SIAM J. SCI. COMPUT.
Vol. 45, No. 4, pp. A1571–A1590

FILTERING FOR ANDERSON ACCELERATION*

SARA POLLOCK† AND LEO G. REBHOLZ‡

Abstract. This work introduces, analyzes, and demonstrates an efficient and theoretically sound filtering strategy to ensure the condition of the least-squares problem solved at each iteration of Anderson acceleration. The filtering strategy consists of two steps: the first controls the length disparity between columns of the least-squares matrix, and the second enforces a lower bound on the angles between subspaces spanned by the columns of that matrix. The combined strategy is shown to control the condition number of the least-squares matrix at each iteration. The method is shown to be effective on a range of problems based on discretizations of partial differential equations. It is shown to be particularly effective for problems where the initial iterate may lie far from the solution and which progress through distinct preasymptotic and asymptotic phases.

Key words. Anderson acceleration, filtering, conditioning

MSC codes. 65B05, 65N30

DOI. 10.1137/22M1536741

1. Introduction.

Nonlinear problems are ubiquitous throughout physical modeling, mathematics, and data sciences. Many basic iterative methods for solving such problems can be written in the form of a fixed-point iteration: $x_{k+1} = g(x_k)$, where a sequence of iterates x_k form approximations to the fixed-point solution x. Anderson acceleration (AA) has become an increasingly popular method for decreasing the number of fixed-point iterations to convergence, and in many cases enabling convergence where the original fixed-point iteration fails. It involves a correction to each fixed-point update formed from a linear combination of a history of iterates and update steps, where the linear combination is chosen so that the corrected update step has least length. Its popularity is due in large part to how effective it is for problems over a wide range of fields, including, to name a few, quantum chemistry, physics, multiphysics, and flow phenomena [1, 11, 13, 18, 23, 27], and recently data science and optimization [20, 26, 30].

AA was introduced in 1965 by Anderson [2] in the context of integral equations, analyzed as a generalized multi-secant or quasi-Newton algorithm in [12, 13], and discussed within a Krylov space framework and popularized by its effective and efficient application across a variety of problems in [29]. Local convergence of the iteration was first shown in [28], and the acceleration property of the method was first theoretically developed in [11, 22, 23]. The method is well known to be sensitive to implementation and parameter choices, including those for the relaxation factor at each iteration and the (maximal) algorithmic depth, as well as whether and how often the method should be restarted. Here we introduce a stabilization through selectively eliminating

*Submitted to the journal's Methods and Algorithms for Scientific Computing section November 23, 2022; accepted for publication (in revised form) March 7, 2023; published electronically July 7, 2023.

https://doi.org/10.1137/22M1536741

Funding: The first author's work was partially supported by NSF grant DMS 2011519. The second author's research was partially supported by NSF grant DMS 2011490.

†Department of Mathematics, University of Florida, Gainesville, FL 32611 USA (s.pollock@ufl.edu).

‡School of Mathematical and Statistical Sciences, Clemson University, Clemson, SC 29634 USA (rebholz@clemson.edu).

steps from the history in order to improve convergence and control the condition of the inner optimization.

Herein we seek a fixed-point x of $g(x)$ by a fixed-point iteration $x_{k+1} = g(x_k)$. Define the difference between iterates e_k and the residual w_k by

$$e_k := x_k - x_{k-1}, \quad w_k := g(x_{k-1}) - x_{k-1}.$$

Then AA can be written as follows, where parameter m is the maximum allowable algorithmic depth, m_k is the algorithmic depth at iteration k, and β_k is the relaxation (also called damping or mixing) parameter used on iteration k.

ALGORITHM 1.1 (Anderson acceleration). *The algorithm starts with a fixed point update step ($k=0$); then the acceleration starts at $k=1$.*

Choose initial iterate x_0 and algorithmic depth parameter m
Compute w_1, set β_0, and update $x_1 = x_0 + \beta_0 w_1$ ▷ $k=0$
1: ***for*** $k=1,\dots$ ***do*** ▷ $k>0$
2: *Compute w_{k+1}*
3: *Set $m_k = \min\{k, m\}$*
4: *Set $F_k = \big((w_{k+1} - w_k) \quad \dots \quad (w_{k+1-(m_k-1)} - w_{k-(m_k-1)})\big)$*
5: *and $E_k = \big((e_k) \quad \dots \quad (e_{k-(m_k-1)})\big)$*
6: *Find $\gamma_k = \operatorname{argmin}\|F_k\gamma - w_{k+1}\|$*
7: *Set relaxation parameter β_k*
8: *Update $x_{k+1} = x_k + \beta_k w_{k+1} - (E_k + \beta_k F_k)\gamma_{k+1}$*
9: ***end for***

Define the optimization gain θ_{k+1} by

$$\theta_{k+1} := \frac{\|F_k\gamma_k - w_{k+1}\|}{\|w_{k+1}\|}. \tag{1.1}$$

As shown in [11, 22, 23], at each iteration k, the first order term in the residual is improved (in comparison to a fixed-point iteration with the same damping factor β_k) by a factor of θ_k, but at the cost of additional higher-order terms.

It is typical but not required to interpret the minimization problem in line 6 of Algorithm 1.1 as least-squares in either the l_2 or a weighted l_2 norm [11, 13, 22, 29, 31]. However, other norms such as l_1 or l_∞ can be used [28]. Here we will restrict our attention to interpreting the minimization as a (weighted) least-squares problem, which can be efficiently solved using, for instance, a QR factorization as in [29]. This finite dimensional Hilbert space setting is used to further understand the theoretical convergence properties of AA and to improve performance through dynamic parameter selection in [22].

The main issue addressed herein is controlling the condition of the matrix F_k used in the least-squares problem in line 6, in order to improve the numerical stability of the algorithm. To this end we introduce a column filtering strategy to efficiently control the condition of F_k. In [13], both truncated singular value decomposition (TSVD) and Householder QR with column pivoting (rank revealing QR) are discussed as methods to address poorly conditioned least-squares problems that may arise in AA. Both of these standard methods for rank deficient and ill-conditioned least-squares problems may ultimately interfere with the convergence of AA, as the columns of the iteration-k least-squares matrix F_k have a natural ordering based on the algorithmic age of the information they represent. The rank revealing QR approach can change the order

of the columns, keeping older information which may pollute the solution even while improving conditioning; and the TSVD approach may reweight the columns without discarding older information to a similar effect. Another approach to this problem is to add a Tikhonov-type regularization term to the least-squares problem. This approach is used, for instance, in [25] in the context of unconstrained optimization and in [26] in the context of deep reinforcement learning. While this regularization can be used to control the size of the optimization coefficients, it can potentially lead to the selection of less effective coefficients.

We will use use the more stable TSVD approach as a standard for comparison in our numerical tests. We will demonstrate in section 4 that our presently proposed filtering algorithm which features a low overall additional computational complexity compares well in terms of the number of iterations to convergence while controlling the condition numbers; often converging in substantially fewer iterations or converging where the TSVD approach fails. The filtering approach further appears more robust with respect to parameter selection.

The remainder of the manuscript is organized as follows. In subsection 1.1 we review the background theory that supports each of the two parts of the filtering method: angle filtering and length filtering. In section 2 we introduce the filtering algorithms, which efficiently control the condition of F_k at each iteration k. In section 3 we analyze the strategy and establish that the combination of the two filtering routines, one controlling small angles between columns and the other controlling relative magnitudes of columns, is sufficient to control the condition. In section 4 the filtering strategy is tested against the standard TSVD on a number of problems with varying complexity.

In the remainder, let $\|A\|_F$ denote the Frobenius norm of A, the matrix with columns $a_1, \ldots, a_m$. Vector norms without a subscript, e.g., $\|a_i\|$, will denote the vector 2-norm.

1.1. Background theory. The proposed filtering method can be thought of as combining two approaches, each of which partially controls the condition of F_k, the matrix that arises in the least-squares problem at each iteration of AA. The first is the angle-filtering approach of [22], which controls the angle between each column of F_k and the subspace spanned by the columns to its left (which contain more recent information). Specifically, define $\mathcal{F}_j$ as the subspace spanned by the first j columns of a given matrix F_k and define the direction sine σ_i by

$$\sigma_i = \sin(\mathcal{F}_{i-1}, f_i),\ i = 2, \ldots, m_k, \text{ and } \sigma_{k,min} = \min_{i=2,\ldots,m_k} \sigma_i, \tag{1.2}$$

where f_i is column i of matrix F_k. Given a QR factorization $F_k = QR$, we have $\sigma_i = |r_{ii}|/\|f_i\|$. This quantity can then be monitored and used to filter out columns of F_k at each iteration k. The second approach is that of [1, 29], which sequentially drops the oldest columns of F_k, updating the QR decomposition and checking the condition number until the condition is within a given bound. Neither of these strategies alone is sufficient to efficiently control the condition of F_k, as the condition can become high due to near linear dependence of the columns either through disparity in lengths or alignment in angle. This work combines elements from both strategies to give a guaranteed bound on the condition number after at most a single update of the QR decomposition, while maintaining the angle condition which is necessary for the convergence analysis of [22] for smooth problems. As we show numerically in section 4, filtering is beneficial to convergence in both the preasymptotic and the asymptotic regimes.

