SIAM J. NUMER. ANAL.
Vol. 60, No. 6, pp. 3145-3180

SECANT ACCELERATION OF SEQUENTIAL RESIDUAL METHODS FOR SOLVING LARGE-SCALE NONLINEAR SYSTEMS OF EQUATIONS*

ERNESTO G. BIRGIN† AND J. M. MARTÍNEZ‡

Abstract. Sequential residual methods try to solve nonlinear systems of equations $F(x) = 0$ by iteratively updating the current approximate solution along a residual-related direction. Therefore, memory requirements are minimal and, consequently, these methods are attractive for solving large-scale nonlinear systems. However, the convergence of these algorithms may be slow in critical cases; therefore, acceleration procedures are welcome. In this paper, we suggest employing a variation of the sequential secant method in order to accelerate sequential residual methods. The performance of the resulting algorithm is illustrated by applying it to the solution of very large problems coming from the discretization of partial differential equations.

Key words. nonlinear systems of equations, sequential residual methods, acceleration, large-scale problems

MSC codes. 65H10, 65K05, 90C53

DOI. 10.1137/20M1388024

1. Introduction.

In the process of solving many real-life problems, it is necessary to handle large-scale nonlinear systems of equations. The most obvious choice for solving these systems is Newton's method, which requires solving a possibly large and sparse linear system of equations at each iteration. Although very effective in many cases, Newton's method cannot be employed for solving very large problems when the Jacobian is unavailable or when it has an unfriendly structure that makes its factorization unaffordable. On the other hand, inexact Newton methods that solve the Newtonian linear system approximately at each iteration are usually effective [22, 25, 26]. Inexact Newton methods based on linear iterative solvers such as GMRES may need many matrix-vector products per iteration. Usually, matrix-vector products of the form $J(x)v$ are replaced with incremental quotients $[F(x+hv) - F(x)]/h$, a procedure that does not deteriorate the overall performance of GMRES [13, 53]. However, when GMRES requires many matrix-vector products for providing a suitable approximate solution to the Newtonian linear system, the number of residual evaluations per inexact Newton iteration may be big. Additional residual evaluations may also be necessary to decide acceptance of trial points at every iteration.

This state of facts led to the introduction of algorithms in which the number of residual evaluations used to compute trial points at each iteration is minimal, as well as the memory used to store directions and the computer effort of linear algebra

* Received by the editors December 28, 2020; accepted for publication (in revised form) September 9, 2022; published electronically December 16, 2022.
https://doi.org/10.1137/20M1388024
Funding: The work of the authors was supported by FAPESP grants 2013/07375-0, 2016/01860-1, and 2018/24293-0 and CNPq grants 302538/2019-4 and 302682/2019-8.

† Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Cidade Universitária, 05508-090, São Paulo, SP, Brazil (egbirgin@ime.usp.br).

‡ Department of Applied Mathematics, Institute of Mathematics, Statistics, and Scientific Computing IMECC, State University of Campinas, 13083-859, Campinas SP, Brazil (martinez@ime.unicamp.br).

calculations. DF-SANE [41] was introduced for solving large problems and has been used for solving equilibrium models for the determination of industrial prices [48], multifractal analysis of spot prices [58], elastoplastic contact problems [31, 32], and PDE equations in reservoir simulations [49], among others. Improved versions of DF-SANE were given in [40, 47]. However, the pure form of DF-SANE may be ineffective for some large problems in which it is necessary to perform many backtrackings per iteration in order to obtain sufficient descent. Therefore, on the one hand, it is necessary to investigate alternative choices of trial steps and, on the other hand, acceleration procedures are welcome.

Acceleration devices for iterative algorithms are frequent in the numerical analysis literature [1, 10, 11, 37, 62, 63]. They incorporate useful information from previous iterations instead of expending evaluations at the current one. In particular, Anderson's acceleration introduced in [1] is known to produce very good results when associated with fixed-point iterations [4, 18, 30, 37, 62], specifically those originated in self-consistent field approaches for electronic structure calculations [43, 54]. Anderson's acceleration is closely related to some quasi-Newton methods [14, 21, 30, 36, 45] and multipoint secant algorithms [2, 34, 38, 46, 51, 64]. The recent survey [12] sheds a lot of light on the properties of Anderson's acceleration, generalizations, and relations with other procedures for accelerating the convergence of sequences.

This work introduces a generalized and accelerated version of DF-SANE. The generalization consists of allowing nonresidual (although residual-related) directions. The acceleration is based on the multipoint secant idea, taking advantage of the residual-like direction steps. Global convergence results that extend the theory of DF-SANE are given.

The paper is organized as follows. Section 2 introduces the accelerated sequential residual methods. Global convergence is established in section 3. Section 4 describes the acceleration process in detail. Implementation features and numerical experiments are given in sections 5 and 6, respectively. The last section presents the conclusions.

Notation. The symbol $\|\cdot\|$ denotes the Euclidean norm. $\mathbb{N} = \{0, 1, 2, \ldots\}$ denotes the set of natural numbers. $J(x)$ denotes the Jacobian matrix of $F : \mathbb{R}^n \to \mathbb{R}^n$ computed at x. If $\{z_k\}_{k\in\mathbb{N}}$ is a sequence and $K = \{k_1, k_2, k_3, \ldots\}$ is an infinite sequence of natural numbers such that $k_i < k_j$ if $i < j$, we denote

$$\lim_{k\in K} z_k = \lim_{j\to\infty} z_{k_j}.$$

Given v and w in the unitary sphere of $\mathbb{R}^n$, we denote by $\mathrm{discur}(v, w)$ the curvilinear distance between these vectors along the geodesic that joins v and w.

2. Accelerated sequential residual methods. Given $F : \mathbb{R}^n \to \mathbb{R}^n$, consider the problem of finding $x \in \mathbb{R}^n$ such that

$$F(x) = 0. \tag{1}$$

A radical iterative approach for solving (1) is to employ only residuals as search directions. Given $\sigma > 0$, problem (1) is clearly equivalent to $x = x - \sigma F(x)$. This trivial observation motivates the introduction of a fixed-point method given by $x^{k+1} = x^k - \sigma_k F(x^k)$, where σ_k is defined at every iteration. Methods based on this approach will be called sequential residual methods (SRM) in the present paper.

Popular SRM were inspired by the Barzilai–Borwein or spectral choice for the minimization of functions [3, 55, 56]. Defining

$$s^k = x^{k+1} - x^k \text{ and } y^k = F(x^{k+1}) - F(x^k), \tag{2}$$

algorithms SANE [42] and DF-SANE [41] compute

$$\sigma_{k+1} = \|s^k\|^2/(y^k)^T s^k, \tag{3}$$

safeguarded in such a way that $|\sigma_{k+1}|$ is bounded and away from zero. This formula had been used in the context of self-scaling variable metric methods for minimization [50] as it provides a scale invariant diagonal first approximation of the Hessian. The choice (3) of σ_{k+1} may be justified with the same arguments that Raydan [55] employed for the choice of the Barzilai–Borwein or spectral step in minimization problems. After the computation of x^{k+1}, we consider the (generally unsolvable) problem of satisfying the secant equation [24] $B_{k+1}s^k = y^k$ subject to $B_{k+1} = cI$. This leads to the minimization of $\|cIs^k - y^k\|^2$, whose solution, if $s^k \neq 0$, is $c = (y^k)^T s^k/\|s^k\|^2$. Therefore, a "natural" residual-based iteration for solving problem (1) could be given by $x^{k+1} = x^k - \sigma_k F(x^k)$ with σ_0 arbitrary and σ_{k+1} defined by a safeguarded version of (3) for all $k \geq 0$.

However, unlike the case of unconstrained minimization, in which $F(x^k)$ is a gradient, the direction $d^k = -\sigma_k F(x^k)$ may not be a descent direction for the natural merit function $f(x)$ defined by

$$f(x) = \frac{1}{2}\|F(x)\|^2 \text{ for all } x \in \mathbb{R}^n.$$

In SANE [42], a test is performed in order to verify whether $F(x^k)$ is a descent direction. If this is the case, since $\nabla f(x) = J(x)^T F(x)$, we should have

$$F(x^k)^T J(x^k) F(x^k) < 0.$$

In order to avoid the employment of derivatives, SANE employs the approximation

$$J(x^k)F(x^k) \approx \frac{F(x^k + hF(x^k)) - F(x^k)}{h}$$

for a small $h > 0$. In this way, the descent test is equivalent to

$$F(x^k)^T F(x^k + hF(x^k)) < \|F(x^k)\|^2,$$

which requires an auxiliary functional evaluation per iteration. The necessity of an auxiliary residual evaluation per iteration in SANE motivated the introduction of DF-SANE [41]. Roughly speaking, in DF-SANE, one gets descent by starting with the trial point $x^k - \sigma_k F(x^k)$ and proceeding to a double backtracking scheme along positive and negative directions, aiming for $\|F(x^{k+1})\|$ to be sufficiently smaller than the maximum value of the residual norm in M consecutive past iterations, where M is given.

Algorithm 2.1, introduced in this section, corresponds to an SRM method that generalizes DF-SANE in three aspects. First, it uses residual-related directions, but not necessarily the residual, as in DF-SANE. Second, it admits an initial step in the search direction that is not necessarily the spectral step, which in DF-SANE is just discarded when it does not satisfy given safeguards. Third, and more important from the practical point of view, the method proposed in the present work allows accelerations. The description of Algorithm 2.1 aims to emphasize the aspects that influence theoretical convergence properties. For this reason, acceleration steps appear only as a small detail in the description of the algorithm, although, in practice, they

are essential for the algorithm's robustness and efficiency. The description of the algorithm follows.

Algorithm 2.1. Let $\gamma \in (0,1)$, $0 < \sigma_{\min} < \sigma_{\max} < \infty$, $0 < \tau_{\min} < \tau_{\max} < 1$, $\alpha_{\text{small}} > 0$, $k_{\text{mon}} \in [1, \infty]$, a positive integer M, a sequence $\{\eta_k\}$ such that $\eta_k > 0$ for all $k \in \mathbb{N}$ and

$$\sum_{k=0}^{\infty} \eta_k = \eta < \infty, \tag{4}$$

and $x_0 \in \mathbb{R}^n$ be given. Set $k \leftarrow 0$ and RANDIR $\leftarrow$ FALSE.

Step 1. If $F(x^k) = 0$, then terminate the execution of the algorithm.

Step 2. Choose σ_k such that $|\sigma_k| \in [\sigma_{\min}, \sigma_{\max}]$ and $v^k \in \mathbb{R}^n$ such that $\|v^k\| = \|F(x^k)\|$. If RANDIR, then compute $v^k = \|F(x^k)\| \, v^k_{\text{ran}}$, where v^k_{ran} is a random vector uniformly chosen in the unitary sphere. Compute

$$\bar{f}_k = \begin{cases} \max\{f(x^k), \ldots, f(x^{\max\{0,k-M+1\}})\} & \text{if } k < k_{\text{mon}}, \\ f(x^k) & \text{otherwise.} \end{cases} \tag{5}$$

Step 2.1. Set $\alpha_+ \leftarrow 1$ and $\alpha_- \leftarrow 1$.

Step 2.2. Set $d \leftarrow -\sigma_k v^k$ and $\alpha \leftarrow \alpha_+$. Consider

$$f(x^k + \alpha d) \leq \bar{f}_k + \eta_k - \gamma \alpha^2 f(x^k). \tag{6}$$

If (6) holds, then define $d^k = d$ and $\alpha_k = \alpha$ and go to Step 3.

Step 2.3. Set $d \leftarrow \sigma_k v^k$ and $\alpha \leftarrow \alpha_-$. If (6) holds, then define $d^k = d$ and $\alpha_k = \alpha$ and go to Step 3.

