A Variant of Anderson Mixing with Minimal Memory Size

Anonymous Author(s)
Affiliation
Address
email

Abstract

Anderson mixing (AM) is an acceleration method for fixed-point problems by exploring the information from historical iterations. Despite its numerical success in various applications, the memory requirement in AM remains a bottleneck when solving large-scale optimization problems in a resource-limited machine. To address this problem, we propose a novel variant of AM method, called Min-AM, by storing only one vector pair, that is the minimal memory size requirement in AM. Our method forms a symmetric approximation to the inverse Hessian matrix and is proved to be equivalent to the full-memory Type-I AM for solving strongly convex quadratic optimization. Moreover, for general nonlinear optimization problems, we establish the convergence properties of Min-AM under reasonable assumptions and show that the mixing parameters can be adaptively chosen by estimating the eigenvalues of the Hessian. Finally, we extend Min-AM to solve stochastic programming problems. Experimental results on logistic regression and network training problems validate the effectiveness of the proposed Min-AM.

1 Introduction

Anderson mixing (AM) [2] is a powerful method for accelerating fixed-point iterations, and has been widely used in scientific computing [1, 5] and machine learning [24, 49]. In each iteration, AM extrapolates a new iterate that satisfies certain optimality property by using the historical iterations, and exhibits empirical acceleration over classical fixed-point iterations [51]. Thus, AM can reduce the computation cost when evaluating the fixed-point map is a time consuming process. Also, the inspiring connections of AM with GMRES [51, 40] and multisecant quasi-Newton methods [18] provide certain theoretical properties for the acceleration benefit from AM. In recent years, there have been many works applying AM to solve various optimization problems [47, 53, 33]. Their promising results in different tasks suggest that these AM-based methods are very competitive with first-order methods while being more economical than second-order methods. However, as commented by the recent review [3], the existing convergence analysis of AM is still unsatisfactory and it is worth noting that AM does not converge in general [33]. Consequently, exploring theoretical properties of AM and its variants and their applications for machine learning problems deserves deep research.

One major concern of AM is the heavier memory overhead compared with the fixed-point iteration. In AM(m), it has to store vector pairs about historical information from the m previous iterations, where m is the memory size. Specifically, AM(m) is called as the limited-memory AM if $m < \infty$, and full-memory AM if $m = \infty$. The AM(0) is the simple fixed-point iteration. In practice, choosing proper memory size m is heuristic and it is observed in [39] that a small m can deteriorate the efficacy. However, for practical applications, a large m can lead to memory issue for solving large-scale and high-dimensional problems when the memory resource is limited.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In this paper, we address the memory issue of AM for solving optimization problems by proposing Min-AM, a novel variant of AM(1) that has minimal memory size in AM. Min-AM is built upon the Type-I AM [18, 51], a variant of AM. Let AM-I(m) denote the Type-I AM with memory size m. Using a projection-mixing framework of AM-I, the Min-AM retains a recursively modified vector pair and adds one extra projection step in each step of AM-I(1). Compared to AM(1), Min-AM forms symmetric approximations to the Hessian and is equivalent to AM-I(∞) when the objective function is strongly convex quadratic. By incorporating the restarting strategy, the restarted Min-AM has faster convergence rate than AM(1) for general nonlinear optimization and the spectrum of Hessian can be cheaply estimated that leads to adaptive choices of the mixing parameters. Min-AM is also extended to stochastic optimization. In summary, we highlight the main contributions of this work as follows.

- We propose Min-AM, a variant of AM(1) that achieves the minimal memory size in AM, for solving optimization problems. Compared to AM(1), Min-AM forms symmetric approximations to the Hessian and incorporates more historical information to obtain the update. Thus, Min-AM can significantly reduce the memory requirement and computational cost of AM while maintaining the fast convergence property.
- We show that the local convergence rate of the restarted Min-AM has optimal dependence on the condition number of the objective function. We provide adaptive choices of the mixing parameters by estimating the spectrum of the Hessian economically in each iteration. Also, we give the convergence and complexity analysis of a stochastic extension of Min-AM for solving nonconvex stochastic optimization.
- We verify the properties of Min-AM, and apply the restarted Min-AM to logistic regression and the stochastic Min-AM to train neural networks. The experimental results are consistent with our theoretical analysis and show that the Min-AM is competitive with the limited-memory quasi-Newton methods with large memory size in deterministic optimization and achieves promising results in training neural networks.

Related work. Fixed-point problems can be recast as solving systems of (non)linear equations. AM is a practical alternate for Newton's method when the Jacobian is unavailable or difficult to compute [28]. In the linear case, it is established in [51, 40] that the full-memory AM is equivalent to GMRES [43], a classical Krylov subspace method for linear systems. In the nonlinear case, AM is recognized as a multisecant quasi-Newton method and a variant, called Type-I AM, has been proposed in [18]. From the perspective of convergence analysis, it is proved that the local convergence rate of the limited-memory AM is no worse than that of fixed-point iteration [50, 12] and the potential improvement depends on the quality of extrapolation [17, 38]. However, a counter example [33] shows the divergence behaviour of AM(1) and the recent review [3] points out the need of further research on the theoretical properties of AM.

The inspiring performance and the easy implementation of AM motivate many new variants of AM for various applications in machine learning. In [46], a regularized variant of AM is proposed to accelerate gradient descent for unconstrained optimization, which is adapted in [19] for nonsmooth convex optimization and in [49] for reinforcement learning. In [53], a stochastic version of AM is developed and has convergence guarantee for stochastic optimization. The Type-I AM is modified in [58] to solve nonsmooth fixed-point problems. A more recent work, the short-term recurrence AM (ST-AM) [54], stores two vector pairs to reduce the memory cost of AM. Our Min-AM method achieves the minimal memory size, i.e. $m = 1$, further reducing the memory size of ST-AM.

The memory overhead is a common issue of quasi-Newton methods. It is desirable to keep memory size as small as possible without sacrificing the efficacy of the algorithm. Some works [30, 8, 7] study the choice of memory size in L-BFGS [32], but the performance with minimal memory size is unsatisfactory. In stochastic programming, the sublinear convergence rate is optimal when only stochastic gradients can be accessed [35]. Thus the heavier memory and computational cost makes quasi-Newton methods [10, 21, 52, 9] less appealing than the more economical first-order methods [41, 29]. In this work, our proposed Min-AM method exhibits the fast convergence as quasi-Newton methods while the memory cost is close to first-order methods.