As in [1, 29], one part of this filtering method drops older information from the system, which reduces the buildup of higher-order (nonlinear) residual terms that can be seen in the analysis of [22]. Assuming the fixed-point operator is sufficiently smooth, with Lipschitz constant κ_g and Lipschitz constant of its derivative $\hat{\kappa}_g$, Theorem 5.1 of [22] proves that the step k residual w_{k+1} satisfies

$$\|w_{k+1}\| \leq \|w_k\| \left\{ \theta_k((1-\beta_k)+\kappa_g\beta_k) + C_k\hat{\kappa}_g\sqrt{1-\theta_k^2} \sum_{n=k-m_{k-1}}^{k} \|w_n\| \right\}, \tag{1.3}$$

where at each iteration k the quantity C_k depends on the previous m_k values of θ_j and β_j (introduced in Algorithm 1.1 and (1.1)) and increases as each of the previous m_k values of $\sigma_{j,min}$ from (1.2) decreases. From this estimate, we observe that in a converging iteration the largest of the higher-order terms will be from older information, i.e., $\|w_k\|\|w_{k-m_{k-1}}\|$, and can be (possibly significantly) larger than $\|w_k\|^2$. Hence strategies to keep C_k small should also weigh the effect that older information may have on the residual.

The length filtering algorithm introduced in subsection 2.2 is based on columnwise bounds for the condition number of a matrix F, assuming a lower bound on its minimal direction sine between columns as defined by (1.2). Computing the Frobenius norm of F columnwise is both straightforward and sharp. Our bounds for $\|F^{-1}\|_F = \|R^{-1}\|_F$ for $F = QR$ are based on the two following results for general rectangular matrices.

The first is a componentwise bound on the entries of R^{-1} from [22].

LEMMA 1.1 (see [22, Lemma 5.1). *Let $F = QR$ be an economy QR decomposition of $n \times m$ matrix F, with $n \geq m \geq 2$. Let $\mathcal{F}_p = \text{span}\{f_1, \ldots, f_p\}$, the subspace spanned by the first p columns of F. Suppose there is a constant $0 < c_s \leq 1$ such that $\sin(f_i, \mathcal{F}_{i-1}) \geq c_s$, by which there is another constant $c_t = \sqrt{1-c_s^2}$ for which $\cos(f_i, f_k) \leq c_t$, for $k = 1, \ldots, i$. Denote $R^{-1} = (s_{ij})$. Then it holds that*

$$s_{11} = \frac{1}{\|f_1\|}, \qquad |s_{1j}| \leq \frac{c_t(c_t+c_s)^{j-2}}{\|f_1\|c_s^{j-1}},\ 2 \leq j \leq m, \text{ and} \tag{1.4}$$

$$s_{ii} \leq \frac{1}{\|f_i\|c_s},\ 2 \leq i \leq m, \qquad |s_{ij}| \leq \frac{c_t(c_t+c_s)^{j-i-1}}{\|f_i\|c_s^{j-i+1}},\ i+1 \leq j \leq m. \tag{1.5}$$

Next, summing by column over the square of the bounds from Lemma 1.1 gives the following bounds on the squared l_2-norm of each column of R^{-1}.

PROPOSITION 1.2. *Suppose the hypotheses of Lemma 1.1. Let s_i denote column i of R^{-1}. The the following bounds hold:*

$$\|s_1\|^2 = \frac{1}{\|f_1\|^2} =: b_1, \tag{1.6}$$

$$\|s_2\|^2 \leq \frac{1}{c_s^2}\left\{ \frac{c_t^2}{\|f_1\|^2} + \frac{1}{\|f_2\|^2} \right\} =: b_2, \tag{1.7}$$

$$\|s_j\|^2 \leq \frac{1}{c_s^2}\left(\frac{c_t^2(c_t+c_s)^{2(j-2)}}{\|f_1\|^2 c_s^{2(j-2)}} + \sum_{i=2}^{j-1} \frac{c_t^2(c_t+c_s)^{2(j-i-1)}}{\|f_i\|^2 c_s^{2(j-i)}} + \frac{1}{\|f_j\|^2} \right\} =: b_j, \quad 3 \leq j \leq m. \tag{1.8}$$

In the length filtering algorithm, the sums of the squared norms of the first l columns of f, multiplied by the sum over b_j, $j = 1, \ldots, l$, as defined in (1.6)–(1.8), are used to bound the square of the Frobenius condition number of F, truncated to its first l columns.

In summary, the (angle) filtering algorithm introduced in [22] removes columns of F_k which are close in angle to the subspace spanned by more recent columns. This controls the constant C_k that multiplies higher-order residual terms of (1.3) as shown in [22, Theorem 5.1]. However, alone this method is not sufficient to control the condition of the least-squares matrix F_k, since poor condition can also arise from large magnitude differences between the lengths of the columns of F_k. The full condition filtering algorithm presented here consists of the length- filtering algorithm (Algorithm 2.5) followed by the angle-filtering algorithm (Algorithm 2.4).

2. Filtered AA. The filtered Anderson acceleration (FAA) algorithm presented here consists of two filtering steps, the first filtering for disparity in column lengths, and the second filtering for small angles between each column and the subspace spanned by the columns to its left. The filtered acceleration algorithm is presented in Algorithm 2.1. The individual filtering routines are described in Algorithm 2.4 and the new algorithm (Algorithm 2.5). The algorithm removes columns from F_k at each iteration $k > 1$ to ensure $\text{cond}_F(F_k) \leq \bar{\kappa}$, where $\bar{\kappa}$ is a user-chosen upper bound on the allowable condition number.

The FAA algorithm (Algorithm 2.1) requires two additional parameters in comparison to the standard AA algorithm (Algorithm 1.1). The first is $\bar{\kappa}$, a user-determined upper bound on the condition of each matrix F_k used in the least-squares problem. The second is c_s, a lower bound on the allowable direction sines between each column of each matrix F_k and the subspace spanned by columns with more recent information, as defined in (1.2). A discussion on parameter selection is included in subsection 2.1.

ALGORITHM 2.1 (Filtered Anderson acceleration). *The algorithm starts with a fixed-point update step* ($k = 0$) *and an* $AA(1)$ *step* ($k = 1$)*; then the filtering starts at* $k = 2$.

Choose initial iterate x_0 *and parameters* c_s, $\bar{\kappa}$, *and* m
Compute w_1, *set* β_0, *and update* $x_1 = x_0 + \beta_0 w_1$ ▷ $k = 0$
Compute w_2 ▷ $k = 1$
Set $F_1 = \begin{pmatrix} w_2 - w_1 \end{pmatrix}$ *and* $E_1 = \begin{pmatrix} x_1 - x_0 \end{pmatrix}$
Compute γ_2 *by* $F_1^* F_1 \gamma_2 = F_1^* w_2$
Set $m_1 = 1$, *set* β_1, *and update* $x_2 = x_1 + \beta_1 w_2 - (E_1 + \beta_1 F_1)\gamma_2$
1: ***for*** $k = 2, \ldots$ ***do*** ▷ $k > 1$
2: *Compute* w_{k+1}
3: *Update* $m_k = \min\{m_{k-1} + 1, m\}$, *and drop the last columns of* E_{k-1},F_{k-1} *if* $m_{k-1} = m$
4: *Set* $F_k = \begin{pmatrix} (w_{k+1} - w_k) & (F_{k-1}) \end{pmatrix}$ *and* $E_k = \begin{pmatrix} (x_k - x_{k-1}) & (E_{k-1}) \end{pmatrix}$
5: $[E_k, F_k, m_k] =$ *Length Filter* $(E_k, F_k, m_k, c_s, \bar{\kappa})$
6: $[E_k, F_k, Q_k, R_k, m_k] =$ *Angle Filter* (E_k, F_k, m_k, c_s)
7: *Compute* γ_{k+1} *by* $R_k \gamma_{k+1} = Q_k^* w_{k+1}$
8: *Set* β_k
9: *Update* $x_{k+1} = x_k + \beta_k w_{k+1} - (E_k + \beta_k F_k)\gamma_{k+1}$
10: ***end for***

Remark 2.2 (TSVD). We compare our numerical results with Algorithm 2.1 against the standard method of controlling the condition number by TSVD, as suggested in [13]. Following [7] we implement the TSVD by computing the economy QR factorization $F_k = QR$, followed by the SVD of $R = U_R \Sigma_R V_R^*$, determine the largest s such that the ratio of singular values $\sigma_1/\sigma_s < \bar{\kappa}$, and then restrict U_R and V_R to their first s columns and σ_R to its first s rows and columns, referred to, respectively, as U_s, V_s, and Σ_s. The solution of the least-squares problem $\gamma_{k+1} = \text{argmin}\|F_k\gamma - w_{k+1}\|$ is then given by $\gamma_{k+1} = V_s \Sigma_s^{-1} U_s^* Q^* w_{k+1}$. This procedure replaces lines 5–7 of Algorithm 2.1.

Remark 2.3 (on the order of length and angle filtering). Algorithm 2.1 does length filtering before angle filtering to potentially reduce the size of the QR factorization. However, the order of the filtering steps could be reversed, provided an additional step to truncate the factors of the QR decomposition is performed after the length filtering. Each order has its own advantage, as discussed in Remark 3.1, which specifies how Theorem 3.1 still holds in the case of angle filtering performed first.