Step 2.4. Choose $\alpha_+^{\text{new}} \in [\tau_{\min}\alpha_+, \tau_{\max}\alpha_+]$ and $\alpha_-^{\text{new}} \in [\tau_{\min}\alpha_-, \tau_{\max}\alpha_-]$, set $\alpha_+ \leftarrow \alpha_+^{\text{new}}$, $\alpha_- \leftarrow \alpha_-^{\text{new}}$, and go to Step 2.2.

Step 3. Compute x^{k+1} such that $f(x^{k+1}) \leq f(x^k + \alpha_k d^k)$, RANDIR $\leftarrow$ $(k+1 \geq k_{\text{mon}}$ and $\alpha_k < \alpha_{\text{small}})$, set $k \leftarrow k+1$, and go to Step 1.

As in DF-SANE, the sufficient decrease in (6) corresponds to a nonmonotone strategy that combines the ones introduced in [35] and [44]. The main differences of Algorithm 2.1 with respect to DF-SANE are the presence of accelerations at Step 3 and the choice of the nonaccelerated step in a residual-related way, but not necessarily in the residual direction. The DF-SANE method presented in [41] is the particular case of Algorithm 2.1 in which $v^k = F(x^k)$ and $x^{k+1} = x^k + \alpha_k d^k$. Taking $v^k = F(x^k)$ is also the default choice in the practical implementation of Algorithm 2.1 described in this paper. However, the generalization of using a direction v^k that satisfies $\|v^k\| = \|F(x^k)\|$ but that is not necessarily the residue opens up the possibility of using other alternatives. Moreover, as will be shown in the next section, taking random residual-related directions an infinite number of times makes the method converge to points at which the gradient of f vanishes. In Algorithm 2.1, a random direction is employed at iteration $k \geq k_{\text{mon}}$ when the step α_{k-1} turned out to be small. A small value of α_{k-1} reveals that the direction chosen at iteration $k-1$ was not very efficient and, so, something radically different (represented by a random direction) should be tried. The property of convergence to stationary points of f does not hold for the original version of DF-SANE which, under the same assumptions on F, possesses convergence to points such that F is orthogonal to $J^T F$, where J is the Jacobian of F; see [41, Thm. 1]. Since being critical of f implies that F is orthogonal to $J^T F$ but the reciprocal is false, the convergence result of Algorithm 2.1 is stronger

than the convergence result of the original version of DF-SANE. The condition of choosing x^{k+1} that satisfies $f(x^{k+1}) \leq f(x^k + \alpha_k d^k)$ evidently allows one to choose $x^{k+1} = x^k + \alpha_k d^k$ as in the case of DF-SANE, but it also opens the possibility of replacing $x^k + \alpha_k d^k$ by something even better, precisely opening the possibility for acceleration. Finally, differently from DF-SANE, we open the possibility of using an almost monotone strategy if k is sufficiently large. With this choice, hypotheses of convergence theory that are plausible in specific cases become always true.

3. Global convergence. In this section we prove global convergence properties of Algorithm 2.1. Our main purpose is to find solutions of $F(x) = 0$ or, at least, points at which the residual norm $\|F(x)\|$ is as small as desired, given an arbitrary tolerance. However, this purpose could be excessively ambitious because, in the worst case, solutions of the system, or even approximate solutions, may not exist. For this reason we analyze the situations in which convergence to a (stationary) point, at which the gradient of the sum of squares vanishes, occurs. The option of taking directions that are not residuals but are residual-related in the sense that their norms coincide with those of the residuals is crucial for this purpose. Roughly speaking, we will prove that, taking random residual-related directions an infinite number of times (but not at all iterations), convergence to stationary points necessarily takes place.

In Lemma 3.1 we prove that, given an arbitrary tolerance $\varepsilon > 0$, Algorithm 2.1 either finds an approximate solution such that $\|F(x^k)\| \leq \varepsilon$ in a finite number of iterations or produces an infinite sequence of steps $\{\alpha_k\}$ that tends to zero. For this purpose we will only use continuity of F. The lemma begins with a simple proof that the iteration is well defined.

LEMMA 3.1. *Assume that F is continuous, $x^k \in \mathbb{R}^n$ is an arbitrary iterate of Algorithm* 2.1, *and $\|F(x^k)\| \neq 0$. Then, α_k and d^k satisfying* (6) *and x^{k+1} satisfying $f(x^{k+1}) \leq f(x^k + \alpha_k d^k)$ are well defined. Moreover, if the algorithm generates an infinite sequence $\{x^k\}$, then either* (i), (ii), *or* (iii) *below holds:*

(i) *there exists an infinite subset of indices $K_1 \subset \mathbb{N}$ such that*

$$\lim_{k \in K_1} \|F(x^k)\| = 0, \tag{7}$$

(ii) *$k_{\text{mon}} = \infty$ and there exists an infinite subset of indices K_2 such that*

$$\lim_{k \in K_2} \alpha_k = 0, \tag{8}$$

(iii) *$k_{\text{mon}} < \infty$ and*

$$\lim_{k \to \infty} \alpha_k = 0. \tag{9}$$

Proof. The fact that α_k, d^k, and x^{k+1} are always well defined follows from the continuity of F using that $\eta_k > 0$ and that α_k can be as small as needed. Assume that the algorithm generates an infinite sequence $\{x^k\}$. Suppose that (7) does not hold. Then, there exists $c_1 > 0$ such that

$$\|F(x^k)\| > c_1 \text{ for all } k \text{ large enough.} \tag{10}$$

Assume first that $k_{\text{mon}} = \infty$. Therefore, if (8) does not hold, then there exists $c_2 > 0$ such that

$$\alpha_k > c_2 \text{ for all } k \text{ large enough.}$$

Then, by (10),

$$\alpha_k^2 f(x^k) > (c_1 c_2)^2/2 \text{ for all } k \text{ large enough.}$$

Then, since x^{k+1} is well defined, for all k large enough,

$$f(x^{k+1}) \leq \bar{f}_k + \eta_k - \gamma \alpha_k^2 f(x^k) \leq \bar{f}_k + \eta_k - \gamma (c_1 c_2)^2/2,$$

where

$$\bar{f}_k = \max\{f(x^k), \ldots, f(x^{\max\{0,k-M+1\}})\}.$$

Since $\eta_k \to 0$, then there exists $k_1 \in \mathbb{N}$ such that for all $k \geq k_1$,

$$f(x^{k+1}) \leq \bar{f}_k - \gamma (c_1 c_2)^2/4$$

and, by induction,

$$f(x^{k+j}) \leq \bar{f}_k - \gamma (c_1 c_2)^2/4$$

for all $j = 1, \ldots, M$. Therefore,

$$\bar{f}_{k+M} \leq \bar{f}_k - \gamma (c_1 c_2)^2/4.$$

Since this inequality holds for all $k \geq k_1$ we conclude that $\bar{f}_{k+\ell M}$ is negative for ℓ large enough, which contradicts the fact that $f(x) \geq 0$ for all $x \in \mathbb{R}^n$.

Consider now the case $k_{\text{mon}} < \infty$ and assume that (9) does not hold. Then, there exists $c_2 > 0$ such that the subset of indices $K_3 = \{k \geq k_{\text{mon}} \mid \alpha_k > c_2\}$ has infinitely many indices. Since $k_{\text{mon}} < \infty$, we have that $\bar{f}_k = f(x^k)$ for all $k \geq k_{\text{mon}}$ and, thus,

$$f(x^{k+1}) \leq f(x^k) + \eta_k - \gamma (c_1 c_2)^2/2$$

for all $k \in K_3$, whereas

$$f(x^{k+1}) \leq f(x^k) + \eta_k$$

for all $k \notin K_3$ and $k \geq k_{\text{mon}}$. Let $\{k_1, k_2, \ldots\}$ be the elements of K_3. Then, for all $j = 1, 2, \ldots,$

$$f(x^{k_{j+1}}) \leq f(x^{k_j}) + \sum_{k=k_j}^{k_{j+1}-1} \eta_k - \gamma (c_1 c_2)^2/2. \tag{11}$$

By (4), $\sum_{k=k_j}^{k_{j+1}-1} \eta_k \leq \gamma (c_1 c_2)^2/4$ for j large enough. Therefore, by (11), $f(x^{k_{j+1}}) \leq f(x^{k_j}) - \gamma (c_1 c_2)^2/4$ for j large enough, i.e., $f(x^{k_{j+1}}) < 0$ for j large enough, which contradicts the fact that $f(x) \geq 0$ for all $x \in \mathbb{R}^n$. This completes the proof. ∎

Algorithm 2.1 could be applied to nonsmooth equation systems in the sense that the only requirement for its well-definedness is the continuity of F. However, its theoretical properties are incomplete in that case since Lemma 3.1 still holds but Lemma 3.2, which tells what happens when a subsequence of $\{\alpha_k\}$ goes to zero, relies on the continuity of the Jacobian of F; thus making the result of its application unpredictable. Note that in the present work the global convergence of the proposed method is proved using the continuity of the Jacobian of F, while classical local convergence results of quasi-Newton methods use the Lipschitz continuity of the Jacobian; see, for example, [24, Thm. 8.2.2, p. 177]. On the other hand, [16, 17] present an extension of quasi-Newton methods to nonsmooth problems based on smoothing techniques. These works would provide a useful path for research concerning the application of the ideas contained in the present work to the nonsmooth case.

LEMMA 3.2. *Assume that $F(x)$ admits continuous derivatives for all $x \in \mathbb{R}^n$ and $\{x^k\}$ is generated by Algorithm* 2.1. *Assume that $\{\|F(x^k)\|\}$ is bounded away from zero and that K_2 is an infinite set of indices such that*

$$\lim_{k \in K_2} \alpha_k = 0. \tag{12}$$

Let $x_* \in \mathbb{R}^n$ be a limit point of $\{x^k\}_{k \in K_2}$. Then, the set of limit points of $\{v^k\}_{k \in K_2}$ is nonempty, and, if v is a limit point of $\{v^k\}_{k \in K_2}$, then we have that

$$\langle J(x_*)v, F(x_*)\rangle = \langle v, J(x_*)^T F(x_*)\rangle = 0. \tag{13}$$

Proof. Assume that $K \subset K_2$ is such that $\lim_{k \in K} x^k = x_*$. Then, by continuity, $\lim_{k \in K} F(x^k) = F(x_*)$. Therefore, the sequence $\{F(x^k)\}_{k \in K}$ is bounded. Since $\|v^k\| = \|F(x^k)\|$ for all k, we have that the sequence $\{v^k\}_{k \in K}$ is bounded too; therefore, it admits a limit point $v \in \mathbb{R}^n$, as we wanted to prove.