Notations. Let Δ denote the forward difference operator, e.g. $\Delta x_k = x_{k+1} - x_k$. $\mathrm{range}(X)$ denotes the subspace spanned by the columns of a matrix X. For every matrix $A \in \mathbb{R}^{d\times d}$, the Krylov subspace $\mathcal{K}_m(A, v) \equiv \mathrm{span}\{v, Av, \ldots, A^{m-1}v\}$, and $\|x\|_A = (x^{\mathrm{T}}Ax)^{1/2}$ denotes the A-norm when A is symmetric positive definite (SPD). "$\dagger$" is the Penrose-Moore inverse.

2 The Anderson mixing scheme

In this work, we consider the minimization problem

$$\min f(x),\ x \in \mathbb{R}^d, \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is continously differentiable. The first-order optimality condition of (1) is equivalent to $x = g(x)$, where $g(x) = x - \nabla f(x)$. We describe the Anderson mixing scheme with the projection-mixing framework [53]. Define $r_k = g(x_k) - x_k = -\nabla f(x_k)$ to be the residual. AM(m) maintains m ($m \leq \min\{d, k\}$) vector pairs stored in $X_k, R_k \in \mathbb{R}^{d\times m}$:

$$X_k = [\Delta x_{k-m}, \Delta x_{k-m+1}, \cdots, \Delta x_{k-1}],\ \ R_k = [\Delta r_{k-m}, \Delta r_{k-m+1}, \cdots, \Delta r_{k-1}]. \tag{2}$$

Each update can be decoupled into two steps, namely the *projection step* and the *mixing step*:

$$\bar{x}_k = x_k - X_k\Gamma_k, \quad \text{(Projection step)},\ \ x_{k+1} = \bar{x}_k + \beta_k\bar{r}_k, \quad \text{(Mixing step)}, \tag{3}$$

where $\bar{r}_k := r_k - R_k\Gamma_k$, $\beta_k > 0$ is the mixing parameter, and $\Gamma_k = \arg\min_{\Gamma\in\mathbb{R}^m} \|r_k - R_k\Gamma\|_2$. It is easy to know that the Galerkin's projection condition $\bar{r}_k \perp \mathrm{range}(R_k)$ holds. Assuming $R_k^{\mathrm{T}}R_k$ is nonsingular, the resulting one-step update of AM(m) is $x_{k+1} = x_k + H_k r_k$ where

$$H_k = \beta_k I - (X_k + \beta_k R_k)(R_k^{\mathrm{T}}R_k)^{-1}R_k^{\mathrm{T}}. \tag{4}$$

The AM-I [18, 51] falls into the same projection-mixing framework, and the only difference lies in the choice of Γ_k. The Γ_k of AM-I is determined by another Galerkin's condition:

$$\bar{r}_k^{\mathrm{T}}X_k = (r_k - R_k\Gamma_k)^{\mathrm{T}}X_k = 0. \tag{5}$$

Substituting the solution of (5) into (3), the approximated inverse Hessian matrix H_k of AM-I is

$$H_k = \beta_k I - (X_k + \beta_k R_k)(X_k^{\mathrm{T}}R_k)^{-1}X_k^{\mathrm{T}}, \tag{6}$$

by assuming $X_k^{\mathrm{T}}R_k$ is nonsingular. AM-I has close relation to Newton's method and an alternative interpretation of AM-I is given in Appendix A.1.

Remark 1. *It is established in [18] that the H_k of AM solves $\min_{H_k} \|H_k - \beta_k I\|_F$ s.t. $H_kR_k = -X_k$ and the H_k of AM-I satisfies $H_k = B_k^{-1}$, where B_k solves $\min_{B_k} \|B_k - \beta_k^{-1}I\|_F$ s.t. $B_kX_k = -R_k$. Thus, both AM and AM-I are multisecant methods but the approximated inverse Hessian matrices H_k defined in* (4) *and* (6) *are generally not symmetric.*

3 The Min-AM methods

In each iteration, AM(m) have to store two matrices $X_k, R_k \in \mathbb{R}^{d\times m}$, which dramatically increase the memory burden in large-scale problems. To reduce the memory requirement, we consider the minimal memory case, i.e. $m = 1$. The proposed Min-AM is a variant of AM(1) and the convergence of Min-AM is established in this section.

3.1 The basic Min-AM

In this section, we consider the simple case of (1): the objective function is $f(x) = \frac{1}{2}x^\top Ax - b^{\mathrm{T}}x$ where A is SPD. Let $s_k := \Delta x_{k-1}, y_k := \Delta r_{k-1}$ and choose $m = 1$ in (4). The approximated inverse Hessian of AM(1) is $H_k = \beta_k I - (s_k + \beta_k y_k)(y_k^{\mathrm{T}}y_k)^{-1}y_k^{\mathrm{T}}$, which is non-symmetric. In contrast, the Hessian matrix of f defined in (1) is naturally symmetric. To achieve better approximation of $\nabla^2 f$, we first recursively modify the vector pair s_k, y_k to incorporate more historical information. Let $p_1 = \Delta x_0, q_1 = \Delta r_0 \in \mathbb{R}^d$, we construct p_k and q_k by

$$p_k = \Delta x_{k-1} - p_{k-1}\zeta_k, \quad q_k = \Delta r_{k-1} - q_{k-1}\zeta_k, \tag{7}$$

where $\zeta_k = (p_{k-1}^{\mathrm{T}}q_{k-1})^{-1}\Delta r_{k-1}^{\mathrm{T}}p_{k-1}$, assuming $p_{k-1}^{\mathrm{T}}q_{k-1} \neq 0$. Moreover, we add an extra projection step to (3) and the resulting scheme is

$$x_k^{(1)} = x_k - p_k\Gamma_k^{(1)}, \qquad \text{(Projection step)} \tag{8a}$$

$$x_k^{(2)} = x_k^{(1)} + \beta_k r_k^{(1)}, \qquad \text{(Mixing step)} \tag{8b}$$

$$x_{k+1} = x_k^{(2)} - p_k\Gamma_k^{(2)}, \quad \text{(Projection step)} \tag{8c}$$

where $r_k^{(1)} := r_k - q_k\Gamma_k^{(1)}$ and $\beta_k > 0$. Define $r_k^{(2)} = r_k^{(1)} - \beta_k A r_k^{(1)}$ and $r_k^{(3)} = r_k^{(2)} - q_k\Gamma_k^{(2)}$. The coefficients $\Gamma_k^{(1)}$ and $\Gamma_k^{(2)}$ are determined by applying the Galerkin's condition of AM-I:

$$r_k^{(1)} = r_k - q_k\Gamma_k^{(1)} \perp p_k, \quad r_k^{(3)} = r_k^{(2)} - q_k\Gamma_k^{(2)} \perp p_k. \tag{9}$$