2.1. Angle filtering. The angle-filtering algorithm was proposed by the authors in [22] in order to determine a one-step residual bound for smooth problems, free from an assumed l_∞ bound on the optimization coefficients.

ALGORITHM 2.4 $(E, F, Q, R, m = \text{Angle filtering}(E, F, m, c_s))$. *Filter out columns of F by direction sine.*

1: *Compute the economy QR decomposition $F = QR$*
2: *Compute $\sigma_i = |r_{ii}|/\|f_i\|$, $i = 2, \ldots, m$, where f_i is column i of F, and r_{ii} is the corresponding diagonal entry of R*
3: *Remove any columns i of E and F for which $\sigma_i < c_s$*
4: *Update m with the new number of columns of F*
5: ***if** any columns were removed **then***
6: *Recompute $F = QR$*
7: ***end if***

Algorithm 2.1 has two additional parameters to choose, in comparison to standard AA. One is $\bar{\kappa}$, the maximum allowable condition number. The second is c_s, the minimum allowable direction sine between columns of F_k. The condition estimate in the length-filtering algorithm is less sharp for smaller values of c_s. However, if c_s is chosen too large, the angle-filtering algorithm (Algorithm 2.4) can reduce the algorithm to standard AA with $m = 1$ (see subsection 4.4 below). We generally found a range for c_s between 0.1 and $1/\sqrt{2}$ to work well. In problems where the solution progresses through distinct preasymptotic and asymptotic regimes, as shown below in subsection 4.5, it can also be advantageous to start with a higher value of c_s and dynamically decrease it to a lower value. Heuristically this makes sense because a more stringent angle-filtering condition can be advantageous in the preasymptotic regime, as it prevents the buildup of higher-order residual terms. However, in the asymptotic regime, a smaller value of c_s may be preferred because the more recent information is kept, and older columns in F_k with orders of magnitude difference in length from the first are discarded. Our numerical examples show that the filtering algorithm (Algorithm 2.1) also decreases the sensitivity of AA to the choice of maximum algorithmic depth m, as m can be chosen quite large, and extra columns will simply be (length) filtered out.

2.2. Length-filtering algorithm. Here we present a novel length-filtering strategy. It is motivated by the bounds on columns of R^{-1} from Proposition 1.2. In this algorithm, the Frobenius norm of the first k columns of F multiplied by the sum over

the bounds of the squared norms of each column of R^{-1}, denoted as b_j, $j = 1, \ldots, k$, as defined in (1.6)–(1.8), are used to bound the square of the Frobenius condition number of F, denoted in the algorithm as C_F. From m_k down to 1, the constant C_F determined by the first k columns of F is updated, until it is less than a desired (user-chosen) number. On terminating this loop with some number of $\bar{k} \geq 1$ columns remaining, the columns $j > \bar{k}$ are removed from the matrices E and F.

The combination of length and angle filtering is further justified in the next section by the main theorem (Theorem 3.1), which shows that the combination of the length-filtering algorithm (Algorithm 2.5) and the angle-filtering algorithm (Algorithm 2.4) controls the Frobenius condition number of the least-squares matrix F_k at each iteration k of the FAA algorithm (Algorithm 2.1).

ALGORITHM 2.5 ($E, F, m =$ Length filtering(E, F, m, c_s)). *Filter out columns of F by length.*

1: *Define* $c_t = \sqrt{1 - c_s^2}$
2: *Compute the vector norms* $\|f_j\|^2$ *and the bounds* b_j *for* $\|s_j\|^2$*, given by* (1.6)–(1.8)*, for* $j = 1, \ldots, m$
3: ***for*** $k = m$ *down to* 1 ***do***
4: *Compute* $C_F = (\sum_{j=1}^{k} \|f_j\|^2)(\sum_{j=1}^{k} b_j)$
5: *Exit loop if* $C_F \leq \bar{\kappa}^2$
6: ***end for***
7: $m \leftarrow k$, $E \leftarrow E(:, 1:m)$, $F \leftarrow F(:, 1:m)$

3. Theory. Together, as shown below in Theorem 3.1, the two low-cost filtering strategies, Algorithms 2.4 and 2.5, bound the Frobenius condition number of each least-squares matrix that arises in AA. Let $F = QR$ be an economy QR decomposition of $n \times m$ matrix F. Then $\|R\|_F = \|F\|_F$, and $\|R\|_2 = \|F\|_2$. By the equivalence $\|F\|_2 \leq \|F\|_F \leq \sqrt{m}\|F\|_2$ [16, Chapter 2], controlling the Frobenius condition number of F controls the 2-norm condition number of F as well.

The main theoretical result is that given a minimum direction sine $0 < c_s \leq 1$ and a maximum condition number $\bar{\kappa}$, the combination of Algorithms 2.4 and 2.5 ensures the matrix F_k from Algorithm 2.1 has a Frobenius condition number no greater than $\bar{\kappa}$ at each iteration k. We are concerned with the case $n \gg m$, meaning the number of degrees of freedom is some orders of magnitude larger than the algorithmic depth $m = m_k$. In this setting, an additional computational cost of Algorithm 2.5 is negligible.

THEOREM 3.1. *Suppose the FAA algorithm (Algorithm* 2.1*) is run with a given $0 < c_s < 1$ and $\bar{\kappa} > 1$. Then, at each iteration k, the matrix F_k of the least-squares problem solved in step* 7 *of the FAA algorithm (Algorithm* 2.1*) by the QR decomposition has a Frobenius and hence a* 2*-norm condition number less than $\bar{\kappa}$. Moreover, both filtering steps are guaranteed to keep the first column of F_k, preserving the use of the most recent information.*

Proof. By the standard inequality $\|F\|_2 \leq \|F\|_F$ [16, Chapter 2], it is sufficient to establish the result for the Frobenius condition number. First we have

$$\text{cond}(F)_F^2 = \|F\|_F^2 \|F^{-1}\|_F^2 = \|F\|_F^2 \|R^{-1}\|_F^2 = \left(\sum_{i=1}^{m} \|f_i\|^2 \right) \|R^{-1}\|_F^2, \tag{3.1}$$

where m is the number of columns of F.

Suppose the length-filtering algorithm (Algorithm 2.5) is done first. Then, since the bounds on column j of R^{-1} in Proposition 1.2 given by (1.4)–(1.5) depend only

on information with indices $p \leq j$, removing columns p of F for which $p > j$ does not change the column sum of index j. Hence the matrix $\hat{F}$ with $\hat{m}$ columns returned from the length- filtering algorithm (Algorithm 2.5) and sent to the angle-filtering algorithm (Algorithm 2.4) would have condition number less than $\bar{\kappa}$ if it satisfied the angle conditions $\sin(f_i, \mathcal{F}_{i-1})$, $i = 2, \ldots, \hat{m}$. The first index $i = 1$ is not included in the previous statement, as the angle condition is satisfied on the first column by default. Moreover, Algorithm 2.5 will never remove the first column of F so long as $\bar{\kappa} \geq 1$ as $k = 1$ yields $C_F = 1$. The angle-filtering algorithm (Algorithm 2.4) may remove any of columns $2, \ldots \hat{m}$ of $\hat{F}$, which would only decrease the bounds in proposition 1.2 by removing terms. Hence the condition estimate holds. ∎

Remark 3.1. If the angle-filtering algorithm is done first, then Theorem 3.1 still holds. In this case, the hypotheses of Lemma 1.1 are satisfied, and by (3.1) and Proposition 1.2, the length-filtering algorithm (Algorithm 2.5) produces an output matrix with $\text{cond}(F)_F < \bar{\kappa}$. To update the QR after length filtering, the QR factors can simply be truncated to agree with the updated F. In particular,

$$R = \begin{pmatrix} R_{11} & R_{12} \\ & R_{22} \end{pmatrix} \text{ and } Q = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix},$$

where R_{11} and Q_1 correspond to the first k columns of F which are kept by the length-filtering algorithm (Algorithm 2.5). Then it suffices to replace R by R_{11} and Q by Q_1 (or $Q^* w_{k+1}$ by the corresponding entries of $Q_1^* w_{k+1}$ if Q itself is not stored).

Although the two filtering steps could be run in either order, we run the length filtering first because it has lower computational complexity and potentially reduces the number of columns used in the QR factorization(s) of the angle-filtering algorithm (Algorithm 2.4). If a sharper condition estimate is preferred, the angle filtering could be run first, after which parameter c_s could be replaced by c_s', the smallest realized value of $\sin(f_i, \mathcal{F}_{i-1}) = |r_{ii}|/\|f_i\|$, where i indexes over the columns remaining after angle filtering. Since $c_s' \geq c_s$, the result is a sharper estimate of the condition from length filtering. In our tests (not shown), this reordering did not make substantial impact on the total iterations to convergence.