Let K_3 be an infinite subset of K and let $v \in \mathbb{R}^n$ be such that $\lim_{k \in K_3} v^k = v$. Since the first trial value for α_k at each iteration is 1, (12) implies that there exist K_4, an infinite subset of K_3, $\alpha_{k,+} > 0$, $\alpha_{k,-} > 0$, $\sigma_k \in [\sigma_{\min}, \sigma_{\max}]$, and $d^k = -\sigma_k v^k$ such that

$$\lim_{k \in K_4} \sigma_k = \sigma \in [\sigma_{\min}, \sigma_{\max}],$$

$$\lim_{k \in K_4} \alpha_{k,+} = \lim_{k \in K_4} \alpha_{k,-} = 0,$$

and, for all $k \in K_4$,

$$f(x^k + \alpha_{k,+} d^k) > \bar{f}_k + \eta_k - \gamma \alpha_{k,+}^2 f(x^k)$$

and

$$f(x^k - \alpha_{k,-} d^k) > \bar{f}_k + \eta_k - \gamma \alpha_{k,-}^2 f(x^k).$$

Therefore, by the definition (5) of $\bar{f}_k$, for all $k \in K_4$,

$$f(x^k + \alpha_{k,+} d^k) > f(x^k) + \eta_k - \gamma \alpha_{k,+}^2 f(x^k)$$

and

$$f(x^k - \alpha_{k,-} d^k) > f(x^k) + \eta_k - \gamma \alpha_{k,-}^2 f(x^k).$$

So, since $\eta_k > 0$,

$$\frac{f(x^k + \alpha_{k,+} d^k) - f(x^k)}{\alpha_{k,+}} > -\gamma \alpha_{k,+} f(x^k)$$

and

$$\frac{f(x^k - \alpha_{k,-} d^k) - f(x^k)}{\alpha_{k,-}} > -\gamma \alpha_{k,-} f(x^k)$$

for all $k \in K_4$. Thus, by the mean value theorem, there exist $\xi_{k,+} \in [0, \alpha_{k,+}]$ and $\xi_{k,-} \in [0, \alpha_{k,-}]$ such that

$$\langle \nabla f(x^k + \xi_{k,+} d^k), d^k \rangle > -\gamma \alpha_{k,+} f(x^k) \tag{14}$$

and

$$-\langle \nabla f(x^k - \xi_{k,-} d^k), d^k \rangle > -\gamma \alpha_{k,-} f(x^k) \tag{15}$$

for all $k \in K_4$. Taking limits for $k \in K_4$ in both sides of (14) and (15) we get

$$\langle \nabla f(x_*), \sigma v \rangle = 0.$$

Therefore, (13) is proved. ∎

Theorem 3.1 will state a sufficient condition for the annihilation of the gradient of $f(x)$ at a limit point of the sequence. For that purpose, we will assume that the absolute value of the cosine of the angle determined by v^k and $J(x^k)^T F(x^k)$ is bigger than a tolerance $\omega > 0$ infinitely many times. Note that we do not require this condition to hold for every $k \in \mathbb{N}$. The distinction between these two alternatives is not negligible. For example, if v^k is chosen infinitely many times as a random vector the required angle condition holds with probability 1. Conversely, if we require the fulfillment of the angle condition for all $k \in \mathbb{N}$ and we choose random directions, the probability of fulfillment would be zero.

THEOREM 3.1. *Assume that $F(x)$ admits continuous derivatives for all $x \in \mathbb{R}^n$ and $\{x^k\}$ is generated by Algorithm* 2.1. *Suppose that the level set $\{x \in \mathbb{R}^n \mid f(x) \leq f(x^0) + \eta\}$ is bounded. Assume that at least one of the following possibilities holds:*

(i) *Algorithm* 2.1 *stops at Step* 1 *with $\|F(x^k)\| = 0$ for some $k \in \mathbb{N}$.*
(ii) *There exists a sequence of indices $K_1 \subset \mathbb{N}$ such that $\lim_{k \in K_1} \|F(x^k)\| = 0$.*
(iii) *There exist $\omega \in [\frac{1}{2}, 1)$ and a sequence of indices K_2 such that*

$$\lim_{k \in K_2} \alpha_k = 0$$

and

$$|\langle v^k, J(x^k)^T F(x^k) \rangle| \geq \omega \|v^k\| \|J(x^k)^T F(x^k)\| \tag{16}$$

for all $k \in K_2$.

Then, for any given $\varepsilon > 0$, there exists $k \in \mathbb{N}$ such that $\|J(x^k)^T F(x^k)\| \leq \varepsilon$. Moreover, if the algorithm does not stop at Step 1 *with $\|F(x^k)\| = 0$, there exists a limit point x_* of the sequence generated by the algorithm such that $\|J(x_*)^T F(x_*)\| = 0$.*

Proof. If the sequence generated by the algorithm stops at some k such that $\|F(x^k)\| = 0$, then the thesis obviously holds.

By the boundedness of the level set defined by $f(x^0)+\eta$, the sequence $\{x^k\}$ is bounded. Therefore, by continuity, $\{\|F(x^k)\|\}$ and $\{\|J(x^k)\|\}$ are also bounded and $\lim_{k\in K_1}\|F(x^k)\|=0$ implies that $\lim_{k\in K_1}\|J(x^k)^TF(x^k)\|=0$. So, the thesis also holds when the second alternative in the hypothesis takes place.

Therefore, we only need to consider the case in which $\|F(x^k)\|$ is bounded away from zero and the third alternative in the hypothesis holds. By the definition of v^k and the boundedness of $\|F(x^k)\|$ we have that $\{\|v^k\|\}$ is also bounded. Therefore, there exist an infinite set $K_3\subset K_2$, $x_*\in\mathbb{R}^n$, and $v\in\mathbb{R}^n$ such that $\lim_{k\in K_3}x^k=x_*$ and $\lim_{k\in K_3}v^k=v$. Therefore, by Lemma 3.2,

$$\langle v, J(x_*)^TF(x_*)\rangle=0.$$

So, by the convergence of $\{v^k\}$ and $\{x^k\}$ for $k\in K_3$ and the continuity of F and J,

$$\lim_{k\in K_3}\langle v^k, J(x^k)^TF(x^k)\rangle=0.$$

Therefore, by (16), since $K_3\subset K_2$,

$$\lim_{k\in K_3}\|v^k\|\|J(x^k)^TF(x^k)\|=0. \tag{17}$$

Since, in this case, $\|F(x^k)\|$ is bounded away from zero, we have that $\|v^k\|$ is bounded away from zero too. Therefore, by (17),

$$\lim_{k\in K_3}\|J(x^k)^TF(x^k)\|=0.$$

Therefore, the thesis is proved. ∎

Remark. The hypotheses of Theorem 3.1 are not plausible for the classical DF-SANE algorithm because, in general, it is impossible to guarantee the angle condition (16) if we only use residual directions. In that case, convergence to points such that $F(x^*)$ is orthogonal to $J(x^*)^TF(x^*)$ but $\|J(x^*)^TF(x^*)\|\neq 0$ is possible. The generalization introduced in this paper for the choice of v^k when $k_{\text{mon}}<\infty$ provides a remedy for that drawback. If Algorithm 2.1 stops at Step 1 with $\|F(x^k)\|=0$ for some $k\in\mathbb{N}$ (item (i) in the hypothesis) or if there exists a sequence of indices $K_1\subset\mathbb{N}$ such that $\lim_{k\in K_1}\|F(x^k)\|=0$ (item (ii) in the hypothesis), then there is nothing to prove. So, we assume that items (i) and (ii) do not hold and, in what follows, we show that item (iii) follows. If items (i) and (ii) do not hold, we have that $\{\|F(x^k\|\}$ is bounded away from zero. In this case, by Lemma 3.1, we have that $\lim_{k\to\infty}\alpha_k=0$. Then $\alpha_k<\alpha_{\text{small}}$ for all k large enough and, therefore, by the definition of Algorithm 2.1, $v^k=\|F(x^k)\|v^k_{\text{ran}}$ for all k sufficiently large. Let $K=\{k\in\mathbb{N}\mid k\geq k_{\text{mon}}$ and $\alpha_{k-1}<\alpha_{\text{small}}\}$. Clearly, if $k\in K$ and $\|J(x^k)^TF(x^k)\|=0$, then inequality (16) holds for all $\omega\in[\frac{1}{2},1)$. Let $k\in K$ be such that $\|J(x^k)^TF(x^k)\|\neq 0$. In this case, (16) is equivalent to

$$\begin{aligned}&\text{angle}\left(v^k_{\text{ran}},\frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}\right)\leq\arccos(\omega)\ \text{ or}\\ &\text{angle}\left(v^k_{\text{ran}},-\frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}\right)\leq\arccos(\omega).\end{aligned} \tag{18}$$

The curvilinear distance between $v^k_{\rm ran}$ and $\frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}$ taken along the geodesic that passes through these points in the unitary sphere is equal to the angle between $v^k_{\rm ran}$ and $\frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}$. Therefore, (18) is equivalent to

$$\text{discur}\left(v^k_{\rm ran}, \frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}\right) \le \arccos(\omega) \quad \text{or} \tag{19}$$
$$\text{discur}\left(v^k_{\rm ran}, -\frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}\right) \le \arccos(\omega).$$

The vector $v^k_{\rm ran}$ is chosen randomly with uniform distribution in the unitary sphere, independently of $\frac{J(x^k)^TF(x^k)}{\|J(x^k)^TF(x^k)\|}$. Therefore, for a given $\omega \in [\frac{1}{2}, 1)$, by (19), the probability that (16) holds is not smaller than $2\arccos(\omega)/\pi > 0$. Let $K_2(\omega) \subset K$ be the set of indices k for which (16) holds. Then, by the Borel–Cantelli lemma [15, 27], $K_2(\omega)$ has infinitely many indices with probability 1.

COROLLARY 3.1. *Suppose that the assumptions of Theorem* 3.1 *hold. Assume, moreover, that there exists* $\gamma > 0$ *such that for all* $k \in \mathbb{N}$,

$$\|J(x^k)^TF(x^k)\| \ge \gamma\|F(x^k)\|. \tag{20}$$

Then, given $\varepsilon > 0$, *there exists an iterate* k *such that* $\|F(x^k)\| \le \varepsilon$. *Moreover, there exists a limit point* x_* *of the sequence generated by the algorithm such that* $\|F(x_*)\| = 0$.

Proof. The thesis of this corollary follows directly from (20) and Theorem 3.1. ∎

COROLLARY 3.2. *Suppose that the assumptions of Theorem* 3.1 *hold and* $J(x)$ *is nonsingular with* $\|J(x)^{-1}\|$ *uniformly bounded for all* $x \in \mathbb{R}^n$. *Then, for all* $\ell = 1, 2, \ldots$, *there exists an iterate* $k(\ell)$ *such that* $\|F(x^{k(\ell)})\| \le 1/\ell$. *Moreover, at every limit point* x_* *of the sequence* $\{x^{k(\ell)}\}$ *we have that* $\|F(x_*)\| = 0$.

The lemma below shows that if $\|F(x^k)\| \to 0$ for a subsequence, then it goes to zero for the whole sequence. We will need the following Assumptions A and B. Assumption A guarantees that the difference between consecutive iterates is not greater than a multiple of the residual norm. Note that such an assumption holds trivially with $c = \sigma_{\max}$ when the iteration is not accelerated and, so, $x^{k+1} = x^k + \alpha_k d^k$. Therefore, Assumption A needs to be stated only with respect to accelerated steps and in practice can be granted by simply discarding the accelerated iterates that do not satisfy it. Assumption B merely states that F is uniformly continuous at least restricted to the set of iterates. Assumption B holds if F is uniformly continuous onto a level set of f that contains the whole sequence generated by the algorithm. A sufficient condition for this fact is the fulfillment of a Lipschitz condition by the function F.

Assumption A. There exists $c > 0$ such that, for all $k \in \mathbb{N}$, whenever $x^{k+1} \ne x^k + \alpha_k d^k$, we have that

$$\|x^{k+1} - x^k\| \le c\|F(x^k)\|.$$

Assumption B. For all $\varepsilon > 0$, there exists δ such that whenever $\|x^{k+1} - x^k\| \le \delta$ one has that $\|F(x^{k+1}) - F(x^k)\| \le \varepsilon$.

Lemma 3.3 below is stronger than Theorem 2 of [41] because here we do not assume the existence of a limit point.