Assume $p_k^\top q_k \neq 0$. The scheme (8a)-(8c) give the x_{k+1} by

$$x_{k+1} = x_k + H_k r_k, \quad H_k = -\frac{p_k p_k^{\mathrm{T}}}{p_k^{\mathrm{T}} q_k} + \beta_k \left(I - \frac{p_k q_k^{\mathrm{T}}}{p_k^{\mathrm{T}} q_k}\right)\left(I - \frac{q_k p_k^{\mathrm{T}}}{p_k^{\mathrm{T}} q_k}\right), \tag{10}$$

leading to the basic Min-AM. It is worth mentioning that the approximated inverse Hessian H_k in (10) is symmetric. Due to the space limit, we give the details of the basic Min-AM in Algorithm 2 in Appendix A.4. Define $P_k = (p_1, p_2, \ldots, p_k), Q_k = (q_1, q_2, \ldots, q_k)$, we summarize the properties of the basic Min-AM in the next theorem.

Theorem 1. *Let $f(x) = \frac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x$ where A is SPD and $\{x_k\}$ be the sequence generated by* (10) *for solving* (1)*. x^* is the exact solution. Then, the following properties hold:*
(i) $\|p_k\|_2 > 0, \mathrm{range}(P_k) = \mathcal{K}_k(A, r_0), \mathrm{range}(Q_k) = A\mathcal{K}_k(A, r_0)$;
(ii) $Q_k = -AP_k, p_i \perp q_j (1 \leq i \neq j \leq k)$;
(iii) $r_k^{(1)} \perp \mathrm{range}(P_k)$ and $x_k^{(1)} = x_0 + z_k$, where $z_k = \arg\min_{z \in \mathcal{K}_k(A, r_0)} \|x_0 + z - x^*\|_A$.
Moreover, if $\|r_k^{(1)}\|_2 = 0$, then $x_{k+1} = x^*$.

The proof is deferred to Appendix A.4. The property (i) ensures $p_k^{\mathrm{T}} q_k \neq 0$ during the iterations, so ζ_k and the update (10) are well defined. The property (iii) indicates $x_k^{(1)} = x_k^{\mathrm{CG}}$, where x_k^{CG} is the k-th iterate of the conjugate gradient (CG) method [25]. Also, due to Proposition 1 in Appendix A.4, the intermediate iterate $\bar{x}_k$ of AM-I(∞) satisfies $\bar{x}_k = x_k^{\mathrm{CG}}$, so the basic Min-AM is essentially equivalent to the full-memory AM-I when solving unconstrained strongly convex quadratic problems.

Remark 2. *Although Min-AM and CG is equivalent for solving SPD linear systems, their numerical performance has large difference for general nonlinear programming as shown in our experiments.*

Remark 3. *The approximated inverse Hessian H_k given in* (10) *has the same formulation as memoryless BFGS method [4], but the key difference lies in the construction of p_k and q_k in* (7)*. In memoryless BFGS method, there are no correction terms $p_{k-1}\zeta_k$ and $q_{k-1}\zeta_k$ and it has no equivalence to the full-memory methods.*

3.2 The restarted Min-AM

For general nonlinear optimization, since global convergence may be unavailable for Min-AM as a counter example exists for AM [33], we introduce the restarting strategy to the basic Min-AM and establish the local convergence analysis. Our purpose is to prove the faster convergence of the restarted Min-AM than that of gradient descent (GD), and sharpen the existing results of AM [50, 17].

Restarting Strategy. Let m_k be the number of iterations from the last restart. Min-AM restarts, i.e. $p_k = q_k = \mathbf{0}$, when any of the following conditions is violated:

$$m_k \leq m, \tag{11a}$$

$$|q_k^{\mathrm{T}} p_k| \geq \tau |q_{k-m_k+1}^{\mathrm{T}} p_{k-m_k+1}|, \tag{11b}$$

$$\|\nabla f(x_k)\|_2 \leq \eta \|\nabla f(x_{k-m_k})\|_2, \tag{11c}$$

where m is a constant positive integer, and $0 < \tau < 1, \eta > 0$ are constants.

The check (11a) to limit the number of high-order terms of residuals accumulated during iterations. The check (11b) is for the numerical stability of Min-AM since $p_k^{\mathrm{T}} q_k$ appears in the denominators in (10). The check (11c) is to control divergence. The algorithm is shown in Algorithm 1. Let $\mathcal{B}_\rho(x) := \{y \in \mathbb{R}^d | \|y - x\|_2 \leq \rho\}$. We impose the following assumptions on f that are standard in the convergence analysis of quasi-Newton methods [37], and the regularized nonlinear acceleration method [47].

Assumption 1. *$f : \mathbb{R}^d \to \mathbb{R}$ is twice Lipschitz continuously differentiable in a neighbourhood of a local minimum x^* where the Hessian $A := \nabla^2 f(x^*)$ is SPD.*

Algorithm 1 Restarted Min-AM for general nonlinear optimization

Input: $x_0 \in \mathbb{R}^d, \beta_k > 0, m > 0, \tau > 0, \eta > 0$.
Output: $x \in \mathbb{R}^d$

$p_0, q_0 = \mathbf{0} \in \mathbb{R}^d, m_k = 0$
for $k = 0, 1, \ldots,$ until convergence **do**
  $r_k = -\nabla f(x_k)$
  if $m_k > m$ or $\|r_k\|_2 > \eta \|r_{k-m_k}\|_2$ **then**
    $m_k = 0$
  end if
  if $m_k > 0$ **then**
    $p = x_k - x_{k-1}, q = r_k - r_{k-1}$
    $\zeta_k = (p_{k-1}^{\mathrm{T}} q_{k-1})^{\dagger} p_{k-1}^{\mathrm{T}} q$
    $p_k = p - p_{k-1}\zeta_k, q_k = q - q_{k-1}\zeta_k$
    if $|p_k^{\mathrm{T}} q_k| < \tau |p_{k-m_k+1}^{\mathrm{T}} q_{k-m_k+1}|$ **then**
      $m_k = 0$
    end if
  end if
  if $m_k = 0$ **then**
    $p_k = q_k = \mathbf{0} \in \mathbb{R}^d$
  end if
  $x_{k+1} = x_k - (p_k^{\mathrm{T}} q_k)^{\dagger} p_k p_k^{\mathrm{T}} r_k + \beta_k \left(I - (p_k^{\mathrm{T}} q_k)^{\dagger} p_k q_k^{\mathrm{T}}\right) \left(I - (p_k^{\mathrm{T}} q_k)^{\dagger} q_k p_k^{\mathrm{T}}\right) r_k$
  $m_k = m_k + 1$
end for
return x_k