4. Numerics. In the following numerical tests, the FAA algorithm (Algorithm 2.1) is tested against Anderson with the TSVD applied to the least-squares problem as in Remark 2.2. The first three examples show the advantages of filtering in more complex applications. The last two examples give insight into how filtering works in the preasymptotic vs. asymptotic regimes. In particular, as seen in subsection 4.4, the length-filtering algorithm (Algorithm 2.5) dominates in the asymptotic regime, where the condition of the least-squares matrix can be compromised by the disparity in the column lengths as the algorithm converges. The angle-filtering algorithm (Algorithm 2.4), on the other hand, stabilizes the algorithm throughout the preasymptotic regime and may be an important tool in convergence from poor initial iterates.

4.1. Nonlinear Helmholtz equation. We consider first the approximation of solutions to the nonlinear Helmholtz (NLH) equation from optics, where the interest is the propagation of continuous-wave laser beams through transparent dielectrics. The NLH system in one dimension is written as follows: Find $u : [0, 10] \to \mathbb{C}$ satisfying

$$u_{xx} + k_0^2 \left(1 + \epsilon(x)|u|^2\right) u = 0, \quad 0 < x < 10,$$

where $u = u(x)$ represents the unknown (complex) scalar electric field, k_0 denotes the linear wavenumber, and $\epsilon(x)$ is a material dependent quantity involving the linear index of refraction and the Kerr coefficient. The physically consistent two-way boundary condition is derived in [14, 5] and is given by

$$u_x + ik_0 u = 2ik_0 \text{ at } x = 0,$$
$$u_x - ik_0 u = 0 \text{ at } x = 10.$$

Despite being one dimensional, this system can be quite challenging for nonlinear solvers due to its cubic nonlinearity [4, 5].

A Picard-type iteration for this NLH system takes the form

$$\begin{aligned} u_{j+1_{xx}} + k_0^2 u_{j+1} + k_0^2 \epsilon(x)|u_j|^2 u_{j+1} = 0, &\qquad 0 < x < 10, \\ u_{j+1_x} + ik_0 u_{j+1} = 2ik_0, &\qquad x = 0, \\ u_{j+1_x} - ik_0 u_{j+1} = 0, &\qquad x = 10. \end{aligned}$$

This iteration is advantageous due to its simplicity and can work directly with complex numbers. The Newton iteration, on the other hand, seems to require decomposition into the real and imaginary parts, leading to fully coupled block linear systems at each iteration. Here, the system is discretized in space with a second-order finite difference approximation using $N = 2001$ equally spaced points, and for an initial guess we use the nodal interpolant of $(\cos(k_0 x) + i \sin(k_0 x))$, which is the linear ($\epsilon = 0$) Helmholtz equation solution for the same k_0.

For our tests, we consider parameters $k_0 = 8$ and constant $\epsilon = 0.2$, and a plot of the solution is shown in Figure 1. Our computations use no relaxation ($\beta = 1$) and depth $m = 20$, and we compare both the condition number and the convergence of usual AA to FAA ($\bar{\kappa} = 10^8$ and varying $c_s = 0.1$, 0.2) to AA with TSVD (varying tolerances 10^3 and 10^8). From Figure 2 (left) we observe the convergence of each method, and it is clear that TSVD and FAA both have much better convergence than usual AA. We also observe that filtering with $c_s = 0.1$ provides the fastest convergence, and TSVD with both 10^3 and 10^8 tolerances and FAA with $c_s = 0.2$ all have similar convergence behavior. Somewhat larger and smaller c_s (0.05, 0.3, 0.4) were also tested, with results similar to that of $c_s = 0.2$. We observe from Figure 2 (right) that while the optimization problem of usual AA has a high condition number near the end of the iteration, the other methods behave as designed: TSVD holds the condition number below its tolerance, and FAA holds the condition number lower than TSVD with tolerance 10^3.

4.2. 2D Navier–Stokes equations. Our next test is for the 2D driven cavity problem for the steady Navier–Stokes equations (NSE), which are given on a domain $\Omega = (0, 1)^2$ by

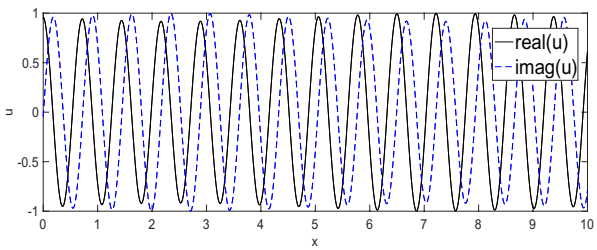

FIG. 1. *The plot above shows the solution of the nonlinear Helmholtz problem with* $k_0 = 8$ *and* $\epsilon = 0.2$.

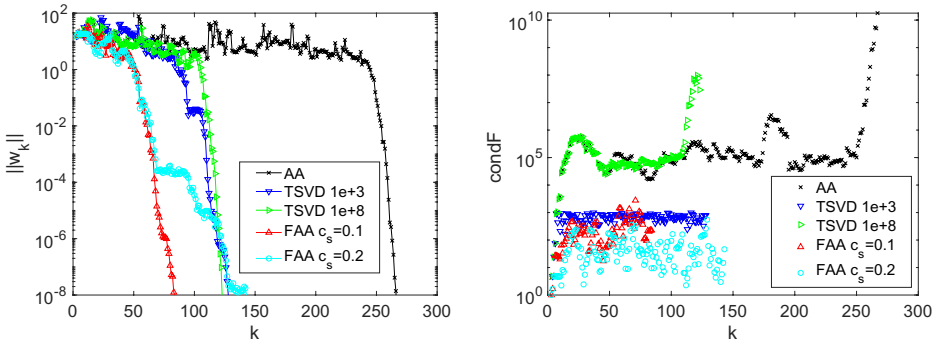

FIG. 2. *Shown above are convergence plots (left) and condition number plots (right) for AA with $m = 20$ applied to the nonlinear Helmholtz problem with varying parameters for TSVD and FAA.*

$$\begin{aligned} u \cdot \nabla u + \nabla p - Re^{-1}\Delta u &= f, \\ \nabla \cdot u &= 0, \\ u|_{\partial\Omega} &= u_{bc}, \end{aligned}$$

where u and p are the unknown velocity and pressure, f is an external forcing, u_{bc} is a given (Dirichlet) boundary condition that is 0 on the sides and bottom and $\langle 1, 0\rangle^T$ on the lid, and Re is the Reynolds number.

The Picard iteration for the steady NSE is given by (suppressing the spatial discretization)

$$\begin{aligned} u_j \cdot \nabla u_{j+1} + \nabla p_{j+1} - Re^{-1}\Delta u_{j+1} &= f, \\ \nabla \cdot u_{j+1} &= 0, \\ u_{j+1}|_{\partial\Omega} &= u_{bc}, \end{aligned}$$

and we use an initial guess of zero $u_0 = 0$ (which implies that u_1 is the Stokes solution with this problem data). The iteration is enhanced with $m = 50$ AA and no relaxation ($\beta = 1$). As found in [22, 24], large m (even $m = k - 1$) for steady NSE problems tends to work well and often better than smaller m when Re is large. We note that for this test, $m = 0$ gave no convergence, $m = 1$ gave slow convergence, and $m \geq 10$ gave convergence similar to $m = 50$ (tests omitted). A key advantage of the filtering strategy proposed herein is that one can choose large m and if m is "too large" on a particular iteration (large condition number of F), then m is effectively reduced in an optimal manner (with respect to the optimization problem).

The spatial discretization is grad-div stabilized (with parameter 1) (P_2, P_1) Taylor–Hood elements on a triangulation created from a uniform 100×100 triangulation that is further refined once around the edges (within 0.1 from the boundary), which provides a total of 190,643 degrees of freedom; a plot of the mesh is shown in Figure 3 (right).

Results for convergence and condition number versus iteration are given in Figure 4 for usual AA, filtering with $\bar{\kappa} = 10^8$ and $c_s = 0.2$ and $1/\sqrt{2}$, and TSVD with tolerances 10^3 and 10^8. We observe that usual AA has a very large condition number near the end of the iteration and that both TSVD and FAA can control this. We observe good convergence with all methods, except that TSVD with tolerance 10^3 displays some erratic behavior and converges somewhat slower than the other methods. Only filtering with $c_s = 1/\sqrt{2}$ beat usual AA, although only slightly.