LEMMA 3.3. *Suppose that F is continuous, Algorithm* 2.1 *generates the infinite sequence $\{x^k\}$, Assumptions* A *and* B *hold, and there exists an infinite subsequence defined by the indices in K_1 such that*

$$\lim_{k\in K_1} \|F(x^k)\| = 0. \tag{21}$$

Then,

$$\lim_{k\to\infty} \|F(x^k)\| = 0 \tag{22}$$

and

$$\lim_{k\to\infty} \|x^{k+1} - x^k\| = 0. \tag{23}$$

Proof. Assume that (22) is not true. Then, there exists an infinite subsequence $\{x^k\}_{k\in K_2}$ such that

$$f(x^k) > \hat{c} > 0 \tag{24}$$

for all $k \in K_2$.

Let us prove first that

$$\lim_{k\in K_1} \|F(x^{k+1})\| = 0. \tag{25}$$

By the definition of Algorithm 2.1, when $x^{k+1} = x^k + \alpha_k d^k$, $\|x^{k+1} - x^k\| \le \sigma_{\max}\|F(x^k)\|$. This, along with Assumption A and (21), implies that

$$\lim_{k\in K_1} \|x^{k+1} - x^k\| = 0. \tag{26}$$

By Assumption B, (26) implies

$$\lim_{k\in K_1} \|F(x^{k+1}) - F(x^k)\| = 0, \tag{27}$$

and (25) follows from (21) and (27).

Consider first the case in which $k_{\mathrm{mon}} = \infty$. By induction, we deduce that, for all $j = 1, \ldots, M$,

$$\lim_{k\in K_1} \|F(x^{k+j})\| = 0.$$

Therefore,

$$\lim_{k\in K_1} \max\{f(x^k), \ldots, f(x^{k+M-1})\} = 0. \tag{28}$$

But

$$\begin{aligned} f(x^{k+M}) &\le \max\{f(x^k), \ldots, f(x^{k+M-1})\} + \eta_{k+M-1} - \gamma\alpha_{k+M-1}^2 f(x^{k+M-1}) \\ &\le \max\{f(x^k), \ldots, f(x^{k+M-1})\} + \eta_{k+M-1}. \end{aligned}$$

Analogously,

$$\begin{aligned} f(x^{k+M+1}) &\leq \max\{f(x^{k+1}),\ldots,f(x^{k+M})\} + \eta_{k+M} - \gamma\alpha_{k+M}^2 f(x^{k+M}) \\ &\leq \max\{f(x^{k+1}),\ldots,f(x^{k+M})\} + \eta_{k+M} \\ &\leq \max\{f(x^{k}),\ldots,f(x^{k+M-1})\} + \eta_{k+M-1} + \eta_{k+M}, \end{aligned}$$

and, inductively,

$$f(x^{k+M+j}) \leq \max\{f(x^{k}),\ldots,f(x^{k+M-1})\} + \eta_{k+M-1} + \eta_{k+M} + \ldots + \eta_{k+M+j-1}$$

for all $j = 0, 1, \ldots, M-1$. Therefore,

$$\max\{f(x^{k+M}),\ldots,f(x^{k+2M-1})\} \leq \max\{f(x^{k}),\ldots,f(x^{k+M-1})\} + \sum_{j=k+M-1}^{k+2M-2} \eta_j.$$

By induction on $\ell = 1, 2, \ldots$, we obtain that

$$\max\{f(x^{k+\ell M}),\ldots,f(x^{k+(\ell+1)M-1})\} \leq \max\{f(x^{k}),\ldots,f(x^{k+M-1})\} + \sum_{j=k+M-1}^{k+(\ell+1)M-2} \eta_j.$$

Therefore, for all $\ell = 1, 2, \ldots$, we have that

(29)

$$\max\{f(x^{k+\ell M}),\ldots,f(x^{k+(\ell+1)M-1})\} \leq \max\{f(x^{k}),\ldots,f(x^{k+M-1})\} + \sum_{j=k+M-1}^{\infty} \eta_j.$$

By (28), the summability of η_j, and (29), there exists $k_1 \in K_1$ such that for all $k \in K_1$ such that $k \geq k_1$,

$$\max\{f(x^{k}),\ldots,f(x^{k+M-1})\} + \sum_{j=k+M-1}^{\infty} \eta_j \leq \hat{c}/2.$$

Therefore, by (29), for all $k \in K_1$ such that $k \geq k_1$ and all $\ell = 1, 2, \ldots$,

$$\max\{f(x^{k+\ell M}),\ldots,f(x^{k+(\ell+1)M-1})\} \leq \hat{c}/2.$$

Clearly, this is incompatible with the existence of a subsequence such that (24) holds. Therefore, (22) is proved.

In the case $k_{\mathrm{mon}} < \infty$, for $k_1 \in K_1$ large enough and $k \geq k_1$,

$$f(x^k) \leq f(x^{k_1}) + \sum_{j=k_1}^{k-1} \eta_j \leq f(x^{k_1}) + \sum_{j=k_1}^{\infty} \eta_j.$$

Then, given $\varepsilon > 0$, there exists $k_1 \in K_1$ such that for all $k \geq k_1$,

$$f(x^k) \leq f(x^{k_1}) + \varepsilon/2.$$

But if $k_1 \in K_1$ is large enough we have that $f(x^{k^1}) \leq \varepsilon/2$. Then, $\lim_{k\to\infty} f(x^k) = 0$. Therefore, (22) is proved.

In both cases $k_{\text{mon}} = \infty$ and $k_{\text{mon}} < \infty$, by Assumption A and (22), (23) also holds. ∎

THEOREM 3.2. *Suppose that the assumptions of Theorem* 3.1 *and Lemma* 3.3 *hold, $J(x)$ is nonsingular, and $\|J(x)^{-1}\|$ is uniformly bounded for all $x \in \mathbb{R}^n$. Then, there exists $x_* \in \mathbb{R}^n$ such that $F(x_*) = 0$ and $\lim_{k\to\infty} x^k = x_*$.*

Proof. By Corollary 3.2, there exists a subsequence $\{x^k\}_{k\in K_1}$ that converges to a point x_* such that $F(x_*) = 0$. Then, by Lemma 3.3,

$$\lim_{k\to\infty} F(x^k) = 0 \tag{30}$$

and

$$\lim_{k\to\infty} \|x^{k+1} - x^k\| = 0.$$

Since $J(x_*)$ is nonsingular, by the inverse function theorem, there exists $\delta > 0$ such that $\|F(x)\| > 0$ whenever $0 < \|x - x_*\| \leq \delta$. Therefore, given $\varepsilon \in (0, \delta]$ arbitrary, there exists $\hat{c} > 0$ such that

$$\|F(x)\| \geq \hat{c} \text{ whenever } \varepsilon \leq \|x - x_*\| \leq \delta.$$

By (30), the number of iterates in the set defined by $\varepsilon \leq \|x - x_*\| \leq \delta$ is finite. On the other hand, if k is large enough, one has that $\|x^{k+1} - x^k\| \leq \delta - \varepsilon$. Since there exists a subsequence of $\{x^k\}$ that tends to x_* it turns out that $\|x^k - x_*\| \leq \varepsilon$ if k is large enough. Since ε was arbitrary this implies that $\lim_{k\to\infty} x^k = x_*$ as we wanted to prove. ∎

We finish this section with a theorem that states that, under some assumptions, if in Algorithm 2.1 we have $x^{k+1} = x^k + \alpha_k d^k$ for all k, i.e., without accelerations, then $\{x^k\}$ converges superlinearly to a solution. A similar result was proved in [33, Thm. 3.1] with respect to a spectral gradient methods with retards for quadratic unconstrained minimization. We will need two assumptions. In Assumption C, the sequence generated by Algorithm 2.1 is assumed to be generated by the spectral residual choice with the first step accepted at each iteration and without acceleration. In Assumption D, the Jacobian $J(x)$ is assumed to be Lipschitz-continuous. The first part of Assumption C ($v^k = F(x^k)$) limits the consequences of Theorem 3.3 to the particular case in which Algorithm 2.1 coincides with the DF-SANE method. The fact that σ_k is well defined in the form stated in this assumption and the fact that the descent condition (6) holds with $\alpha = 1$ are in fact assumptions on the behavior of the algorithm, with the same status as the hypotheses (31) and (32) of Theorem 3.3. We do not make any claim about the frequency with which these hypotheses hold.

Assumption C. Assume that, at Step 2 of Algorithm 2.1, we choose

$$v^k = F(x^k)$$

and that there exists $k_0 \geq 1$ such that for all $k \geq k_0$,

$$(s^{k-1})^T y^{k-1} \neq 0$$

and

$$\sigma_k = \frac{(s^{k-1})^T s^{k-1}}{(s^{k-1})^T y^{k-1}},$$

where s^k and y^k are defined by (2). Assume, moreover, that, for all $k \geq k_0$, (6) holds with $\alpha = 1$ and, at Step 3, $x^{k+1} = x^k + \alpha_k d^k$.

Assumption C merely states that the spectral step $(s^{k-1})^T s^{k-1}/((s^{k-1})^T y^{k-1})$ is well defined and that σ_k coincides with it for all k sufficiently large. By the mean value theorem $y^{k-1} = [\int_0^1 J(x^{k-1} + ts^{k-1})dt]s^{k-1}$. Therefore,

$$(s^{k-1})^T y^{k-1} = \int_0^1 [(s^{k-1})^T J(x^{k-1} + ts^{k-1})s^{k-1}]dt.$$

Suppose that for all $x \in \mathbb{R}^n$ and $v \neq 0$ we have that $v^T J(x)v/v^T v$ is bounded and bounded away from zero. This implies that $(s^{k-1})^T y^{k-1}/(s^{k-1})^T s^{k-1}$ is bounded and bounded away from zero whenever $s^{k-1} \neq 0$. Analogously, if $-v^T J(x)v/v^T v$ is bounded and bounded away from zero we have that $(s^{k-1})^T y^{k-1}/(s^{k-1})^T s^{k-1}$ is bounded and bounded away from zero. Since the sequence does not terminate at Step 1 we have that $s^{k-1} \neq 0$ for all k. Therefore we see that a sufficient condition for the existence of $\sigma_{\min}$ and $\sigma_{\max}$ (parameters of Algorithm 2.1) such that $|\sigma_k| \in [\sigma_{\min}, \sigma_{\max}]$ holds with $\sigma_k = (s^{k-1})^T s^{k-1}/((s^{k-1})^T y^{k-1})$ is the boundedness and (positive or negative) definiteness of the Jacobian $J(x)$. In case $\sigma_{\min}$ and $\sigma_{\max}$ do not exist or, as parameters of Algorithm 2.1, they are set with wrong values that imply the truncation of σ_k for some k, the thesis of Theorem 3.3 may not hold.

Assumption D. There exists $L > 0$ such that for all x, z in an open and convex set that contains the whole sequence $\{x^k\}$ generated by Algorithm 2.1, the Jacobian is continuous and satisfies

$$\|J(z) - J(x)\| \leq L\|z - x\|.$$

THEOREM 3.3. *Assume that the sequence $\{x^k\}$, generated by Algorithm* 2.1, *does not terminate at Step* 1 *and that Assumptions* C *and* D *hold. Suppose that $x_* \in \mathbb{R}^n$ is such that*

$$\lim_{k\to\infty} x^k = x_*, \tag{31}$$

the Jacobian $J(x_*)$ is nonsingular, and $s \in \mathbb{R}^n$ is such that

$$\lim_{k\to\infty} \frac{s^k}{\|s^k\|} = s. \tag{32}$$

Then,

$$\lim_{k\to\infty} \frac{(s^k)^T y^k}{(s^k)^T s^k} = s^T J(x_*)s, \tag{33}$$

$$F(x_*) = 0,$$

and

$$x^k \text{ converges Q-superlinearly to } x_*.$$

Proof. By Assumption D, there exists an open and convex set that contains the whole sequence $\{x^k\}$ such that for all x, z in that set,

$$F(z) = F(x) + J(x)(z - x) + O(\|z - x\|^2). \tag{34}$$

By Assumption C, the sequence $\{x^k\}$ is such that for all $k \geq k_0$,

$$x^{k+1} = x^k - \frac{1}{\kappa_k} F(x^k), \tag{35}$$

where

$$\kappa_{k+1} = \frac{(s^k)^T y^k}{(s^k)^T s^k}.$$

By (34),

$$y^k = J(x^k)s^k + O(\|s^k\|^2) \tag{36}$$

for all $k \geq k_0$. Therefore, by (36),

$$\kappa_{k+1} = \frac{(s^k)^T [J(x^k)s^k + O(\|s^k\|^2)]}{(s^k)^T s^k}. \tag{37}$$

Thus, by (31), (32), and the continuity of $J(x)$,

$$\lim_{k \to \infty} \kappa_{k+1} = s^T J(x_*) s.$$

Therefore, (33) is proved.