From Assumption 1, there exist positive constants $\hat{\rho}, \hat{\kappa}$ and μ, L such that for all $x \in \mathcal{B}_{\hat{\rho}}(x^*)$,

$$\mu \leq \frac{y^{\mathrm{T}} \nabla^2 f(x) y}{y^{\mathrm{T}} y} \leq L, \text{ for all } y \in \mathbb{R}^d \text{ and } y \neq \mathbf{0}, \tag{12a}$$

$$\|\nabla f(x) - \nabla^2 f(x^*)(x - x^*)\|_2 \leq \frac{1}{2}\hat{\kappa}\|x - x^*\|_2^2. \tag{12b}$$

With the restarting strategy, we apply a multi-step analysis and has the following convergence theory:

Theorem 2. *Suppose that the Assumption 1 holds. Let $\{x_k\}$ be the sequence generated by the restarted Min-AM for solving* (1) *and $\theta_k = \max\{|1 - \beta_k L|, |1 - \beta_k \mu|\} \leq \theta$ for a constant $\theta \in (0, 1)$. Then there is a positive constant $\rho \leq \hat{\rho}$ such that for any $x_0 \in \mathcal{B}_{\rho}(x^*)$, we have*

$$\|x_{k+1} - x^*\|_A \leq 2\theta_k \sqrt{L/\mu} \left(\frac{\sqrt{L/\mu} - 1}{\sqrt{L/\mu} + 1}\right)^{m_k} \|x_{k-m_k} - x^*\|_A + \hat{\kappa} \cdot \mathcal{O}(\|x_{k-m_k} - x^*\|_2^2). \tag{13}$$

Moreover, if $m_k \geq d$, then $\|x_{k+1} - x^*\|_2 = \hat{\kappa}\mathcal{O}(\|x_{k-m_k} - x^*\|_2^2)$.

The proof is in Appendix A.5. The result from Theorem 2 suggests that the restarted Min-AM method has a local linear-quadratic convergence rate, and the rate has optimal dependence on the condition number. Compared to the existing results that show the local linear convergence rate of AM is no worse than that of gradient descent (GD) method [50, 12], the bound (13) clearly improves the convergence rate of GD.

Remark 4. *Classical nonlinear CG methods [13] are also memory-efficient with recursively modified searching directions, but the step size relies on line-search or Hessian-vector products which can be costly for large-scale optimization.*

Remark 5. *The momentum based methods, e.g. Nesterov's accelerated gradient (NAG) method [36], also achieve the optimal convergence rate, but they always need to know the parameters μ and L. In the next section, we give an economic method that approximates the best step size $2/(L + \mu)$.*

3.3 Eigenvalue estimates and choice of mixing parameters

The mixing parameter β_k is critical for AM and AM-I, and an improper choice of β_k can lead to divergence [18]. Fortunately, the spectrum of the Hessian can be economically estimated as a

by-product of Min-AM. To better explain the idea, we consider $\min\ f(x) = \frac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x$ where A is SPD. We apply a projection method to estimate the eigenvalues:

$$v \in \mathcal{K}_k(A, r_0), \quad (A - \lambda I)v \perp A\mathcal{K}_k(A, r_0), \tag{14}$$

where $v \in \mathbb{R}^d$ is the approximate eigenvector in the Krylov subspace $\mathcal{K}_k(A, r_0)$ and $\lambda \in \mathbb{R}$ is an eigenvalue estimate, which is known as a generalized Ritz value [34]. Our goal is compute λ. Since $\mathrm{range}(P_k) = \mathcal{K}_k(A, r_0)$ as shown in Theorem 1, we can select $v = P_k y,\ y \in \mathbb{R}^k$. Then the projection condition (14) leads to

$$P_k^{\mathrm{T}} A A P_k y = \lambda P_k^{\mathrm{T}} A P_k y. \tag{15}$$

From the construction of P_k and Q_k, we can derive the three-term recurrence relations (see details in Appendix A.6): there exists a tridiagonal matrix $\bar{T}_k \in \mathbb{R}^{(k+1)\times k}$, such that

$$AP_k = P_{k+1}\bar{T}_k, \quad AQ_k = Q_{k+1}\bar{T}_k. \tag{16}$$

Then, we have $P_k^{\mathrm{T}} A A P_k = P_k^{\mathrm{T}} A P_{k+1}\bar{T}_{k+1}$, By Theorem 1, $P_k^{\mathrm{T}} A p_{k+1} = -Q_k^{\mathrm{T}} p_{k+1} = 0$. Then,

$$P_k^{\mathrm{T}} A P_{k+1}\bar{T}_{k+1} = P_k^{\mathrm{T}} A P_k T_k, \tag{17}$$

where $T_k \in \mathbb{R}^{k\times k}$ is obtained from $\bar{T}_k$ by deleting its last row. By (15) and (17), we obtain $T_k y = \lambda y$, which indicates that the eigenvalue problem (14) is reduced to solve the eigenvalues of a tridiagonal matrix T_k. Moreover, the three elements of the k-th column of $\bar{T}_k$ are easy to be obtained from the coefficients $\Gamma_k^{(1)}, \Gamma_k^{(2)}, \beta_{k-1}$ and β_k on the fly (see details in Appendix A.6):

$$t_k^{(k-1)} = \frac{\Gamma_{k-1}^{(2)}}{\beta_{k-1}(1-\Gamma_k^{(1)})}, \ \ t_k^{(k)} = \frac{1}{1-\Gamma_k^{(1)}}\left(\frac{1}{\beta_{k-1}} - \frac{\Gamma_k^{(2)}}{\beta_k}\right), \ \ t_k^{(k+1)} = -\frac{1}{\beta_k(1-\Gamma_k^{(1)})}. \tag{18}$$

Due to the tridiagonal structure of T_k, its eigenvalues can be computed by efficient algorithms [20]. For general nonlinear optimization, the eigenvalues of $T_k \in \mathbb{R}^{m_k\times m_k}$ from (18) can still serve as promising estimates of the spectrum of the Hessian.