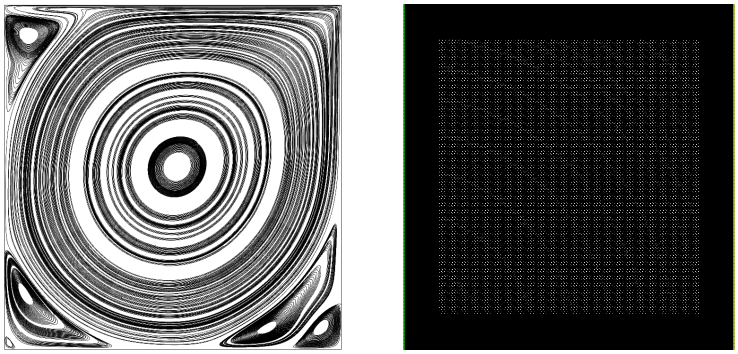

FIG. 3. *The plot above shows the solution of the steady Navier-Stokes 2D driven cavity problem at* $Re = 10,000$ *(left) and the mesh used in our numerical tests (right).*

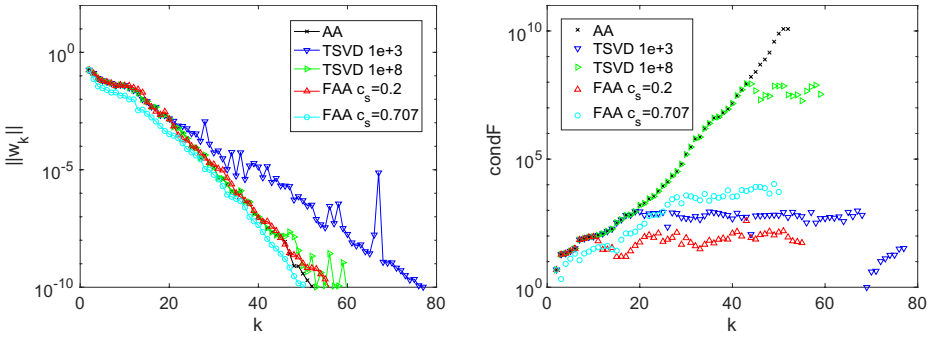

FIG. 4. *Shown above are convergence plots (left) and condition number plots (right) for the 2D Navier-Stokes driven cavity test problem with varying parameters for TSVD and filtering.*

4.3. Gross–Pitaevskii equations. For our next test, we consider the Gross–Pitaevskii equations (GPE). In addition to demonstrating how well FAA works on another important application problem, this test will show how filtering naturally works in conjunction with the dynamic depth techniques of [22]. The GPE are given by

$$\begin{aligned} \mu\phi(x) &= -\frac{1}{2}\Delta\phi(x) + V(x)\phi(x) + \eta|\phi(x)|^2\phi(x), \quad x \in \Omega, \\ \phi(x) &= 0, \quad x \in \partial\Omega, \\ \int_\Omega |\phi(x)|^2 \, dx &= 1, \end{aligned}$$

where V is a given trapping potential of the form $V(x) = \frac{1}{2}(\gamma_1^2 x_1^2 + \cdots + \gamma_d^2 x_d^2)$ with $\gamma_i > 0$ for all i and real parameter η, and ϕ is the unknown and the eigenvalue μ can be calculated as

$$\mu = \int_\Omega \left(\frac{1}{2}|\nabla\phi|^2 + V|\phi|^2 + \eta|\phi|^4 \right) \, dx.$$

This system models stationary solutions of the nonlinear Schrödinger (NLS) equation, which is also commonly referred to as the the nonrotational GPE in the context of Bose–Einstein condensates (BEC) [17, 21, 3]. In the GPE setting, ϕ represents

the macroscopic wave function of the condensate and the parameter η being positive/negative represents attraction/repulsion of the condensate atoms in GPE.

We apply AA with and without TSVD, and FAA to a FEM discretization of the Picard-projection iteration from [15], which we call PP_h and is defined by

PP_h Step 1: Given $\phi_k \in X_h$ with $\|\phi_k\| = 1$, find $\hat{\phi}_{k+1} \in X_h$ satisfying

$$\frac{1}{2}(\nabla\hat{\phi}_{k+1}, \nabla\chi) + (V\hat{\phi}_{k+1}, \chi) + \eta(|\phi_k|^2\hat{\phi}_{k+1}, \chi) = (\phi_k, \chi).$$

PP_h Step 2: Calculate

$$\phi_{k+1} = \frac{\hat{\phi}_{k+1}}{\|\hat{\phi}_{k+1}\|}.$$

We consider PP_h as a fixed-point iteration solver for a 2D test from [3] that represents a harmonic oscillator potential together with a potential from a stirrer that corresponds to a far-blue detuned Gaussian laser beam, i.e.,

$$V(x,y) = \frac{1}{2}\left(x^2 + y^2\right) + 4e^{-\left((x-1)^2+y^2\right)}.$$

Solutions are computed on $\Omega = (-8,8)^2$ using η =10,000 and initial guess

$$\phi_0(x,y) = \frac{1}{\pi^{1/2}}e^{-(x^2+y^2)/2},$$

using a uniform 128 x 128 triangulation and globally continuous P_2 elements. A plot of the solution is shown in Figure 5 as a surface plot, which agrees well with the literature [15, 3].

PP_h is run with AA $m = 1$ until the residual is below 10^{-2}, after which $m = 20$; no relaxation is used. This multilevel depth strategy of small m early and large m late in the iteration proposed in [22] (and motivated by their analytical convergence analysis) was found effective in [15] for this test problem, and in fact we find that if $m \geq 10$ is used for the entire iteration, then we do not get convergence (a plot of failed convergence for constant $m = 20$ AA is shown in Figure 6, and we note that AA with constant $m = 20$ neither TSVD nor filtering with the same parameters as below provided for convergence).

Tests are run with multilevel $m = 1$ and then $m = 20$ AA with no filtering, FAA with $\bar{\kappa} = 10^8$ and c_s =0.2:0.1:0.7, and TSVD with tolerances 10^3 and 10^8. Plots of convergence and condition numbers for these tests are shown in Figure 6, and we observe that usual AA, TSVD with tolerances 10^8 and 10^3, and FAA with c_s =0.2

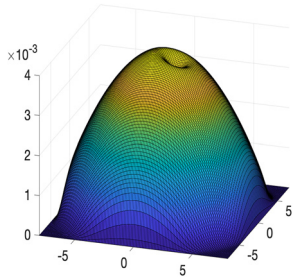

FIG. 5. *Shown above is a surface plot of the converged ground state solution using* $\eta = 10,000$.

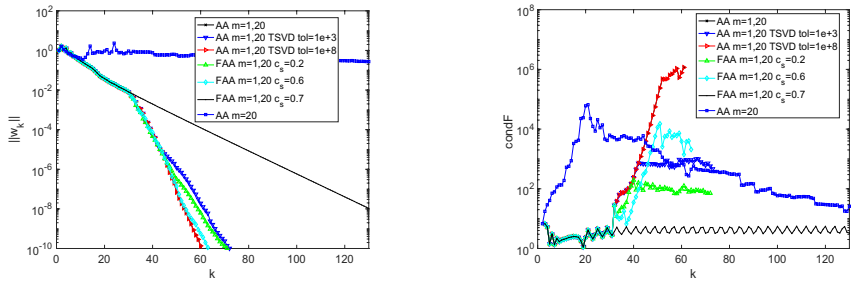

FIG. 6. *Shown above are convergence plots (left) and condition number plots (right) for the Gross–Pitaevskii test problem.*

and 0.6 all had similar good convergence ($c_s = 0.3$, 0.4, and 0.5 converged similarly as well; plots omitted). FAA with c_s =0.7 converged significantly slower, and upon inspection we found that this filtering effectively reduced the AA to $m = 1$ at each step. AA with constant $m = 20$ failed to converge, with or without filtering and TSVD (at least with the same parameters as above).

4.4. A monotone quasi-linear equation. This example of an iteration for a finite element discretization of monotone quasi-linear problems, adapted from [8, Example 1], illustrates the convergence behavior of filtering in the asymptotic regime. We consider the PDE

$$-\mathrm{div}(\mu(|\nabla u|)\nabla u) = f \ \text{ in } \Omega = (0,1)\times(0,1), \quad u = 0 \text{ on } \partial\Omega, \tag{4.1}$$

$$\text{with} \quad \mu(|\nabla u|) = 1 + \arctan(|\nabla u|), \quad f = \pi. \tag{4.2}$$

In our experiments the solution space V_h consists of C^0 piecewise quadratic (P_2) finite elements on a uniform triangularization of the unit square with right triangles and 256 subdivisions along each of the x- and y-axes, for a total of 263,169 degrees of freedom. Each update step w_{k+1} is found by the solution to the following linear problem: Find $w_{k+1} \in V_h$ such that

$$(\nabla w_{k+1}, \nabla v) = (f, v) - (\mu(|\nabla u_k|)\nabla u_k, \nabla v) \quad \text{for all } v \in V_h. \tag{4.3}$$

By the analysis of [8], the fixed-point iteration $g(u_k) = u_k + \beta w_{k+1}$ is contractive for $0 < \beta \le \beta^* := (1 + \sqrt{3}/2 + \pi/3)^{-2}$. Accelerating the fixed-point iteration written in terms of a constant damping factor β is the same as accelerating the original fixed-point iteration $g(u_k) = u_k + w_{k+1}$ using $\beta_k = \beta$ for each iteration k in either the original AA or the FAA algorithm (Algorithm 2.1). We consider iterations both with and without the damping factor β sufficiently small. In Table 1 we show the number of iterations to residual convergence $\|w_k\| < 10^{-10}$ from the starting guess of $u_0 = 0$. We compare the filtering algorithm with maximum condition number $\bar{\kappa} = 10^8$ with the TSVD algorithm with maximum condition number up to 10^8.