By (34) we have that

$$F(x^{k+1}) = F(x^k) + J(x^k)s^k + O(\|s^k\|^2).$$

Then, by (35),

$$-\kappa_{k+1} s^{k+1} = -\kappa_k s^k + J(x^k)s^k + O(\|s^k\|^2)$$

for all $k \in \mathbb{N}$. Therefore, for all $k \in \mathbb{N}$,

$$s^{k+1} = \frac{\kappa_k s^k - J(x^k)s^k}{\kappa_{k+1}} - \frac{O(\|s^k\|^2)}{\kappa_{k+1}}.$$

Therefore,

$$s^{k+1} = -\frac{1}{\kappa_{k+1}} [(J(x^k) - \kappa_k I)s^k + O(\|s^k\|^2)].$$

Then

$$\frac{s^{k+1}}{\|s^{k+1}\|} = \frac{-[(J(x^k) - \kappa_k I)s^k + O(\|s^k\|^2)]}{\| - [(J(x^k) - \kappa_k I)s^k + O(\|s^k\|^2)]\|}.$$

So,

$$\frac{s^{k+1}}{\|s^{k+1}\|} = \frac{-[(J(x^k) - \kappa_k I)s^k/\|s^k\| + O(\|s^k\|^2)/\|s^k\|]}{\| - [(J(x^k) - \kappa_k I)s^k/\|s^k\| + O(\|s^k\|^2)/\|s^k\|]\|}. \tag{38}$$

Define

$$\kappa_* = s^T J(x_*)s. \tag{39}$$

Assume that

$$\|(J(x_*) - \kappa_* I)s\| \neq 0. \tag{40}$$

Then, by (40), (33), and (32), taking limits in both sides of (38),

$$s = \frac{-[(J(x_*) - \kappa_* I)s]}{\| - [(J(x_*) - \kappa_* I)s\|}. \tag{41}$$

Premultiplying both sides of (41) by s^T, as $s^T s = 1$, we obtain

$$1 = \frac{-[(s^T J(x_*)s - \kappa_*]}{\| - [(J(x_*) - \kappa_* I)s\|}.$$

Then, by (39), we get a contradiction. This implies that the assumption (40) is false. So,

$$\|(J(x_*) - \kappa_* I)s\| = 0.$$

Therefore,

$$\lim_{k\to\infty} \frac{\kappa_k I s^k - J(x_*)s^k}{\|s^k\|} = 0. \tag{42}$$

But (42) is the Dennis–Moré condition [23] related with the iteration $x^{k+1} = x^k - B_k^{-1}F(x^k)$ with $B_k = \kappa_k I$. Then, we have that $F(x_*) = 0$ and the convergence is superlinear. ∎

4. Acceleration. The sequential secant method (SSM) [2, 34, 38, 46, 64], called the secant method of $n + 1$ points in the classical book of Ortega and Rheinboldt [51], is a classical procedure for solving nonlinear systems of equations. Given $n + 1$ consecutive iterates $x^k, x^{k-1}, \ldots, x^{k-n}$, the SSM implicitly considers an interpolatory affine function $A : \mathbb{R}^n \to \mathbb{R}^n$ such that $A(x^j) = F(x^j)$ for $j = k, k-1, \ldots, k-n$ and defines x^{k+1} as a solution of $A(x) = 0$. If the points $x^k, x^{k-1}, \ldots, x^{k-n}$ are affinely independent there is only one affine function with the interpolatory property, which turns out to be defined by

$$A(x) = F(x^k) + (y^{k-1}, \ldots, y^{k-n})(s^{k-1}, \ldots, s^{k-n})^{-1}(x - x^k),$$

where

$$s^j = x^{j+1} - x^j \text{ and } y^j = F(x^{j+1}) - F(x^j) \text{ for all } j \leq k-1.$$

Moreover, if $F(x^k), F(x^{k-1}), \ldots, F(x^{k-n})$ are also affinely independent we have that $y^{k-1}, \ldots, y^{k-n}$ are linearly independent and $A(x) = 0$ has a unique solution given by

$$x^{k+1} = x^k - (s^{k-1}, \ldots, s^{k-n})(y^{k-1}, \ldots, y^{k-n})^{-1}F(x^k).$$

Practical implementations of SSM try to maintain nonsingularity of the matrices $(s^{k-1}, \ldots, s^{k-n})$ and $(y^{k-1}, \ldots, y^{k-n})$ employing auxiliary points in the case that linear independence fails. See [46] and references therein.

If $s^{k-1}, \ldots, s^{k-n}$ are linearly independent but $y^{k-1}, \ldots, y^{k-n}$ are not, a solution of $A(x) = 0$ may not exist. In this case we may define x^{k+1} as a solution of

$$\text{Minimize } \|F(x^k) + (y^{k-1}, \ldots, y^{k-n})(s^{k-1}, \ldots, s^{k-n})^{-1}(x - x^k)\|^2.$$

A sensible choice for solving this problem is

$$x^{k+1} = x^k - (s^{k-1}, \ldots, s^{k-n})(y^{k-1}, \ldots, y^{k-n})^{\dagger}F(x^k),$$

where $Y^{\dagger}$ denotes the Moore–Penrose pseudoinverse of a matrix Y.

This formula suggests a generalization of the SSM. Instead of using x^k and the n previous iterates to define x^{k+1} we may employ $p < n$ previous iterates. This leads to the iteration

$$x^{k+1} = x^k - (s^{k-1}, \ldots, s^{k-p})(y^{k-1}, \ldots, y^{k-p})^{\dagger}F(x^k). \tag{43}$$

Formula (43) admits a nice geometrical interpretation. Let F_* be the vector of minimal norm in the affine subspace spanned by $F(x^k), \ldots, F(x^{k-p})$. Then,

$$F_* = F(x^k) + \lambda_{k-1}(F(x^k) - F(x^{k-1})) + \cdots + \lambda_{k-p}(F(x^{k-p+1}) - F(x^{k-p})). \tag{44}$$

The idea in (43) consists of using the same coefficients $\lambda_{k-1}, \ldots, \lambda_{k-p}$ in order to compute

$$x^{k+1} = x^k + \lambda_{k-1}(x^k - x^{k-1}) + \cdots + \lambda_{k-p}(x^{k-p+1} - x^{k-p}).$$

In other words, we apply to $x^k, x^{k-1}, \ldots, x^{k-p}$ the same transformation that is applied to $F(x^k), F(x^{k-1}), \ldots, F(x^{k-p})$ for obtaining the minimum norm element in the affine subspace generated by these vectors. The only ambiguity in the representation above is that the coefficients $\lambda_{k-1}, \ldots, \lambda_{k-p}$ are not unique if $F(x^k), F(x^{k-1}), \ldots, F(x^{k-p})$ are not affinely independent. This ambiguity can be removed by taking $\lambda_{k-1}, \ldots, \lambda_{k-p}$ that minimize $\sum_{j=1}^{p} \lambda_{k-j}^2$ subject to (44). This decision leads to the use of the Moore–Penrose pseudoinverse in (43).

However, although x^{k+1} may represent an improvement with respect to x^k, employing $p < n$ previous iterates is not a satisfactory strategy for solving nonlinear systems because it implies that x^{k+1} lies in the $(p+1)$-dimensional affine subspace

generated by $x^k, x^{k-1}, \ldots, x^{k-p}$. Therefore, if we repeat the procedure (43), we obtain that all the iterates $x^{k+1}, x^{k+2}, \ldots$ lie in the same affine subspace of dimension $p+1$ related to $x^k, x^{k-1}, \ldots, x^{k-p}$. Of course, this is a bad strategy if we want to solve a nonlinear system whose solution is not in that affine subspace.

In spite of this drawback a variation of the above formula can be used to accelerate a pre-existent iterative process that produced the iterates $x^{k-p}, \ldots, x^k$. The idea is the following. Assume that $x^k, \ldots, x^{k-p}$ are "previous iterates" of a given method for solving nonlinear systems. Then, we compute, with the given method, x^{k+1}_{trial} as a tentative candidate for being the next iterate x^{k+1}. Using this trial point we compute a possible acceleration by means of

$$x^{k+1}_{\text{accel}} = x^k - (s^k_{\text{trial}}, s^{k-1}, \ldots, s^{k-p})(y^k_{\text{trial}}, y^{k-1}, \ldots, y^{k-p})^\dagger F(x^k),$$

where

$$s^j_{\text{trial}} = x^{j+1}_{\text{trial}} - x^j \text{ and } y^j_{\text{trial}} = F(x^{j+1}_{\text{trial}}) - F(x^j) \text{ for all } j \leq k. \tag{45}$$

Then, if $\|F(x^{k+1}_{\text{accel}})\| < \|F(x^{k+1}_{\text{trial}})\|$, we define $x^{k+1} = x^{k+1}_{\text{accel}}$; otherwise we define $x^{k+1} = x^{k+1}_{\text{trial}}$. In this way, even in the case that x^{k+1} is x^{k+1}_{accel}, x^{k+1} does not need to lie in the affine subspace generated by $x^k, \ldots, x^{k-p}$. Therefore, the curse of the persistent subspace is overcome.

Remark 1. The recurrence

$$x^{k+1} = x^k - (s^{k-1}_{\text{trial}}, \ldots, s^{k-p}_{\text{trial}})(y^{k-1}_{\text{trial}}, \ldots, y^{k-p}_{\text{trial}})^\dagger F(x^k),$$

with $x^{j+1}_{\text{trial}} = x^j - F(x^j)$ for all $j \leq k$ and s^j_{trial} and y^j_{trial} defined by (45), defines an Anderson-type acceleration method for solving nonlinear systems [1]. Many variations of this method have been introduced in recent years with theoretical justifications about the reasons why the Anderson scheme many times accelerates the fixed-point iteration $x^{k+1} = x^k - F(x^k)$. In addition to the references cited in the introduction of this paper we can mention the contributions given in [19, 28, 52, 57, 60, 61, 65]. With the exception of [65], these articles assume that the basic fixed-point iteration is convergent. In the case of [60], only contractivity is assumed. In [19], Anderson acceleration with local and superlinear convergence is introduced, allowing variation of the "depth" p along different iterations and discarding stored iterates according to their relevance. In [28], it is proved that Anderson acceleration improves the convergence rate of contractive fixed-point iterations as a function of the improvement at each step. In [52], the acceleration is used to improve the alternate direction multiplier method. In [57], Anderson acceleration is interpreted as a quasi-Newton method with GMRES-like solution of the linear systems. In [60], Anderson acceleration is used to improve the performance of arbitrary minimization methods. In [61], it is proved that Anderson's method is locally r-linearly convergent if the fixed-point map is a contraction and the coefficients in the linear combination that define the new iteration remain bounded. In [65], Anderson acceleration is applied to the solution of general nonsmooth fixed-point problems. Using safeguarding steps, regularization, and restarts for checking linear independence of the increments, global convergence is proved using only nonexpansiveness of the fixed-point iteration.

