilon = 1 \times 10^{-8}$.

(b) Initial relative residual reduction by a factor of 0.01.

Figure 7: Performance of AJ, AAJ, and GMRES methods as a function of the quality of the initial guess. The linear system under consideration is 'P4a3', obtained by the discretization of the three-dimensional non-periodic Poisson equation.

5.2.2. Helmholtz equation

Next, we consider the Helmholtz equation arising in periodic real-space OFDFT calculations [31, 27]:

$$\text{Problem 6: } -\frac{1}{4\pi}\nabla^2 V(\mathbf{r}) + Q\,V(\mathbf{r}) = P\,\rho^{\alpha}(\mathbf{r}) \;\text{ in }\; \Omega, \quad \begin{cases} V(\mathbf{r}) = V(\mathbf{r} + L\hat{\mathbf{e}}_i) \;\text{ on }\; \partial\Omega\,, \\ -\hat{\mathbf{e}}_i \cdot \nabla V(\mathbf{r}) = \hat{\mathbf{e}}_i \cdot \nabla V(\mathbf{r} + L\hat{\mathbf{e}}_i) \;\text{ on }\; \partial\Omega\,, \end{cases} \tag{38}$$

where $\Omega \in \mathbb{R}^3$ is a cubic domain of side L. The constants $\alpha = \frac{5}{6} + \frac{\sqrt{5}}{6}$, $P = 0.0296 + i\,0.0217$, and $Q = -0.1284 - i\,0.1269$. The resulting matrices $\mathbf{A}$ are complex-symmetric non-Hermitian. As before, the electron density $\rho(\mathbf{r})$ is evaluated by superimposing isolated-atom electron densities. In Table 4, we present the nomenclature for the resulting systems of equations.

Discretization parameters	Nodes in each direction (n_d)				
	30	45	60	75	90
$L = 7.65$ Bohr	P6a1	P6a2	P6a3	P6a4	P6a5
$h = 0.51$ Bohr	P6b1	P6b2	P6b3	P6b4	P6b5

Table 4: Nomenclature for the different linear systems of equations arising from the discretization of the three-dimensional periodic Helmholtz equation. 'P6a' denotes a single face centered cubic (FCC) unit cell of Aluminum with varying h, whereas 'P6b1', 'P6b2', 'P6b3', 'P6b4', and 'P6b5' correspond to 2, 3, 4, 5, and 6 FCC unit cells of Aluminum in each direction, with a vacancy. The lattice constant of FCC Aluminum is chosen to be 7.65 Bohr.

In Fig. 8, we compare the performance of the AJ, AAJ, and GMRES methods for the aforedescribed linear systems of equations. It is clear that AJ and AAJ are again able to outperform GMRES, with AAJ demonstrating the best performance. Furthermore, AAJ is able to achieve close to linear scaling with system size, and therefore its performance relative to AJ and GMRES increases for larger systems. In particular, AAJ demonstrates nearly an order of magnitude speed-up over GMRES for the 'P6a5' system. It is worth noting that unlike GMRES and AJ, which show large differences in solution times for fixed-domain and fixed-mesh cases, AAJ has nearly identical performance. Overall, we conclude that AAJ represents a highly efficient method compared to Krylov subspace methods like GMRES, even for complex non-Hermitian linear systems of equations.

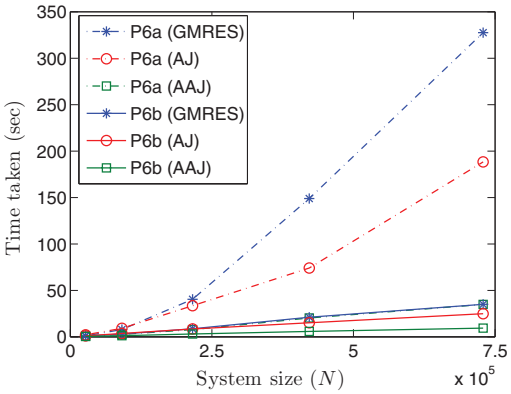


Figure 8: Performance of AJ, AAJ, and GMRES methods for the linear systems obtained from the discretization of the three-dimensional periodic Helmholtz equation.