For the contractive iteration with $\beta = \beta^*$, we see that the TSVD approach for limiting the condition number overall slows convergence as the maximum allowed condition number $\bar{\kappa}$ is decreased. The filtering approach also limits the condition number but improves convergence, so long as the minimum allowable angle c_s between columns of the least-squares matrix F_k in the angle-filtering algorithm (Algorithm 2.5) is not too large. In this case $c_s = 2^{-1/2}$ is too large and restricts the algorithmic depth after filtering to $m = 1$. Results for the FAA algorithm (Algorithm 2.1) are

TABLE 1

The number of iterations to residual convergence $\|w_k\| < 10^{-10}$ for the monotone quasi-linear problem (4.1) with iteration (4.3), using FAA with $c_s = 0.1, 0.4, 2^{-1/2}$, and the TSVD with different maximum condition numbers $\bar{\kappa}$ and $m = 5, 10, 20, 40$. Without acceleration, the iteration with $\beta = \beta^*$ converges in 175 iterations, and the iteration with $\beta = 1$ does not converge.

	$\beta = \beta^*$				$\beta = 1$			
FAA, $\bar{\kappa} = 10^8$	$m = 5$	$m = 10$	$m = 20$	$m = 40$	$m = 5$	$m = 10$	$m = 20$	$m = 40$
$c_s = 0.1$	32	27	27	27	21	20	20	20
$c_s = 0.4$	31	31	31	31	21	21	21	21
$c_s = 2^{-1/2}$	96	96	96	96	22	23	23	23
TSVD	$m = 5$	$m = 10$	$m = 20$	$m = 40$	$m = 5$	$m = 10$	$m = 20$	$m = 40$
$\bar{\kappa} = 10^2$	42	57	77	100	45	64	119	238
$\bar{\kappa} = 10^3$	35	45	68	96	32	38	80	164
$\bar{\kappa} = 10^4$	30	40	64	91	30	33	63	121
$\bar{\kappa} = 10^6$	33	36	58	96	30	26	45	85
$\bar{\kappa} = 10^8$	33	38	64	92	30	22	35	51

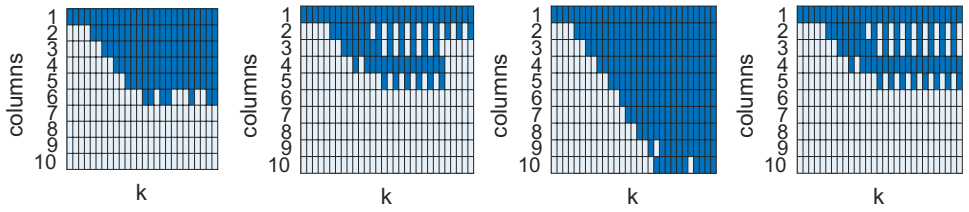

FIG. 7. *The columns used after filtering for the monotone quasi-linear problem (4.1) with iteration (4.3) with $m = 10$ and $\beta = \beta^*$. Column 1 is always used, and column 2 may be filtered out in angle filtering. From left to right: $c_s = 0.1, \bar{\kappa} = 10^8$; $c_s = 0.4, \bar{\kappa} = 10^8$ (as in Table 1); $c_s = 0.1, \bar{\kappa} = 10^{16}$; $c_s = 0.4, \bar{\kappa} = 10^{16}$.*

only shown here for the maximum condition number $\bar{\kappa}$ shown for the TSVD since the length-filtering algorithm (Algorithm 2.5) overestimates the condition and hence as shown in the previous examples actually restricts the condition number to several orders of magnitude less than $\bar{\kappa}$. The agreement in the iteration counts for filtering as algorithmic depth m is increased reflects the length-filtering algorithm cutting off older columns from F_k so that increasing m does not actually change the iteration after a certain point, whereas the TSVD continues to use all columns available which can slow convergence.

Filtering in the contractive setting is further illustrated with Figure 7, which shows the columns of F_k used at each iteration k with the filtering approach with algorithmic depth $m = 10$. The left two plots illustrate the columns of F_k used for the data shown in Table 1 with $\bar{\kappa} = 10^8$ and $c_s = 0.1$ (leftmost) and $c_s = 0.4$ (to its right). The case with $c_s = 2^{-1/2}$ is not shown because it simply restricts the iteration to $m = 1$ throughout the entire iteration. The right two plots in Figure 7 show the columns of F_k used for $c_s = 0.1$ and $c_s = 0.4$ (rightmost) with $\bar{\kappa} = 10^{16}$. In this case we see for $c_s = 0.1$ that the difference between the lower and higher condition bounds is that the length filtering in the second half of the iteration is suppressed for $\bar{\kappa} = 10^{16}$, although this does not have much of an effect on the total number of iterations. For $c_s = 0.4$, the angle filtering, which filters out earlier columns, dominates throughout both iterations.

For damping factor $\beta = 1$, the iteration defined by update step (4.3) is not contractive, and the iteration does not converge without acceleration. It is interesting

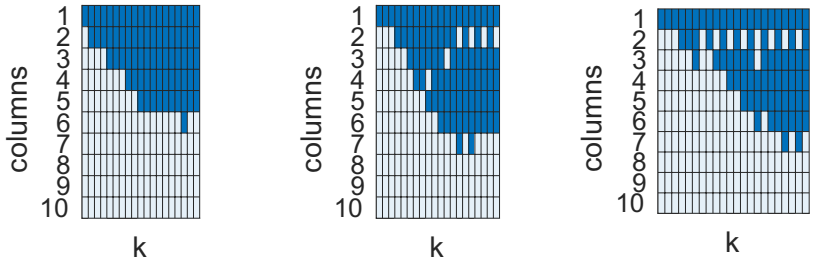

FIG. 8. *The columns used after filtering for the monotone quasi-linear problem* (4.1) *with iteration* (4.3) *with* $m = 10$, $\beta = 1$, *and* $\bar{\kappa} = 10^8$, *as in Table* 1. *Column* 1 *is always used, and column* 2 *may be filtered out in angle filtering. Left:* $c_s = 0.1$, *center:* $c_s = 0.4$, *right:* $c_s = 2^{-1/2}$.

to compare the iteration counts for $c_s = 2^{-1/2}$ for $\beta = \beta^*$ and $\beta = 1$. For $\beta = 1$, the stricter requirement on the angles between columns of F_k does not make a large impact on convergence. As seen in the rightmost plot of Figure 8, only the second column is regularly filtered out. This illustrates that smaller damping factors can actually cause alignment of columns of F_k, which can interfere with both convergence and the condition of the least-squares problem. Comparing with the TSVD in Table 1, the filtered algorithm converges after fewer total iterations, and is robust across parameter ranges tested, whereas the TSVD shows similar performance only when the right number of columns $m = 10$ is used and when the allowable condition number is high enough. The second interesting observation that we see in both Figures 7 and 8 is that the angle-filtering algorithm (Algorithm 2.4) tends to filter out the earlier (leftmost) rather than the later columns, and most often the second. We see this again in the next example for a problem where the approximation progresses through a more substantial preasymptotic regime and where a more stringent condition on the angle between columns of F_k can be very useful.

4.5. *p*-Laplace. In this example we consider finite element discretizations of the elliptic p-Laplace equation

$$-\mathrm{div}\left((|\nabla u|^2/2)^{(p-2)/2}\nabla u\right) = f \text{ in } \Omega.$$

The p-Laplace (or p-Poisson) equation arises frequently as a model problem in non-Newtonian and turbulent flows and flows in porous media [9, 10], as well as in the discretization of abstract linear operators in Banach space [19]. If $1 < p < 2$, the equation is singular, as the coefficient on the nonlinear diffusion coefficient is negative and blows up as $|\nabla u|$ goes to zero. Hence we consider a regularized equation

$$-\mathrm{div}\left(\left(\epsilon^2 + \frac{1}{2}|\nabla u|^2\right)^{(p-2)/2}\nabla u\right) = f \text{ in } \Omega \text{ with } \epsilon = 10^{-14}. \tag{4.4}$$

The equation for $1 < p < 2$ is more challenging to solve for p closer to one, and here we use $p = 1.04$, which is quite small; cf. [10, 22]. Similarly to [6, 22], we consider (4.4) over the domain $\Omega = (0,2) \times (0,2)$ with homogeneous Dirichlet boundary conditions, constant forcing $f = \pi$, and a relatively bad initial guess $u_0 = xy(x-1)(y-1)(x-2)(y-2)$ for the Picard iteration given by the following update step: Find $w_{k+1} \in V_h$ satisfying for all $v \in V_h$

$$\int_\Omega a_{\epsilon,p}(u_k)\nabla w_{k+1} \cdot \nabla v \, \mathrm{d}x = \int_\Omega f v \, \mathrm{d}x - \int_\Omega a_{\epsilon,p}(u_k)\nabla u_k \cdot \nabla v \, \mathrm{d}x, \tag{4.5}$$

TABLE 2

The number of iterations to residual convergence $\|w_k\| < 10^{-10}$ for problem (4.4) using update (4.5) discretized with Lagrange P_k elements with $k = 1, 2, 3, 4$. Results are shown using the TSVD with different maximum condition numbers $\bar{\kappa}$, and FAA with $c_s = 0.1, 2^{-1/2}$ and a dynamically chosen c_s given by (4.6). For runs that did not converge within 500 iterations, >500 demarks cases where the last 10 residuals are less than 1, suggesting the iteration might eventually converge; otherwise, iterations failing to converge are marked F. All runs used algorithmic depth $m = 10$.