Theorem 3. *Under the same assumptions of Theorem 2, assume that $|1-\Gamma_j^{(1)}| \geq \tau_0$ for some constant $\tau_0 > 0$, $j = 1, \ldots, k$, and let λ be an eigenvalue of T_k. The $\Theta^{(m_k)}$ denotes the set of Ritz values computed by an m_k-step A-norm based Lanczos algorithm (Algorithm 3 in Appendix A.6) for $\nabla f^2(x^*)$ with starting vector $\nabla^2 f(x^*)(x^* - x_{k-m_k})$. Then there is a positive constant $\rho \leq \hat{\rho}$ such that for any $x_0 \in \mathcal{B}_\rho(x^*)$, we have*

$$\min_{\hat{\lambda}\in\Theta^{(m_k)}} |\hat{\lambda} - \lambda| = \hat{\kappa}\mathcal{O}(\|x_{k-m_k} - x^*\|_2). \tag{19}$$

The proof is in Appendix A.6. It suggests that if $\|x_0 - x^*\|_2$ is sufficiently small, Min-AM can obtain accurate enough eigenvalue estimates. Suppose that at the k-th ($k \geq 2$) iteration, we obtain $\tilde{\mu}$ and $\tilde{L}$ as the estimates of μ and L, then we can adaptively take the mixing parameter as $\beta_k = 2/(|\tilde{\mu}| + |\tilde{L}|)$ as an estimate of the optimal value $2/(\mu + L)$. Hence, this strategy can be a useful and very economical option when tuning the β_k is time-consuming and expensive.

Remark 6. *The spectrum of Hessian is an important property of the optimization problem. We provide a practical method to estimate the spectrum during the optimization process. Once we obtain the estimation of μ and L by Min-AM, we can apply them to many first-order methods.*

3.4 Stochastic optimization

For many applications in deep learning, the large data size prohibits the evaluation of the full gradient, and the stochastic optimization stands out as a solution. Specifically, we consider developing stochastic version of Min-AM to solve the following nonconvex stochastic optimization problem:

$$\min_{x\in\mathbb{R}^d} f(x) := \frac{1}{T}\sum_{i=1}^{T} f_{\xi_i}(x), \tag{20}$$

where $f_{\xi_i} : \mathbb{R}^d \to \mathbb{R}$ may be nonconvex. The gradient estimate in a sampled mini-batch $S_k \subseteq [T] := \{1, 2, \ldots, T\}$ is $f_{S_k}(x_k) := \frac{1}{n_k}\sum_{i\in S_k} f_{\xi_i}(x_k)$, where $n_k := |S_k|$ is the batch size. Then, $r_k := -\nabla f_{S_k}(x_k)$ is an unbiased estimate of the negative gradient evaluated at S_k.

Due to the noise in the gradient estimates and the nonconvexity of f, a direct application of the deterministic Min-AM to stochastic optimization may be problematic. For example, the searching direction may be not a descent direction since H_k can be indefinite. Hence, we should introduce some fundamental modifications to the original Min-AM.

Regularization. To ensure a descent direction, inspired by the regularization used in [19], we define the function:

$$\Phi(p, q, \delta) = p^{\mathrm{T}} q - \delta(\|p\|_2^2 + \|q\|_2^2). \tag{21}$$

Let $\Phi_k := \Phi(p_k, q_k, \delta_k^{(2)})$ for some $\delta_k^{(2)} > 0$. We substitute Φ_k for the $p_k^{\mathrm{T}} q_k$ appeared in (10) and define $\rho_k = \Phi_k^{\dagger}$, i.e. $\rho_k = 0$ if $\Phi_k = 0$ and $\rho_k = \Phi_k^{-1}$ if $\Phi_k \neq 0$. The resulting regularized Min-AM update is $x_{k+1} = x_k + H_k^A r_k$, where

$$H_k^A = -\rho_k p_k p_k^{\mathrm{T}} + \beta_k (I - \rho_k p_k q_k^{\mathrm{T}})(I - \rho_k q_k p_k^{\mathrm{T}}). \tag{22}$$

Damping. Damping is useful for the stability of many numerical algorithms, e.g. damped Newton's method [37]. We propose a specialized damping technique for Min-AM, which interpolates the updates of stochastic gradient descent [42] (SGD) and Min-AM, and is described as

$$x_k^G = x_k + \beta_k r_k, \quad x_k^A = x_k + H_k^A r_k, \quad x_{k+1} = (1 - \alpha_k) x_k^G + \alpha_k x_k^A, \tag{23a}$$

where x_k^G is the SGD update, x_k^A is the Min-AM update, and α_k is the damping parameter. It can be seen that SGD and Min-AM serve as the predicted step, and x_{k+1} is a weighted average of them. Moreover, the Hessian approximation is

$$H_k = (1 - \alpha_k)\beta_k I + \alpha_k H_k^A \succeq (1 - \alpha_k)\beta_k I \succ 0 \tag{24}$$

provided that $\alpha_k \in [0, 1)$ and $H_k^A \succeq 0$.

By incorporating damping and regularization, we obtain the stochastic Min-AM (sMin-AM), and the algorithm is summarized in Algorithm 4 in Appendix A.7. Note that we also introduce regularization to the construction of p_k and q_k. To analyze the convergence properties, we first impose the following assumptions on the objective function f.

Assumption 2. *The objective function $f : \mathbb{R}^d \to \mathbb{R}$ in (20) satisfies: (1)* continuously differentiable*; (2) $f(x) \geq f^{low} > -\infty$, $\forall x \in \mathbb{R}^d$; (3) ∇f is L-Lipschitz continuous.*

Assumption 3. *For any iteration k, the gradient estimate $\nabla f_{\xi_k}(x_k)$ satisfies $\mathbb{E}_{\xi_k}[\nabla f_{\xi_k}(x_k)] = \nabla f(x_k)$, $\mathbb{E}_{\xi_k}[\|\nabla f_{\xi_k}(x_k) - \nabla f(x_k)\|_2^2] \leq \sigma^2$, where $\sigma > 0$, and $\xi_k, k = 0, 1, \ldots$ are independent samples that are independent of $\{x_j\}_{j=0}^k$.*

The diminishing condition about β_k is

$$\sum_{k=0}^{+\infty} \beta_k = +\infty, \qquad \sum_{k=0}^{+\infty} \beta_k^2 < +\infty. \tag{25}$$

With chosen β_k, we choose the α_k and $\delta_k^{(2)}$ of sMin-AM to meet the following assumption:

Assumption 4. *Given constants $\mu \in (0, 1)$, $C_1 > 0$ and $C_2 > 0$, the $\alpha_k \in [0, \min\{C_1 \beta_k^{1/2}, 1 - \mu\}]$ and $\delta_k^{(2)}$ is chosen such that $\rho_k \leq 0$ and $-\rho_k \|p_k\|_2^2 - 2\beta_k \rho_k \|p_k\|_2 \|q_k\|_2 + \beta_k \rho_k^2 \|p_k\|_2^2 \|q_k\|_2^2 \leq \beta_k C_2$.*

We establish the convergence results of sMin-AM in nonconvex (stochastic) optimization and proofs can be found in Appendix A.7.