Remark 2. The Anderson mixing scheme is presented as an independent method for solving nonlinear systems in [30, sect. 2.5]. In this method one defines

$$\bar{x}^k = x^k - (s^{k-1}, \ldots, s^{k-p})(y^{k-1}, \ldots, y^{k-p})^\dagger F(x^k), \tag{46}$$
$$\bar{F}_k = F(x^k) - (y^{k-1}, \ldots, y^{k-p})(y^{k-1}, \ldots, y^{k-p})^\dagger F(x^k),$$

and

$$x^{k+1} = \bar{x}^k - \beta_k \bar{F}_k, \tag{47}$$

where $\beta_k \equiv \beta$ for all k and β is a problem-dependent parameter in the experiments of [30]. So, $\bar{F}_k$ is the minimum norm element in the affine subspace spanned by $F(x^k), \ldots F(x^{k-p})$. If $p = n$ and this subspace has dimension $n + 1$ we have that $\bar{F}_k = 0$ and the scheme coincides with the SSM.

Algorithm 4.1 formally describes the way in which, at iteration k of Algorithm 2.1 (Step 3), using an acceleration scheme, we compute x^{k+1}.

Algorithm 4.1. Let the integer number $p \geq 1$ be given.

Step 1. Define $x^{k+1}_{\text{trial}} = x^k + \alpha_k d^k$.

Step 2. Define $\underline{k} = \max\{0, k - p + 1\}$,

$$\begin{aligned} s^j &= x^{j+1} - x^j \text{ for } j = \underline{k}, \ldots, k-1, \\ y^j &= F(x^{j+1}) - F(x^j) \text{ for } j = \underline{k}, \ldots, k-1, \\ s^k &= x^{k+1}_{\text{trial}} - x^k, \\ y^k &= F(x^{k+1}_{\text{trial}}) - F(x^k), \\ S_k &= (s^{\underline{k}}, \ldots, s^{k-1}, s^k), \\ Y_k &= (y^{\underline{k}}, \ldots, y^{k-1}, y^k) \end{aligned}$$

and

$$x^{k+1}_{\text{accel}} = x^k - S_k Y_k^\dagger F(x^k), \tag{48}$$

where $Y_k^\dagger$ is the Moore–Penrose pseudoinverse of Y_k.

Step 3. Choose $x^{k+1} \in \{x^{k+1}_{\text{trial}}, x^{k+1}_{\text{accel}}\}$ such that $\|F(x^{k+1})\| = \min\{\|F(x^{k+1}_{\text{trial}})\|, \|F(x^{k+1}_{\text{accel}})\|\}$.

The theorem below helps to explain the behavior of Algorithm 2.1 with x^{k+1} given by Algorithm 4.1, called Algorithms 2.1–4.1 from now on. Briefly speaking, we are going to prove that, under some assumptions, the sequence generated by Algorithms 2.1–4.1 is such that $\{F(x^k)\}$ converges superlinearly to zero. Thus, if $\{x^k\}$ converges to x_* and $J(x_*)$ is nonsingular, the convergence of $\{x^k\}$ to x_* is superlinear. We do not mean that these assumptions are "reasonable" in the sense that they usually, or even frequently, hold. The theorem aims to show the correlation between different properties of the method, which is probably useful for understanding the algorithm and, perhaps, for seeking modifications and improvements.

THEOREM 4.1. *Assume that $\{x^k\}$ is the sequence generated by Algorithms 2.1–4.1 with $p > 1$. In addition, suppose that the following hold:*

H1: *For all k large enough we have that $x^{k+1} = x^{k+1}_{\text{accel}}$.*

H2: *There exists a positive sequence $\beta_k \to 0$ such that for all k large enough, the columns of Y_k are linear independent and*

$$\left\|(S_{k+1}Y_{k+1}^{\dagger} - S_kY_k^{\dagger})y^k\right\| \leq \beta_k\|y^k\|. \tag{49}$$

H3: *There exists $c > 0$ such that*

$$\|S_kY_k^{\dagger}F(x^{k+1})\| \geq c\|F(x^{k+1})\| \tag{50}$$

for k large enough.

Then, $\|F(x^k)\|$ converges Q-superlinearly to 0.

Proof. By (48) and H1, we have that, for k large enough,

$$x^{k+1} = x^k - S_kY_k^{\dagger}F(x^k). \tag{51}$$

But, by the definition of S_{k+1} and Y_{k+1}, the linear independence of the columns of Y_{k+1}, and the fact that $p > 1$, for all k large enough,

$$S_{k+1}Y_{k+1}^{\dagger}y^k = s^k.$$

Therefore, by (49) in H2,

$$\|S_kY_k^{\dagger}y^k - s^k\| \leq \beta_k\|y^k\|.$$

Then, by (51),

$$\|S_kY_k^{\dagger}F(x^{k+1})\| \leq \beta_k\|y^k\|.$$

Thus, by (50) in H3,

$$c\|F(x^{k+1})\| \leq \beta_k\|y^k\|.$$

So,

$$c\|F(x^{k+1})\| \leq \beta_k(\|F(x^{k+1})\| + \|F(x^k)\|).$$

Thus,

$$(c - \beta_k)\|F(x^{k+1})\| \leq \beta_k\|F(x^k)\|.$$

Therefore,

$$\|F(x^{k+1})\| \leq \frac{\beta_k}{c - \beta_k}\|F(x^k)\|$$

for k large enough, which means that $F(x^k)$ tends to zero Q-superlinearly. ∎

If (i) $p = n$, (ii) the initial point x^0 is close enough to a solution x^*, (iii) the Jacobian $J(x^*)$ is nonsingular, and (iv) the increments $s^j = x^{j+1} - x^j$ for $j = \underline{k}, \ldots, k-1$ (columns of S_k) are uniformly linearly independent [51], then the matrices $S_k Y_k^\dagger$ converge to $J(x^*)^{-1}$. So, hypothesis H2 trivially holds. In addition, under conditions (i) to (iv), the sequence given by $x^{k+1} = x^k - S_k Y_k^\dagger F(x^k)$ converges superlinearly to x^*; see [51]. Therefore the acceleration is successful and accepted for generating the new iterate, i.e., hypothesis H1 also holds. Moreover, by the nonsingularity of $J(x^*)$ and the convergence of $S_k Y_k^\dagger$ to $J(x^*)^{-1}$, H3 also holds. The uniformly linear independence of the increments s^j can be guaranteed using well-known procedures; see [46]. Conditions (i) to (iv) are sufficient assumptions. Of course we expect that the hypotheses of Theorem 4.1 hold under much weaker conditions.

5. Implementation. In this section, we discuss the implementation of Algorithms 2.1–4.1.

5.1. Stopping criterion. At Step 1 of Algorithm 2.1, given $\varepsilon > 0$, we replace the stopping criterion $\|F(x^k)\| = 0$ with

$$\|F(x^k)\|_2 \le \varepsilon. \tag{52}$$

To cope with the situation of a problem for which $x \in \mathbb{R}^n$ such that $F(x) = 0$ does not exist, we also consider, at Step 1, the stopping criterion

$$k \ge k_{\max}, \tag{53}$$

where $k_{\max} \ge 0$ is a given parameter.

5.2. Choice of the direction and the scaling factor. At Step 2 of Algorithm 2.1, we must choose σ_k and v_k. For this purpose, we considered the residual choice $v^k = -F(x^k)$ whenever a random direction is not prescribed. When a random direction is required, $v^k = \|F(x^k)\| v^k_{\text{ran}}$, where v^k_{ran} is a random vector uniformly chosen in the unit sphere. Vector v^k_{ran} is computed as $v_k = z/\|z\|$, where $z \sim \mathcal{N}(0, I_n)$, i.e., $z = (z_1, z_2, \ldots, z_n)^T$ and $z_i \sim \mathcal{N}(0,1)$ for $i = 1, \ldots, n$. Arbitrarily, we set $\sigma_0 = 1$. A natural choice for σ_k ($k \ge 1$) would be the spectral step given by

$$\sigma_k^{\text{spg}} = \frac{(x^k - x^{k-1})^T (x^k - x^{k-1})}{(x^k - x^{k-1})^T (F(x^k) - F(x^{k-1}))}$$

with safeguards that guarantee that $|\sigma_k|$ belong to $[\sigma_{\min}, \sigma_{\max}]$. It turns out that defining $\sigma_{\max} > 1$, preliminary numerical experiments showed that, *in the problems considered in the present work*, Step 2 of Algorithm 2.1 with this choice of σ_k performs several backtrackings per iteration that result in a total number of functional evaluations one order of magnitude larger than the number of iterations, which is an unusual behavior of nonmonotone methods based on the Barzilai–Borwein spectral choice [6, 7, 8, 9, 41, 42, 55, 56]. On the other hand, it was also observed that, at Step 3 of Algorithm 4.1, the acceleration step was always chosen. This means that the costly obtained x^{k+1}_{trial} was always beaten by x^{k+1}_{accel}. These two observations suggested that a more conservative, i.e., small, scaling factor σ_k should be considered. A small σ_k could result in a trial point $x^{k+1}_{\text{trial}} = x^k + \alpha_k d^k$ with $\alpha_k = \pm 1$ that satisfies (6), i.e., no backtracking, and that provides the information required to compute a successful x^{k+1}_{accel} at Step 2 of Algorithm 4.1.

The conservative choice of σ_k employed in our implementation is as follows. We consider an algorithmic parameter

$$h_{\text{init}} > 0$$

and we define

$$\sigma_{\min} = \sqrt{\epsilon},\ \sigma_{\max} = 1, \tag{54}$$

$$\sigma_0 = 1,$$

$$\bar{\sigma}_k = h_{\text{init}} \frac{\|x^k - x^{k-1}\|}{\|F(x^k)\|} \text{ for all } k \geq 1,$$

and

$$\sigma_k = \bar{\sigma}_k \text{ if } k \geq 1 \text{ and } \bar{\sigma}_k \in [\max\{1, \|x^k\|\}\sigma_{\min}, \sigma_{\max}].$$

Finally, if $k \geq 1$ and $\bar{\sigma}_k \notin [\max\{1, \|x^k\|\}\sigma_{\min}, \sigma_{\max}]$, we define

$$\bar{\bar{\sigma}}_k = h_{\text{init}} \frac{\|x^k\|}{\|F(x^k)\|}$$

and we compute σ_k as the projection of $\bar{\bar{\sigma}}_k$ onto the interval $[\max\{1, \|x^k\|\}\sigma_{\min}, \sigma_{\max}]$. In (54), $\epsilon \approx 10^{-16}$ is the machine precision.

5.3. Procedure for reducing the step. At Step 2.4 of Algorithm 2.1, we compute α_+^{new} as the minimizer of the univariate quadratic $q(\alpha)$ that interpolates $q(0) = f(x^k)$, $q(\alpha_+) = f(x^k - \alpha_+\sigma_k F(x^k))$, and $q'(0) = -\sigma_k F(x^k)^T \nabla f(x^k) = -\sigma_k F(x^k)^T J(x^k) F(x^k)$. Following [41], since we consider $J(x^k)$ unavailable, we consider $J(x^k) = I$. Thus,

$$\alpha_+^{\text{new}} = \max\left\{\tau_{\min}\alpha_+, \min\left\{\frac{\alpha_+^2 f(x^k)}{f(x^k - \alpha_+\sigma_k F(x^k)) + (2\alpha_+ - 1)f(x^k)}, \tau_{\max}\alpha_+\right\}\right\}.$$

Analogously,

$$\alpha_-^{\text{new}} = \max\left\{\tau_{\min}\alpha_-, \min\left\{\frac{\alpha_-^2 f(x^k)}{f(x^k + \alpha_-\sigma_k F(x^k)) + (2\alpha_- - 1)f(x^k)}, \tau_{\max}\alpha_-\right\}\right\}.$$

5.4. Computing the acceleration. In Step 2 of Algorithm 4.1, computing x_{accel}^{k+1} as defined in (48) is equivalent to computing the minimum-norm least-squares solution $\bar{\omega}$ to the linear system $Y_k\omega = F(x_{\text{trial}}^{k+1})$ and defining $x_{\text{accel}}^{k+1} = x_{\text{trial}}^{k+1} - S_k\bar{\omega}$. In practice, we compute the minimum-norm least-squares solution with a complete orthogonalization of Y_k. Computing this factorization from scratch at each iteration could be expensive. However, by definition, Y_k corresponds to a slight variation of Y_{k-1}. In practice, when Y_k does not have full numerical rank, one extra column is added to it, and if Y_k has null numerical rank, a new Y_k is computed from scratch. For completeness, the practical implementation of Algorithm 4.1 is given below.