FAA, $\bar{\kappa} = 10^8$	P_1	P_2	P_3	P_4
$c_s = 0.1$	F	364	273	218
$c_s = 2^{-1/2}$	198	171	203	243
c_s dynamic	142	134	152	215
TSVD	P_1	P_2	P_3	P_4
$\bar{\kappa} = 10^2$	>500	207	154	>500
$\bar{\kappa} = 10^3$	F	F	>500	F
$\bar{\kappa} = 10^4$	F	F	>500	F
$\bar{\kappa} = 10^6$	F	F	>500	F
$\bar{\kappa} = 10^8$	F	F	>500	F

with $a_{\epsilon,p}(u_k) = \left(\epsilon^2 + \frac{1}{2}|\nabla u_k|^2\right)^{(p-2)/2}$. We consider discretizations with C^0 P_k Lagrange elements for $k = 1, 2, 3, 4$, over a uniform triangularization of Ω with right triangles with 256 subdivisions in each of the x- and y-axes, for a total of 66,049 (P_1); 263,169 (P_2); 591,361 (P_3); and 1,050,625 (P_4) respective degrees of freedom. We compare the FAA algorithm (Algorithm 2.1) with both fixed and dynamically chosen parameters to the TSVD approach. The angle- filtering parameters used in this example are $c_s = 0.1$, $c_s = 2^{-1/2}$ and the dynamic choice is

$$c_s = \max\left\{\min\left\{\|w_{k+1}\|^{1/2}, 2^{-1/2}\right\}, 0.1\right\}, \tag{4.6}$$

which transitions between the smaller and larger choices of c_s based on the norm of the fixed-point update step.

Table 2 shows the number of iterations to residual convergence $\|w_k\| < 10^{-10}$ for the filtered algorithm (Algorithm 2.1). We see that the filtering algorithm is an enabling technology that allows us to solve this problem at all, whereas the TSVD approach is only successful when the condition number is strictly controlled with $\bar{\kappa} = 10^2$, and even then we only see a successful solve within 500 iterations for the P_2 and P_3 elements. The filtering method is successful in all but one case: $c_s = 0.1$ with P_1 elements.

The next observation we can make is that the angle-filtering algorithm (Algorithm 2.4) is important although not always sufficient for successful passage through the preasymptotic regime. Figure 9 shows residual convergence using the full (length plus angle) filtering approach, just angle filtering and just length filtering. The plot on the left with $c_s = 0.1$ for P_2 elements shows that the length filtering (with no condition enforced on the angles between columns of F_k) still can work well in the asympototic regime and in this case is an adequate if not efficient way through the preasymptotic regime. The center plot shows P_2 elements with $c_s = 0.5$, where with a stronger angle condition the angle filtering alone is sufficient to achieve asymptotic convergence, whereas the sharper estimate on the condition number from a larger value of c_s does not force the length filtering alone to remove enough columns to converge. The plot on the right for P_4 elements with $c_s = 0.1$ reinforces these observations, demonstrating that the combination of length and angle filtering rather than one strategy alone is more successful, in agreement with the theory.

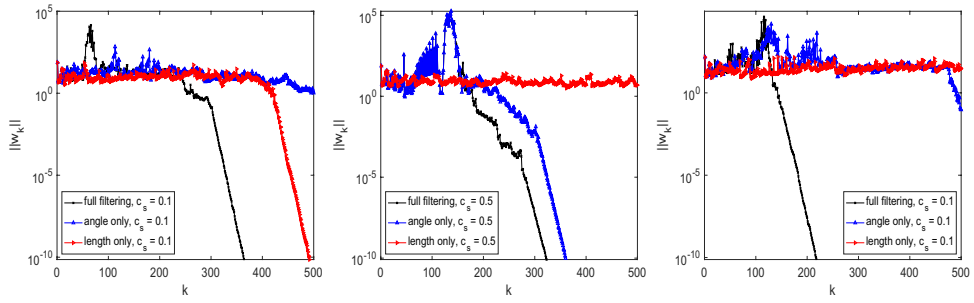

FIG. 9. *Residual convergence for* (4.4) *using update* (4.5)*, comparing full filtering, angle filtering only, and length filtering only. Left: Lagrange* P_2 *elements with parameter* $c_s = 0.1$*; center: Lagrange* P_2 *elements with parameter* $c_s = 0.5$*; right: Lagrange* P_4 *elements with parameter* $c_s = 0.1$*.*

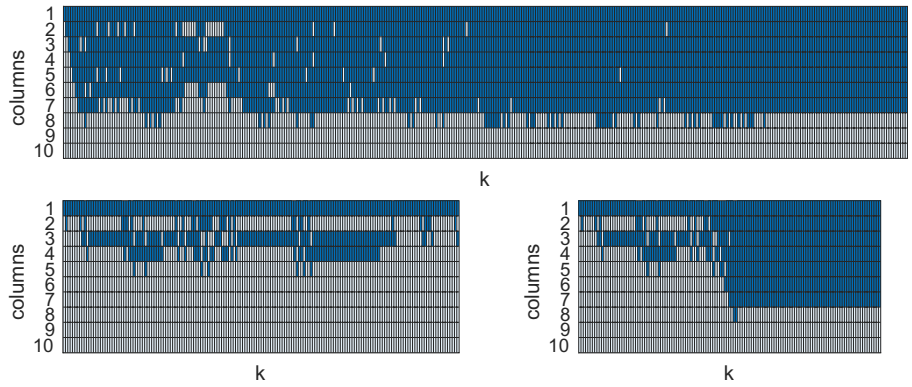

FIG. 10. *The columns used after filtering for problem* (4.4) *using update* (4.5)*, as in Table* 2 *with Lagrange* P_2 *elements. Column* 1 *is always used, and column* 2 *may be filtered out in angle filtering. Top:* $c_s = 0.1$*, bottom left:* $c_s = 2^{-1/2}$*, bottom right:* c_s *dynamically set by* (4.6)*. Each simulation uses algorithmic depth* $m = 10$*.*

We further see the importance of angle filtering in the preasymptotic regime by comparing the first plots in Figures 10 and 11, which show the columns of F_k used at each iteration k. For $c_s = 0.1$ for P_2 and respectively P_4 elements, we see less angle filtering in the P_1 elements throughout the early stages of the iteration, whereas the P_4 elements see more alignment in the columns of F_k and progress more quickly to the asymptotic regime with a small angle- filtering parameter. This is at least in part due to the residual bound (1.3), as the angle filtering not only controls the condition number, but also controls the scaling and buildup of higher-order terms in the residual, which matter more when the residual is larger and less when it is smaller. On the other hand, with a larger angle-filtering parameter $c_s = 2^{-1/2}$, the angle filtering remains heavy throughout the iteration, which ultimately slows convergence in the asymptotic regime.

For this example, where successful solves transition from a poor initial guess through a preasymptotic regime into an asymptotic regime, the dynamic choice of angle parameter c_s performs the best, as seen in Table 2. This strategy controls the accumulation and scaling of higher-order terms in the residual early in the iteration, and allows greater algorithmic depth when the residual terms are small enough not to matter, by which the improvement in the first-order term in the residual bound (1.3) from addition columns of F_k yields a faster solve.

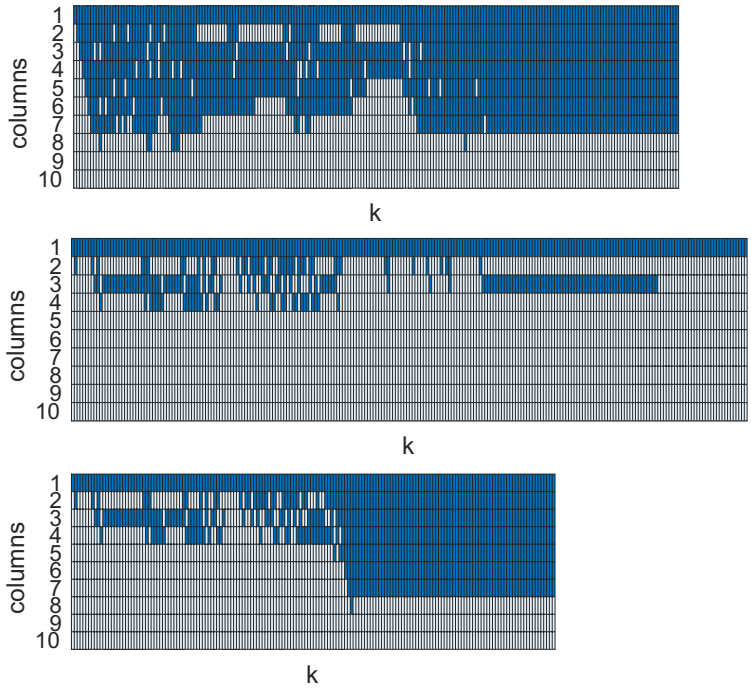

FIG. 11. *The columns used after filtering for problem* (4.4) *using update* (4.5)*, as in Table* 2 *with Lagrange* P_4 *elements. Column* 1 *is always used, and column* 2 *may be filtered out in angle filtering. Top:* $c_s = 1$*, middle:* $c_s = 2^{-1/2}$*, bottom:* c_s *dynamically set by* (4.6)*. Each simulation uses algorithmic depth* $m = 10$*.*

5. Conclusion. In this paper we developed a column-filtering strategy to control the condition of the least-squares problem in Anderson acceleration. We demonstrated theoretically that the method controls the condition number of the matrix used in the least-squares problem at each iteration of the algorithm. The filtering algorithm consists of two phases, one which filters for disparity in column length, and the second which imposes a lower bound on the angles between subspaces spanned by the columns of the least-squares matrix. The angle-filtering approach has already been demonstrated by the authors to aid convergence in [22], as it enforces a sufficient condition for controlling the scaling of higher-order terms in the residual expansion for nonlinear problems. The angle filtering is seen to be active more in the preasymptotic regime, whereas the presently introduced length-filtering approach is more active in the asymptotic regime where older columns in the least-squares matrix can be many orders of magnitude greater in length than more recent columns due to the convergence of the algorithm. The method is demonstrated on a range of problems based on discretized partial differential equations.