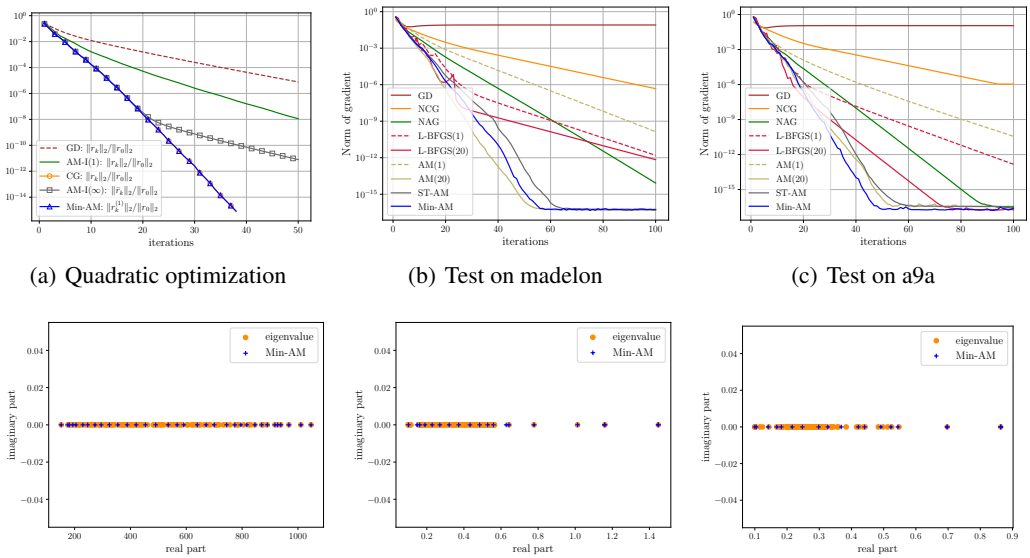
Theorem 4. *Suppose that Assumptions 2-4 hold and $\{x_k\}$ is the sequence generated by sMin-AM with batch size $n_k = n \leq T$. If $\beta_k \in (0, \frac{\mu}{2L(1+C_2)^2}]$ and satisfies (25), then*

$$\liminf_{k \to \infty} \|\nabla f(x_k)\|_2 = 0 \textit{ with probability } 1, \tag{26}$$

and for all k, there exists $M_f > 0$ such that $\mathbb{E}[f(x_k)] \leq M_f$.

If $\mathbb{E}_{\xi_k}[\|\nabla f_{\xi_k}(x_k)\|_2^2] \leq M_g$, $\forall k$ where $M_g > 0$ is a constant, we have

$$\lim_{k \to \infty} \|\nabla f(x_k)\|_2 = 0 \textit{ with probability } 1. \tag{27}$$



(a) Quadratic optimization (b) Test on madelon (c) Test on a9a

(d) Eigenvalue estimates (e) Eigenvalue estimates (f) Eigenvalue estimates

Figure 1: Experiments about the deterministic Min-AM. Left column (a)(d): the residuals and eigenvalue estimates for strongly convex quadratic optimization. For AM-I(∞) and Min-AM, the curves are about $\bar{r}_k, r_k^{(1)}$ respectively. Middle column (b)(e): regularized logistic regression on madelon dataset. Right column (c)(f): regularized logistic regression on a9a dataset.

Theorem 5. *Suppose that Assumptions 2-4 hold and $\{x_k\}_{k=0}^{N-1}$ is the first N iterations generated by sMin-AM with fixed batch size $n_k = n$. Let $\beta_k = \min\{\frac{\mu}{2L(1+C_2)^2}, \frac{\tilde{D}}{\sigma\sqrt{N}}\}$, where $\tilde{D}$ is a problem-independent constant. R is a random variable following $P_R(k) := Prob\{R = k\} = 1/N$, then*

$$\mathbb{E}[\|\nabla f(x_R)\|_2^2] \leq \frac{8D_f L(1+C_2)^2}{N\mu^2} + \frac{\sigma}{\mu\sqrt{N}}\left(\frac{4D_f}{\tilde{D}} + \frac{2(\mu^{-1}C_1^2C_2^2 + L(1+C_2)^2)\tilde{D}}{n}\right), \quad (28)$$

where $D_f := f(x_0) - f^{low}$ and the expectation is taken with respect to R and $\{S_j\}_{j=0}^{N-1}$. To ensure $\mathbb{E}[\|\nabla f(x_R)\|_2^2] \leq \epsilon$, the number of iterations is $\mathcal{O}(1/\epsilon^2)$.

4 Experiments

We validated the proposed Min-AM methods in deterministic optimization and stochastic optimization. First, we tested the basic Min-AM in solving strongly convex quadratic optimization (cf. Theorem 1) and the restarted Min-AM on logistic regression (cf. Theorem 2). Then, we apply the sMin-AM to train deep neural networks on CIFAR [31] and ImageNet [15]. Details about the experimental settings and hyperparameters can be found in Appendix B.

Strongly Convex Quadratic Optimization. As shown in Figure 1(a), the first intermediate step $x_k^{(1)}$ of Min-AM exactly matches the k-th iterate of CG, which verifies Theorem 1. It can also be found that AM-I(∞) fails to coincide with CG and Min-AM due to the numerical weakness to solve (6) in finite arithmetic. The eigenvalue estimates computed by Min-AM are also very accurate, converging to the true eigenvalues in high precision.

Regularized logistic regression. We conducted the regularized logistic regression using the datasets 'madelon' and 'a9a' from LIBSVM [11]. The compared methods are GD, nonlinear CG (NCG) with line-search, Nesterov's accelerated gradient (NAG), L-BFGS(m), AM(m), and the ST-AM. We applied the standard Lanczos algorithm [20] to compute 100 eigenvalue values of $\nabla^2 f(x^*)$, in which the smallest and largest ones were used for the setting of μ, L in NAG. The results in Figure 1 show that Min-AM is competitive with AM(20). It converges much faster than AM(1), L-BFGS(1) (i.e. memoryless BFGS), NCG and outperforms NAG. Also, in both datasets, the eigenvalue estimates

from Min-AM are accurate enough to approximate the largest and the smallest eigenvalues, even though the objective function is non-quadratic.