The superior performance of AJ/AAJ compared to GMRES merits further consideration. Notably, when Anderson's extrapolation with complete history ($m = \infty$) is applied to the Richardson iteration, it is equivalent to GMRES without restart in exact arithmetic [12]. This is because GMRES and Anderson's method utilize the same Krylov subspace—albeit with a different parametrization—within which the residual is minimized. In numerical computations, non-restarted Jacobi preconditioned GMRES is expected to perform favorably compared to complete-history AJ/AAJ since linear dependency within the Krylov subspace is prevented through orthogonalization. However, restarted GMRES is almost always employed in practice to reduce orthogonalization and storage costs. Similarly, finite mixing histories are essential to the performance of AJ/AAJ. A significant difference between the GMRES restart and the finite-history AJ/AAJ used in practice is that the restart in GMRES starts the approach afresh, while the AJ/AAJ methods retain a constant mixing history size. Leveraging more such history information at each iteration may contribute to the superior performance of AJ/AAJ over GMRES in practice. A more complete understanding of why AJ is able to outperform GMRES and why AAJ is able to outperform AJ is a worthy subject of further research, which the authors are pursuing presently.

5.3. Matrix Market: Nonsymmetric matrices

Finally, we demonstrate the generality of AAJ by considering nonsymmetric linear systems obtained via finite-element discretizations. Specifically, we consider the FIDAPM series of matrices in the Matrix Market[1] repository. In Table 5, we present the computational time taken by AAJ and GMRES for three of these systems. Within AAJ, we choose two values of the relaxation parameter $\beta = \omega$, while retaining $\{m, p\} = \{10, 6\}$ as in all the previous examples. We compare the results so obtained with GMRES for two choices of restarts: 30 and 750 iterations. In order to ensure a fair comparison, we use Jacobi preconditioning with GMRES, i.e., we solve the system $\mathbf{D}^{-1}\mathbf{A}\mathbf{x} = \mathbf{D}^{-1}\mathbf{b}$, using the same $\mathbf{D}^{-1}$ as in the Jacobi iteration. The tolerance for the relative residual is set to $\epsilon = 1 \times 10^{-8}$ and a vector of all ones is used as the starting guess $\mathbf{x}_0$. We observe that AAJ is able to outperform GMRES for these nonsymmetric finite-element matrices and choice of parameters. Overall, while applicable to nonsymmetric systems with a variety of spectra, we find that AAJ is generally less efficient for systems wherein the smallest real part of the eigenvalues of $\mathbf{D}^{-1}\mathbf{A}$ are negative, as may be expected given its Jacobi aspect. As an example, for the

[1] http://math.nist.gov/MatrixMarket/

'utm300' system in the TOKAMAK collection, GMRES with restart of 150 is factor of 1.2 faster than AAJ with $\{\omega, \beta, m, p\} = \{0.3, 0.3, 150, 6\}$.

Matrix	N	AAJ		GMRES for $\mathbf{D}^{-1}\mathbf{A}\,\mathbf{x} = \mathbf{D}^{-1}\mathbf{b}$	
		$\beta = 0.3$	$\beta = 0.4$	$restart = 30$	$restart = 750$
fidap008	3096	315.05	152.72	-	877.14
fidap029	2870	0.009	0.007	0.014	0.014
fidapm37	9152	91.76	26.40	-	233.89

Table 5: Time taken in seconds by AAJ and GMRES for linear systems from Matrix Market. In the table, '-' indicates that convergence was not achieved within 1000 sec.

6. Concluding remarks

We have employed Anderson extrapolation to accelerate the classical Jacobi iterative method for the solution of large, sparse linear systems. The resulting Alternating Anderson-Jacobi (AAJ) method combines weighted Jacobi iterations with periodic Anderson extrapolations to produce a technique which retains as far as possible the essential simplicity and computational locality of the classical Jacobi method while accelerating convergence substantially. The efficacy of the method was demonstrated in a series of test cases, including nonsymmetric systems as well as 3D Poisson and complex-valued Helmholtz problems arising in electronic structure calculations. The proposed approach was shown to scale very favorably with system size, and possess a small prefactor, even in the absence of a preconditioner. In particular, AAJ was not only able to accelerate the classical Jacobi iteration by factors in excess of 10,000, but also substantially outperform GMRES in the problems considered, with relative speed-up increasing with system size. The simplicity and locality of the AAJ method make it an attractive alternative for solving large, sparse linear systems on massively parallel computers, which is currently being pursued by the authors. Additional mathematical analysis which provides further insights into the performance of the method is also a worthy subject for future research.