Overall, we see an improvement in convergence properties using the introduced FAA algorithm. A discussion of effective parameter ranges and a dynamic strategy for choosing algorithm parameters is included. Comparing to AA using TSVD, in our numerical tests FAA performed always at least as well, but sometimes substantially better. In our tests where both FAA and AA with TSVD converged in a similar number of iterations, FAA maintained a lower condition number for the least-squares problem. Future work includes integrating the filtering strategies introduced herein

with dynamic selection of the relaxation parameter to further enhance stability in the preasymptotic and efficiency in the asymptotic phases of the solution process.

Acknowledgment. The authors would like to thank the anonymous referees for providing suggestions that improved the clarity and presentation of this article.

REFERENCES

[1] H. An, X. Jia, and H. Walker, *Anderson acceleration and application to the three-temperature energy equations*, J. Comput. Phys., 347 (2017), pp. 1–19, https://doi.org/10.1016/j.jcp.2017.06.031.

[2] D. G. Anderson, *Iterative procedures for nonlinear integral equations*, J. Assoc. Comput. Mach., 12 (1965), pp. 547–560, https://doi.org/10.1145/321296.321305.

[3] W. Bao and Q. Du, *Computing the ground state solution of Bose–Einstein condensates by a normalized gradient flow*, SIAM J. Sci. Comput., 25 (2004), pp. 1674–1697, https://doi.org/10.1137/S1064827503422956.

[4] G. Baruch, G. Fibich, and S. Tsynkov, *High-order numerical method for the nonlinear Helmholtz equation with material discontinuities in one space dimension*, J. Comput. Phys., 227 (2007), pp. 820–850.

[5] G. Baruch, G. Fibich, and S. Tsynkov, *A high-order numerical method for the nonlinear Helmholtz equation in multidimensional layered media*, J. Comput. Phys., 228 (2009), pp. 3789–3815.

[6] P. R. Brune, M. G. Knepley, B. F. Smith, and X. Tu, *Composing scalable nonlinear algebraic solvers*, SIAM Rev., 57 (2015), pp. 535–565, https://doi.org/10.1137/130936725.

[7] T. F. Chan, *Algorithm* 581*: An improved algorithm for computing the singular value decomposition* [*f*1], ACM Trans. Math. Software, 8 (1982), pp. 84–88, https://doi.org/10.1145/355984.355991.

[8] S. Congreve and T. P. Wihler, *Iterative Galerkin discretizations for strongly monotone problems*, J. Comput. Appl. Math., 311 (2017), pp. 457–472, https://doi.org/10.1016/j.cam.2016.08.014.

[9] J. I. Diaz and F. De Thelin, *On a nonlinear parabolic problem arising in some models related to turbulent flows*, SIAM J. Math. Anal., 25 (1994), pp. 1085–1111, https://doi.org/10.1137/S0036141091217731.

[10] L. Diening, M. Fornasier, R. Tomasi, and M. Wank, *A relaxed Kačanov iteration for the p-Poisson problem*, Numer. Math., 145 (2020), pp. 1–34, https://doi.org/10.1007/s00211-020-01107-1.

[11] C. Evans, S. Pollock, L. G. Rebholz, and M. Xiao, *A proof that Anderson acceleration improves the convergence rate in linearly converging fixed-point methods (but not in those converging quadratically)*, SIAM J. Numer. Anal., 58 (2020), pp. 788–810, https://doi.org/10.1137/19M1245384.

[12] V. Eyert, *A comparative study on methods for convergence acceleration of iterative vector sequences*, J. Comput. Phys., 124 (1996), pp. 271–285, https://doi.org/10.1006/jcph.1996.0059.

[13] H. Fang and Y. Saad, *Two classes of multisecant methods for nonlinear acceleration*, Numer. Linear Algebra Appl., 16 (2009), pp. 197–221, https://doi.org/10.1002/nla.617.

[14] G. Fibich and S. Tsynkov, *High-order two-way artificial boundary conditions for nonlinear wave propagation with backscattering*, J. Comput. Phys., 171 (2001), pp. 632–677.

[15] D. Forbes, L. Rebholz, and F. Xue, *Anderson acceleration of nonlinear solvers for the stationary Gross-Pitaevskii equation*, Adv. Appl. Math. Mech., 13 (2021), pp. 1096–1125.

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.

[17] L. Landau and E. Lifschitz, *Quantum Mechanics: Non-relativistic Theory*, Pergamon Press, New York, 1977.

[18] P. Lott, H. Walker, C. Woodward, and U. Yang, *An accelerated Picard method for nonlinear systems related to variably saturated flow*, Adv. Water Resour., 38 (2012), pp. 92–101, https://doi.org/10.1016/j.advwatres.2011.12.013.

[19] I. Muga and K. G. van der Zee, *Discretization of linear problems in Banach spaces: Residual minimization, nonlinear Petrov–Galerkin, and monotone mixed methods*, SIAM J. Numer. Anal., 58 (2020), pp. 3406–3426, https://doi.org/10.1137/20M1324338.

[20] Y. Peng, B. Deng, J. Zhang, F. Geng, W. Qin, and L. Liu, *Anderson acceleration for geometry optimization and physics simulation*, ACM Trans. Graph., 42 (2018), pp. 1–14.

[21] L. Pitaevskii, *Vortex lines in an imperfect Bose gas,* Sov. Phys. JETP, 13 (1961), pp. 451–454.
[22] S. Pollock and L. Rebholz, *Anderson acceleration for contractive and noncontractive operators*, IMA J. Numer. Anal., 41 (2021), pp. 2841–2872.
[23] S. Pollock, L. G. Rebholz, and M. Xiao, *Anderson-accelerated convergence of Picard iterations for incompressible Navier–Stokes equations*, SIAM J. Numer. Anal., 57 (2019), pp. 615–637, https://doi.org/10.1137/18M1206151.
[24] L. Rebholz, D. Vargun, and M. Xiao, *Enabling fast convergence of the iterated penalty Picard iteration with $\mathcal{O}(1)$ penalty parameter for incompressible Navier-Stokes via Anderson acceleration*, Comput. Method Appl. Mech. Engrg., 387 (2021), pp. 1–17.
[25] D. Scieur, A. d' Aspremont, and F. Bach, *Regularized nonlinear acceleration*, in Advances in Neural Information Processing Systems, Vol. 29, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, eds., Curran Associates, Red Hook, NY, 2016, https://proceedings.neurips.cc/paper/2016/file/bbf94b34eb32268ada57a3be5062fe7d-Paper.pdf.
[26] W. Shi, S. Song, H. Wu, Y.-C. Hsu, C. Wu, and G. Huang, *Regularized Anderson acceleration for off-policy deep reinforcement learning*, in Advances in Neural Information Processing Systems, Vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, eds., Curran Associates, Red Hook, NY, 2019, https://proceedings.neurips.cc/paper/2019/file/bb1443cc31d7396bf73e7858cea114e1-Paper.pdf.
[27] A. Toth, C. Kelley, S. Slattery, S. Hamilton, K. Clarno, and R. Pawlowski, *Analysis of Anderson acceleration on a simplified neutronics/thermal hydraulics system*, in Proceedings of the ANS MC2015 Joint International Conference on Mathematics and Computation (M&C), Supercomputing in Nuclear Applications (SNA) and the Monte Carlo (MC) Method, ANS MC2015 CD, 2015, pp. 1–12, https://doi.org/10.1016/j.jcp.2016.02.012.
[28] A. Toth and C. T. Kelley, *Convergence analysis for Anderson acceleration*, SIAM J. Numer. Anal., 53 (2015), pp. 805–819, https://doi.org/10.1137/130919398.
[29] H. F. Walker and P. Ni, *Anderson acceleration for fixed-point iterations*, SIAM J. Numer. Anal., 49 (2011), pp. 1715–1735, https://doi.org/10.1137/10078356X.
[30] D. Wang, Y. He, and H. D. Sterck, *On the asymptotic linear convergence speed of Anderson acceleration applied to ADMM*, J. Sci. Comput., 88 (2021), 38, https://doi.org/10.1007/s10915-021-01548-2.
[31] Y. Yang, A. Townsend, and D. Appelö, *Anderson acceleration based on the $\mathcal{H}^{-s}$ Sobolev norm for contractive and noncontractive fixed-point operators*, J. Comput. Appl. Math., 403 (2022), 113844. https://doi.org/10.1016/j.cam.2021.113844.