Practical implementation of Algorithm 4.1. Let $0 < h_{\text{small}} < h_{\text{large}}$ and $p \geq 1$ be given. Set $r_{\max} \leftarrow 0$ and $\ell \leftarrow 1$.

Step 1. Define $x_{\text{trial}}^{k+1} = x^k + \alpha_k d^k$.

Step 2.

Step 2.1. If $k = 0$, then set S_k and Y_k as matrices with zero columns. Otherwise, set $S_k \leftarrow S_{k-1}$ and $Y_k \leftarrow Y_{k-1}$.

Step 2.2. If S_k and Y_k have p columns, then remove the leftmost column of S_k and Y_k.

Step 2.3. Add $s^k = x_{\text{trial}}^{k+1} - x^k$ and $y^k = F(x_{\text{trial}}^{k+1}) - F(x^k)$ as the rightmost column of S_k and Y_k, respectively, and set $r_{\max} \leftarrow \max\{r_{\max}, \text{rank}(Y_k)\}$.

Step 2.4. If $\text{rank}(Y_k) < r_{\max}$, then execute Steps 2.4.1–2.4.2.

Step 2.4.1. If S_k and Y_k have p columns, then remove the leftmost column of S_k and Y_k.

Step 2.4.2. Add $s_{\text{extra}} = x_{\text{extra}} - x^k$ and $y_{\text{extra}} = F(x_{\text{extra}}) - F(x^k)$ as the rightmost column of S_k and Y_k, respectively, where $x_{\text{extra}} = x^k + h_{\text{small}}\, e_\ell$. Set $r_{\max} \leftarrow \max\{r_{\max}, \text{rank}(Y_k)\}$ and $\ell \leftarrow \text{mod}(\ell, n) + 1$.

Step 2.5. If $\text{rank}(Y_k) \neq 0$, then execute Steps 2.5.1–2.5.3.

Step 2.5.1. Compute the minimum-norm least-squares solution $\bar{\omega}$ to the linear system $Y_k\omega = F(x^k)$ and define $x_{\text{accel}}^{k+1} = x^k - S_k\bar{\omega}$.

Step 2.5.2. If Step 2.4.2 was executed, then remove the rightmost column of S_k and Y_k, i.e., columns s_{extra} and y_{extra}.

Step 2.5.3. If $x_{\text{accel}}^{k+1} \neq x^k$, $\|x_{\text{accel}}^{k+1}\| \leq 10 \max\{1, \|x^k\|\}$, and $\|F(x_{\text{accel}}^{k+1})\| < \|F(x_{\text{trial}}^{k+1})\|$, then redefine $x_{\text{trial}}^{k+1} = x_{\text{accel}}^{k+1}$, substitute the rightmost column of S_k and Y_k, i.e., columns s^k and y^k, with $s_{\text{accel}} = x_{\text{accel}}^{k+1} - x^k$ and $y_{\text{accel}} = F(x_{\text{accel}}^{k+1}) - F(x^k)$, respectively, and set $r_{\max} \leftarrow \max\{r_{\max}, \text{rank}(Y_k)\}$.

Step 2.6. If $\text{rank}(Y_k) = 0$, then execute Steps 2.6.1–2.6.3.

Step 2.6.1. Redefine matrices S_k and Y_k as matrices with zero columns.

Step 2.6.2. Execute Steps 2.6.2.1 $p - 1$ times.

Step 2.6.2.1. Add $s_{\text{extra}} = x_{\text{extra}} - x_{\text{trial}}^{k+1}$ and $y_{\text{extra}} = F(x_{\text{extra}}) - F(x_{\text{trial}}^{k+1})$ as rightmost column of S_k and Y_k, respectively, where $x_{\text{extra}} = x^k + h_{\text{large}}\, e_\ell$. Set $r_{\max} \leftarrow \max\{r_{\max}, \text{rank}(Y_k)\}$ and $\ell \leftarrow \text{mod}(\ell, n) + 1$.

Step 2.6.3. Add $s^k = x_{\text{trial}}^{k+1} - x^k$ and $y^k = F(x_{\text{trial}}^{k+1}) - F(x^k)$ as rightmost column of S_k and Y_k, respectively, and set $r_{\max} \leftarrow \max\{r_{\max}, \text{rank}(Y_k)\}$.

Step 2.7. Compute the minimum-norm least-squares solution $\bar{\omega}$ to the linear system $Y_k\omega = F(x^k)$ and define $x_{\text{accel}}^{k+1} = x^k - S_k\bar{\omega}$.

Step 2.8. If $x_{\text{accel}}^{k+1} \neq x^k$, $\|x_{\text{accel}}^{k+1}\| \leq 10 \max\{1, \|x^k\|\}$, and $\|F(x_{\text{accel}}^{k+1})\| < \|F(x_{\text{trial}}^{k+1})\|$, then redefine $x_{\text{trial}}^{k+1} = x_{\text{accel}}^{k+1}$ and substitute the rightmost column of S_k and Y_k, i.e., columns s^k and y^k, with $s_{\text{accel}} = x_{\text{accel}}^{k+1} - x^k$ and $y_{\text{accel}} = F(x_{\text{accel}}^{k+1}) - F(x^k)$, respectively, and set $r_{\max} \leftarrow \max\{r_{\max}, \text{rank}(Y_k)\}$.

Step 3. Define $x^{k+1} = x_{\text{trial}}^{k+1}$.

In general, Step 2.6 is not executed. If iteration k is such that Step 2.6 was not executed in the previous p iterations, then matrices S_k and Y_k correspond to removing the leftmost column from and adding a new rightmost column to S_{k-1} and Y_{k-1}, respectively. The leftmost column is not removed if the maximum number of columns p was not reached yet. When $k = 0$, Y_k is a single-column matrix for which

matrices P, Q, and R of the rank-revealing QR decomposition $Y_k P = QR$ can be trivially computed. For $k > 0$, the QR decomposition of Y_k can be obtained with time complexity $O(n\,p)$ by updating, via Givens rotations, the QR decomposition of Y_{k-1}. Moreover, by using the QR decomposition of Y_k, the minimum-norm least-squares solution $\bar{\omega}$ can be computed with time complexity $O(n + p^2)$ if Y_k is full-rank and with time complexity $O(n+p^3)$ in the rank-deficient case. Summing up, by assuming that $p \ll n$ and that p does not depend on n, iterations of this implementation of Algorithm 4.1 can be implemented with time complexity $O(n)$. The space complexity is $O(n\,p + p^2)$ and it is related to the fact that we must save matrix $S_k \in \mathbb{R}^{n\times p}$ and matrices $Q \in \mathbb{R}^{n\times p}$ and $R \in \mathbb{R}^{p\times p}$ of the QR decomposition of Y_k. Of course, the permutation matrix P can be saved in an array of size p. Thus, the space complexity of this implementation of Algorithm 4.1 is also $O(n)$ under the same assumptions. When Step 2.6 is executed, the QR decomposition of Y_k must be computed from scratch, with time complexity $O(n\,p^2)$.

6. Numerical experiments. We implemented Algorithm 2.1 and the practical implementation of Algorithm 4.1 (described in section 5.4) in Fortran 90. To allow reproducibility, the source code necessary to reproduce all numerical experiments described in this section is available at http://www.ime.usp.br/∼egbirgin/. Normally distributed pseudorandom numbers are generated with routine `r8_normal_01` from J. Burkardt, available at https://people.sc.fsu.edu/∼jburkardt/f_src/normal/normal.html. In the numerical experiments, we considered the standard values (in the context of algorithms that consider the spectral step and a nonmonotone line search, like DF-SANE) $\gamma = 10^{-4}$, $\tau_{\min} = 0.1$, $\tau_{\max} = 0.5$, $M = 10$, $k_{\mathrm{mon}} = \infty$, $\alpha_{\mathrm{small}} = \sqrt{\epsilon}$, $\sigma_{\min} = \sqrt{\epsilon}$, $\sigma_{\max} = 1/\sqrt{\epsilon}$, and $\eta_k = 2^{-k}\min\{\frac{1}{2}\|F(x^0)\|, \sqrt{\|F(x^0)\|}\}$, where $\epsilon \approx 10^{-16}$ is the machine precision. More crucial to the method performance are the choices of h_{init}, h_{small}, h_{large}, and p, whose values are mentioned below. All tests were conducted on a computer with a 3.4 GHz Intel Core i5 processor and 8 GB 1600 MHz DDR3 RAM memory, running macOS Mojave (version 10.14.6). Code was compiled by the GFortran compiler of GCC (version 8.2.0) with the -O3 optimization directive enabled.

6.1. Two- and three-dimensional Bratu problem. In a first experiment, we considered two- and three-dimensional (2D and 3D) versions of the Bratu problem

$$-\Delta u + \theta e^u = \phi(\bar{u}) \text{ in } \Omega \tag{55}$$

with boundary conditions $u = \bar{u}$ on $\partial\Omega$. In the 2D case, $\Omega = [0,1]^2$ and, following [39], we set $\bar{u} = 10u_1u_2(1-u_1)(1-u_2)e^{u_1^{4.5}}$, while in the 3D case, $\Omega = [0,1]^3$ and $\bar{u} = 10u_1u_2u_3(1-u_1)(1-u_2)(1-u_3)e^{u_1^{4.5}}$. In both cases, $\phi(u) = -\Delta u + \theta e^u$, so the problem has $\bar{u}$ as a known solution. Considering n_p discretization points in each dimension and approximating

$$\Delta u(x) \approx \frac{u(x \pm he_1) + u(x \pm he_2) - 4u(x)}{h^2}$$

and

$$\Delta u(x) \approx \frac{u(x \pm he_1) + u(x \pm he_2) + u(x \pm he_3) - 6u(x)}{h^2},$$

where $h = 1/(n_p - 1)$ and e_i is the ith canonical vector in the corresponding space ($\mathbb{R}^2$ or $\mathbb{R}^3$), we obtain nonlinear systems of equations with $n = (n_p - 2)^2$ and $n =$

$(n_p - 2)^3$ variables in the 2D and 3D cases, respectively. Starting from $u = 0$, fixing $\theta = -100$, and varying $n_p \in \{100, 125, \ldots, 400\}$ and $n_p \in \{10, 15, \ldots, 70\}$ in the 2D and 3D cases, respectively, we run NITSOL (file NITSOL,11-1-05.TAR.GZ downloaded from https://users.wpi.edu/∼walker/NITSOL/ on October 26, 2020) and the method proposed in the present work, which will be called Accelerated DF-SANE from now on. As stopping criteria, we considered (52) with $\varepsilon = 10^{-6}\sqrt{n}$ and (53) with $k_{\max} =$ 100,000. For NITSOL, we use all its default parameters, except the maximum number of (nonlinear) iterations, which was increased in order to avoid premature stops. By default, NITSOL corresponds to an inexact Newton method in which Newtonian systems are solved with GMRES (maximum Krylov subspace dimension equal to 20), approximating the Jacobian-vector products by finite-differences. For Accelerated DF-SANE, based on preliminary experiments, we set $h_{\text{small}} = 10^{-4}$, $h_{\text{large}} = 0.1$, and $h_{\text{init}} = 0.01$ in the 2D case and $h_{\text{small}} = h_{\text{large}} = 0.1$ and $h_{\text{init}} = 1$ in the 3D case. In both cases, we considered $p = 5$.