7. Acknowledgements

This work was performed, in part, under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07-NA27344 and the Exascale Co-design Center for Materials in Extreme Environments supported by Office of Science Advanced Scientific Computing Research Program. The authors also gratefully acknowledge the support of National Science Foundation under Grant Number 1333500.

References

[1] Y. Saad, Iterative Methods for Sparse Linear System (2nd ed), SIAM, 2003.

[2] J. R. Shewchuk, An introduction to the conjugate gradient method without the agonizing pain, 1994.

[3] Y. Saad, M. H. Schultz, SIAM Journal on scientific and statistical computing 7 (1986) 856–869.

[4] X. I. Yang, R. Mittal, Journal of Computational Physics 274 (2014) 695 – 708.

[5] D. G. Anderson, Journal of the Association for Computing Machinery 12 (1965) 547–560.

[6] E. Cancès, C. Le Bris, International Journal of Quantum Chemistry 79 (2000) 82–90.

[7] H.-r. Fang, Y. Saad, Numer. Linear Algebra Appl. 16 (2009) 197–221.

[8] L. Lin, C. Yang, SIAM Journal on Scientific Computing 35 (2013) S277–S298.

[9] P. Pulay, Chemical Physics Letters 73 (1980) 393–398.

[10] V. Ganine, N. Hills, B. Lapworth, International Journal for Numerical Methods in Fluids 71 (2013) 939–959.

[11] J. Willert, W. T. Taitano, D. Knoll, Journal of Computational Physics 273 (2014) 278–286.

[12] T. Rohwedder, R. Schneider, Journal of mathematical chemistry 49 (2011) 1889–1914.

[13] H. F. Walker, P. Ni, SIAM Journal on Numerical Analysis 49 (2011) 1715–1735.

[14] F. A. Potra, H. Engler, Linear Algebra and its Applications 438 (2013) 1002–1011.

[15] A. Quarteroni, R. Sacco, F. Saleri, Numerical mathematics, volume 37, Springer, 2007.

[16] A. Greenbaum, Iterative methods for solving linear systems, volume 17, Siam, 1997.

[17] E. Weinan, Principles of multiscale modeling, Cambridge University Press, 2011.

[18] L. Marks, D. Luke, Physical Review B 78 (2008) 075114.

[19] I. Khabaza, The Computer Journal 6 (1963) 202–206.

[20] P. Courrieu, arXiv preprint arXiv:0804.4809 (2008).

[21] W. Hackbusch, Multi-grid methods and applications, volume 4, Springer Science & Business Media, 2013.

[22] M. U. Guide, Inc., Natick, MA 5 (1998).

[23] J. E. Pask, P. A. Sterne, Phys. Rev. B 71 (2005) 113101.

[24] P. Suryanarayana, V. Gavini, T. Blesgen, K. Bhattacharya, M. Ortiz, Journal of the Mechanics and Physics of Solids 58 (2010) 256 – 280.

[25] P. Suryanarayana, K. Bhattacharya, M. Ortiz, Journal of Computational Physics 230 (2011) 5226 – 5238.

[26] J. E. Pask, N. Sukumar, S. E. Mousavi, International Journal for Multiscale Computational Engineering 10 (2012) 83–99.

[27] S. Ghosh, P. Suryanarayana, arXiv preprint arXiv:1412.8250 (2014).

[28] P. Suryanarayana, D. Phanish, Journal of Computational Physics 275 (2014) 524 – 538.

[29] N. Troullier, J. L. Martins, Physical Review B 43 (1991) 1993.

[30] P. Suryanarayana, K. Bhattacharya, M. Ortiz, Journal of the Mechanics and Physics of Solids 61 (2013) 38–60.

[31] N. Choly, E. Kaxiras, Solid State Communications 121 (2002) 281 – 286.