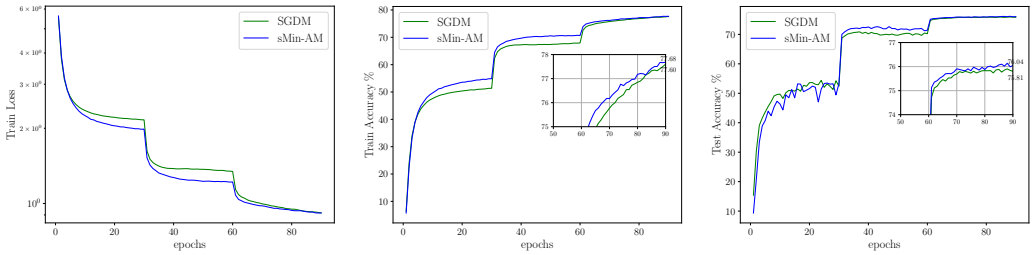
Table 1: Experiments on CIFAR10/CIFAR100. "-" means failing to complete the test in our device.

(a) Final TOP1 test accuracy (mean ± standard deviation) (%) on CIFAR10/CIFAR100.

Method	CIFAR10						CIFAR100		
	VGG16	ResNet18	ResNet20	ResNet44	ResNet56	WRN16-4	ResNet18	ResNeXt	DenseNet
SGDM	93.52±.15	94.82±.15	92.26±.16	93.23±.18	93.47±.28	94.86±.06	77.27±.09	78.41±.54	78.49±.12
Adam	92.29±.09	93.06±.22	91.13±.13	92.21±.33	92.42±.49	92.58±.04	72.41±.17	73.81±.13	70.23±.54
AdaHessian	93.42±.11	94.36±.09	91.92±.32	92.74±.11	92.40±.06	94.04±.12	76.59±.42	-	-
SAM(1)	93.35±.12	94.95±.10	92.10±.05	93.15±.23	93.44±.19	95.02±.32	77.06±.48	79.12±.40	79.58±.29
SAM(10)	93.59±.11	95.17±.10	**92.43±.19**	93.57±.14	93.77±.12	**95.23±.07**	78.13±.14	79.31±.27	80.09±.52
ST-AM	93.72±.10	95.27±.04	92.39±.11	93.52±.02	93.69±.18	95.21±.09	77.91±.22	79.53±.34	**80.36±.25**
sMin-AM	**94.09±.12**	**95.28±.04**	92.42±.16	**93.72±.27**	**93.82±.10**	95.13±.03	**78.26±.18**	**79.64±.19**	79.88±.09

(b) The memory and computation cost compared with SGDM. The notations "m","t/e" and "t" are abbreviations of memory, per-epoch time and total running time, respectively.

Cost (× SGDM)	CIFAR10/ResNet18			CIFAR10/VGG16			CIFAR100/ResNeXt50			CIFAR100/DenseNet121		
	m	t/e	t	m	t/e	t	m	t/e	t	m	t/e	t
SGDM	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SAM(10)	1.73	1.78	1.00	2.51	2.59	2.59	1.30	1.16	0.58	1.16	1.19	0.60
ST-AM	1.05	1.46	0.82	1.55	1.91	1.67	1.04	1.07	0.54	1.01	1.11	0.55
sMin-AM	1.01	1.15	0.64	1.35	1.25	0.78	1.03	1.00	0.50	1.01	1.09	0.55



(a) Training loss (b) Train accuracy (c) Test accuracy

Figure 2: Experiment of training ResNet50 on ImageNet.

Experiments on CIFAR. We applied sMin-AM to train VGG16 [48], ResNet18/20/44/56 [23], WideResNet16-4 [57] (abbr. WRN16-4) on CIFAR-10, and ResNet18, ResNeXt50 [55], DenseNet121 [27] on CIFAR-100. The baseline optimizers were SGDM [41], Adam [29], AdaHessian [56], stochastic AM (SAM) [53] and ST-AM [54]. As shown in Table 1(a), the sMin-AM improves SAM(1) and has comparable accuracy to SAM(10). By setting the accuracy of SGDM as baseline, Table 1(b) shows the memory, per-epoch time and the total time for an optimizer to achieve a comparable accuracy to SGDM. sMin-AM significantly reduces the memory cost of SAM(10) and the total training time is less than SGDM.

Experiments on ImageNet. We trained ResNet50 on ImageNet with SGDM and sMin-AM. Figure 2 shows that the training process of sMin-AM is faster than SGDM. sMin-AM can achieve both higher train accuracy and higher test accuracy in fewer epochs.

5 Conclusion

In this paper, we propose Min-AM, a variant of AM(1) with minimal memory size. Min-AM only stores one recursively modified vector pair and is essentially equivalent to the full-memory Type-I AM in strongly convex quadratic optimization. We establish a complete convergence analysis for Min-AM under deterministic and stochastic setting respectively. For deterministic optimization, we also propose an economical method to estimate the spectrum of Hessian, which can be used to adaptively choose the mixing parameters. The experiment results validates the effectiveness of our methods.

References

[1] Hengbin An, Xiaowei Jia, and Homer F Walker. Anderson acceleration and application to the three-temperature energy equations. *Journal of Computational Physics*, 347:1–19, 2017.

[2] Donald G Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 12(4):547–560, 1965.

[3] Donald GM Anderson. Comments on "Anderson acceleration, mixing and extrapolation". *Numerical Algorithms*, 80(1):135–234, 2019.

[4] Neculai Andrei. A note on memory-less SR1 and memory-less BFGS methods for large-scale unconstrained optimization. *Numerical Algorithms*, pages 1–18, 2021.

[5] Akash Arora, David C Morse, Frank S Bates, and Kevin D Dorfman. Accelerating self-consistent field theory of block polymers in a variable unit cell. *The Journal of chemical physics*, 146(24):244902, 2017.

[6] Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. SIAM, 2000.

[7] Albert S Berahas, Frank E Curtis, and Baoyu Zhou. Limited-memory BFGS with displacement aggregation. *Mathematical Programming*, pages 1–37, 2021.

[8] Paul T Boggs and Richard H Byrd. Adaptive, limited-memory BFGS algorithms for unconstrained optimization. *SIAM Journal on Optimization*, 29(2):1282–1299, 2019.

[9] Raghu Bollapragada, Jorge Nocedal, Dheevatsa Mudigere, Hao-Jun Shi, and Ping Tak Peter Tang. A progressive batching L-BFGS method for machine learning. In *International Conference on Machine Learning*, pages 620–629. PMLR, 2018.

[10] Richard H Byrd, Samantha L Hansen, Jorge Nocedal, and Yoram Singer. A stochastic quasi-Newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

[11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[12] Xiaojun Chen and CT Kelley. Convergence of the EDIIS algorithm for nonlinear equations. *SIAM Journal on Scientific Computing*, 41(1):A365–A379, 2019.

[13] Arthur I Cohen. Rate of convergence of several conjugate gradient algorithms. *SIAM Journal on Numerical Analysis*, 9(2):248–259, 1972.

[14] Yu-Hong Dai, William W Hager, Klaus Schittkowski, and Hongchao Zhang. The cyclic Barzilai—Borwein method for unconstrained optimization. *IMA Journal of Numerical Analysis*, 26(3):604–627, 2006.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[16] Rick Durrett. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.

[17] Claire Evans, Sara Pollock, Leo G Rebholz, and Mengying Xiao. A proof that Anderson acceleration improves the convergence rate in linearly converging fixed-point methods (but not in those converging quadratically). *SIAM Journal on Numerical Analysis*, 58(1):788–810, 2020.

[18] Haw-ren Fang and Yousef Saad. Two classes of multisecant methods for nonlinear acceleration. *Numerical Linear Algebra with Applications*, 16(3):197–221, 2009.

[19] Anqi Fu, Junzi Zhang, and Stephen Boyd. Anderson accelerated Douglas–Rachford splitting. *SIAM Journal on Scientific Computing*, 42(6):A3560–A3583, 2020.

[20] Gene H Golub and Charles F Van Loan. Matrix computations, 4th. *Johns Hopkins*, 2013.

[21] Robert Gower, Donald Goldfarb, and Peter Richtárik. Stochastic block BFGS: Squeezing more curvature out of data. In *International Conference on Machine Learning*, pages 1869–1878, 2016.

[22] Anne Greenbaum. *Iterative methods for solving linear systems*. SIAM, 1997.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Nicholas C Henderson and Ravi Varadhan. Damped anderson acceleration with restarts and monotonicity control for accelerating em and em-like algorithms. *Journal of Computational and Graphical Statistics*, 28(4):834–846, 2019.

[25] Magnus R Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems1. *Journal cf Research of the National Bureau of Standards Vol*, 49(6), 1952.

[26] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[27] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[28] Carl T Kelley. Numerical methods for nonlinear equations. *Acta Numerica*, 27:207–287, 2018.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Tamara G Kolda, Dianne P O'leary, and Larry Nazareth. BFGS with update skipping and varying memory. *SIAM Journal on Optimization*, 8(4):1060–1083, 1998.

[31] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[32] Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[33] Vien Mai and Mikael Johansson. Anderson acceleration of proximal gradient methods. In *International Conference on Machine Learning*, pages 6620–6629. PMLR, 2020.

[34] Thomas A Manteuffel and James S Otto. On the roots of the orthogonal polynomials and residual polynomials associated with a conjugate gradient method. *Numerical linear algebra with applications*, 1(5):449–475, 1994.

[35] Arkadij Semenovič Nemirovski and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

[36] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

[37] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[38] Sara Pollock and Leo G Rebholz. Anderson acceleration for contractive and noncontractive operators. *IMA Journal of Numerical Analysis*, 41(4):2841–2872, 2021.

[39] Sara Pollock, Leo G Rebholz, and Mengying Xiao. Anderson-accelerated convergence of Picard iterations for incompressible Navier–Stokes equations. *SIAM Journal on Numerical Analysis*, 57(2):615–637, 2019.

[40] Florian A Potra and Hans Engler. A characterization of the behavior of the Anderson acceleration on linear problems. *linear Algebra and its Applications*, 438(3):1002–1011, 2013.

[41] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[42] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[43] Youcef Saad and Martin H Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7 (3):856–869, 1986.

[44] Yousef Saad. Krylov subspace methods for solving large unsymmetric linear systems. *Mathematics of computation*, 37(155):105–126, 1981.

[45] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[46] Damien Scieur, Alexandre d'Aspremont, and Francis Bach. Regularized nonlinear acceleration. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 712–720, 2016.

[47] Damien Scieur, Alexandre d'Aspremont, and Francis Bach. Regularized nonlinear acceleration. *Mathematical Programming*, 179(1):47–83, 2020.

[48] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[49] Ke Sun, Yafei Wang, Yi Liu, Bo Pan, Shangling Jui, Bei Jiang, Linglong Kong, et al. Damped anderson mixing for deep reinforcement learning: Acceleration, convergence, and stabilization. *Advances in Neural Information Processing Systems*, 34, 2021.

[50] Alex Toth and CT Kelley. Convergence analysis for Anderson acceleration. *SIAM Journal on Numerical Analysis*, 53(2):805–819, 2015.

[51] Homer F Walker and Peng Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011.

[52] Xiao Wang, Shiqian Ma, Donald Goldfarb, and Wei Liu. Stochastic quasi-Newton methods for nonconvex stochastic optimization. *SIAM Journal on Optimization*, 27(2):927–956, 2017.

[53] Fuchao Wei, Chenglong Bao, and Yang Liu. Stochastic Anderson mixing for nonconvex stochastic optimization. *Advances in Neural Information Processing Systems*, 34, 2021.

[54] Fuchao Wei, Chenglong Bao, and Yang Liu. A class of short-term recurrence Anderson mixing methods and their applications. In *International Conference on Learning Representations*, 2022.

[55] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[56] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer, and Michael Mahoney. Adahessian: An adaptive second order optimizer for machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10665–10673, 2021.

[57] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference 2016*. British Machine Vision Association, 2016.

[58] Junzi Zhang, Brendan O'Donoghue, and Stephen Boyd. Globally convergent type-i anderson acceleration for nonsmooth fixed-point iterations. *SIAM Journal on Optimization*, 30(4):3170–3197, 2020.

[59] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9597–9608, 2019.

[60] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in Neural Information Processing Systems*, 33, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A in the supplemental material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See supplemental material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We reported standard deviation instead.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The assets are publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]