Tables 1 and 2 show the results. In the tables, $\#\text{it}_1$ corresponds to the nonlinear iterations (outer iterations) of NITSOL, while $\#\text{it}_2$ corresponds to its linear iterations (inner or GMRES iterations). For both methods, "fcnt" stands for number of evaluations of F, "Time" stands for CPU time in seconds, and "SC" stands for "stopping criterion." Remaining columns are self-explanatory. In the case of NITSOL, a stopping criterion equal to 0 means success, while 6 means "failure to reach an acceptable step through backtracking." Accelerated DF-SANE satisfied the stopping criterion (52) related to success in all problems. Regarding the 2D problems, it should be first noted that NITSOL failed in satisfying the stopping criterion for $n_p \geq 225$. Accelerated DF-SANE is faster than NITSOL in all problems that both methods solved. In the larger problem that both methods solved, Accelerated DF-SANE is around 30 times faster than NITSOL. Regarding the 3D problems (Table 2), both methods satisfied the stopping criterion related to success in all problems. Accelerated DF-SANE is faster than NITSOL in all problems in the table. In the larger problem in the table ($n_p = 70$), Accelerated DF-SANE is more than 20 times faster than NITSOL.

The difficulty of the Bratu problem varies with the value of θ, which may be positive or negative. For the formulation given in (55), negative values of θ correspond to more difficult problems. Therefore, results in Tables 1 and 2 raise the question

TABLE 1
Performance of NITSOL and Accelerated DF-SANE in the 2D Bratu problem.

n_p	n	NITSOL (Newton-GMRES)						Accelerated DF-SANE			
		SC	$\|F(x_*)\|_2$	$\#\text{it}_1$	$\#\text{it}_2$	fcnt	Time	$\|F(x_*)\|_2$	#it	fcnt	Time
100	9,604	0	9.7e−05	220	197,732	197,953	39.63	9.8e−05	5,219	10,688	4.85
125	15,129	0	1.2e−04	631	473,616	474,248	151.88	1.2e−04	2,612	5,489	3.65
150	21,904	0	1.5e−04	379	315,338	315,718	165.07	1.5e−04	2,946	6,007	6.03
175	29,929	0	1.7e−04	401	331,380	331,782	265.53	1.7e−04	4,475	10,007	13.93
200	39,204	0	2.0e−04	4,421	846,957	851,385	983.97	2.0e−04	6,775	14,385	28.22
225	49,729	6	3.3e+01	3,944	173,310	177,283	261.03	2.2e−04	4,328	8,927	24.63
250	61,504	6	4.1e+01	105	46,897	47,019	82.69	2.5e−04	12,661	26,353	86.73
275	74,529	6	5.3e+01	1,411	94,194	95,635	224.74	2.7e−04	8,809	19,583	79.08
300	88,804	6	7.2e+01	1,341	104,735	106,112	317.71	3.0e−04	15,858	34,194	173.36
325	104,329	6	9.6e+01	2,430	194,640	197,126	705.53	3.2e−04	10,791	23,403	142.58
350	121,104	6	1.4e+02	195	82,940	83,151	319.00	3.5e−04	10,805	25,915	161.67
375	139,129	6	1.7e+02	1,127	121,100	122,243	649.94	3.7e−04	16,335	38,648	310.91
400	158,404	6	2.2e+02	2,936	144,460	147,415	851.67	4.0e−04	21,106	55,901	483.90

TABLE 2
Performance of NITSOL and Accelerated DF-SANE in the 3D Bratu problem.

n_p	n	NITSOL (Newton-GMRES)					Accelerated DF-SANE			
		$\|F(x_*)\|_2$	#it$_1$	#it$_2$	fcnt	Time	$\|F(x_*)\|_2$	#it	fcnt	Time
10	512	1.7e−05	5	213	219	0.00	2.1e−05	126	308	0.00
15	2,197	3.7e−05	7	1,621	1,629	0.07	4.7e−05	223	662	0.05
20	5,832	6.1e−05	11	7,157	7,169	0.97	7.2e−05	1,221	4,271	0.90
25	12,167	8.8e−05	19	15,893	15,913	4.62	1.1e−04	529	1,840	0.75
30	21,952	1.2e−04	25	21,991	22,017	12.72	1.4e−04	812	3,012	2.40
35	35,937	1.6e−04	38	34,935	34,974	36.31	1.9e−04	1,210	4,530	6.53
40	54,872	2.2e−04	70	66,115	66,186	118.56	2.3e−04	1,109	4,379	8.74
45	79,507	2.8e−04	147	137,626	137,774	386.09	2.8e−04	1,315	5,444	16.72
50	110,592	3.3e−04	217	199,042	199,260	759.61	3.3e−04	1,574	6,501	27.16
55	148,877	3.8e−04	357	312,023	312,381	1736.95	3.8e−04	1,706	7,254	43.95
60	195,112	4.4e−04	504	418,201	418,706	3074.25	4.4e−04	1,821	8,019	66.28
65	250,047	5.0e−04	988	691,192	692,181	6715.18	4.9e−04	2,061	9,379	102.73
70	314,432	5.6e−04	609	486,651	487,261	5735.90	5.6e−04	1,836	8,431	116.34

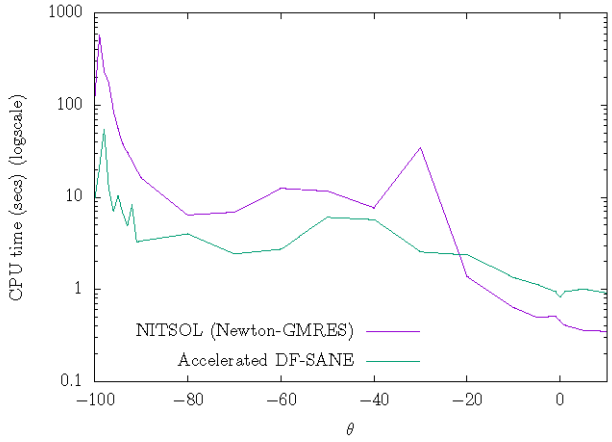

FIG. 1. *CPU time (in seconds) used by NITSOL and Accelerated DF-SANE to solve the 3D Bratu problem with* $n_p = 40$ *and* $\theta \in [-100, 10]$.

of how the performances of the methods compare to each other for different values of θ. So, we arbitrarily considered the 3D Bratu problem with $n_p = 40$, which is affordable for both methods and varied $\theta \in [-100, 10]$. Figure 1 shows the results of applying NITSOL and Accelerated DF-SANE. The graphic shows that for $\theta \geq -20$ both methods use less than a second and NITSOL outperforms Accelerated DF-SANE. On the other hand, in the most difficult problems (i.e., $\theta < -20$), where up to 1,000 seconds are required, Accelerated DF-SANE outperforms NITSOL by approximately an order of magnitude. (Note that the y-axis is in logarithmic scale.) Considering the number of functional evaluations as a performance metric, instead of the CPU time, results are mostly the same.

Another relevant question is how the performance of Accelerated DF-SANE varies as a function of its parameter p. So, considering again the 3D Bratu problem with $\theta = -100$ and $n_p = 40$, we ran Accelerated DF-SANE with $p \in \{3, 4, \ldots, 17\}$. Figure 2 shows the results. The figure on the left shows that the number of iterations of Accelerated DF-SANE is nearly constant as a function of p, while the figure on the right shows, as expected, that the larger the p, the larger the CPU time per iteration.

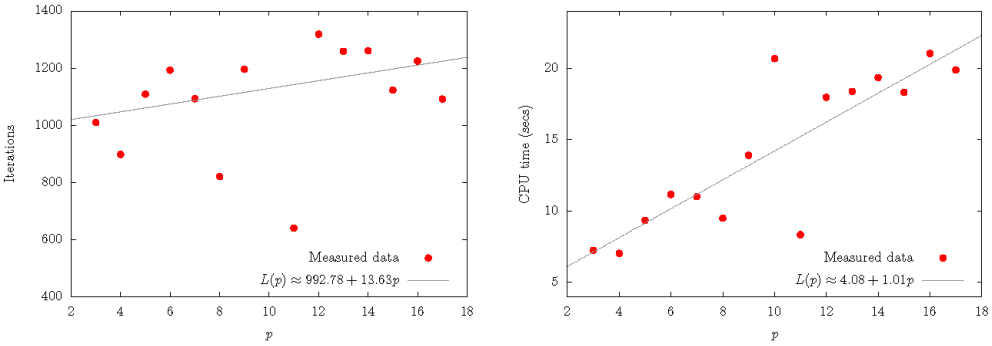

FIG. 2. *Performance of Accelerated DF-SANE in the 3D Bratu problem with* $n_p = 40$ *and* $\theta = -100$ *varying the number* p *of columns in matrices* S_k *and* Y_k.

Note that for the considered values of p, the variation on the number of iterations is up to 20%, while the variation in CPU time is up to 300%.

Another relevant question regarding Accelerated DF-SANE is how it compares against DF-SANE, i.e., the original method without the acceleration being proposed in the present work. We were unable to run DF-SANE in the set of problems considered in Tables 1 and 2, because DF-SANE is unable to reach the stopping criterion (52) with $\varepsilon = 10^{-6}\sqrt{n}$ within an affordable time. As an alternative, we considered the four instances of the 3D Bratu problem with $n_p \in \{40, 70\}$ and $\theta \in \{-100, 10\}$. In the two instances with $\theta = 10$, DF-SANE was able to satisfy the imposed stopping criterion within an affordable time. This result was in fact expected because for this value of θ the Bratu problem resembles the minimization of a convex quadratic function, and Barzilai–Borwein or spectral step based methods are expected to perform especially well in this case. In the two instances with $\theta = -100$, we first run Accelerated DF-SANE and then we run DF-SANE using as CPU time limit the time consumed by Accelerated DF-SANE. Figure 3 shows the results. It is very clear from the four graphics that the acceleration process is very efficient in its purpose of accelerating DF-SANE. (In the cases with $\theta = -100$, with excruciatingly slow progress, DF-SANE is far from reaching convergence after a couple of hours of CPU time.)

6.2. Modified Bratu, driven cavity, and flow in a porous medium problems. In a second set of experiments, we considered three PDE-based problems described in [53]. Fortran implementations of these problems are included as examples of usage in the NITSOL distribution. The three problems are discretizations of 2D PDE-based systems of nonlinear equations. All parameters of the problems were set as suggested in [53]. We varied the value of n_p for the three problems, trying to illustrate the behavior of the methods and solving problems that are as large as possible with both methods within an affordable time. NITSOL was run with all its default parameters, as in the previous section, while Accelerated DF-SANE was run with the same parameters already reported for the 2D Bratu problem in the previous section. Tables 3–5 show the results. Once again, Accelerated DF-SANE satisfied the stopping criterion (52) related to success in all problems. Table 3 shows that NITSOL failed in satisfying the stopping criterion for $n_p \geq 135$ in the driven cavity problem. Accelerated DF-SANE was faster than NITSOL in all instances that both methods solved. In the largest instance that both methods solved, Accelerated DF-SANE is more than 30 times faster than NITSOL. Table 4 shows that NITSOL is faster than Accelerated

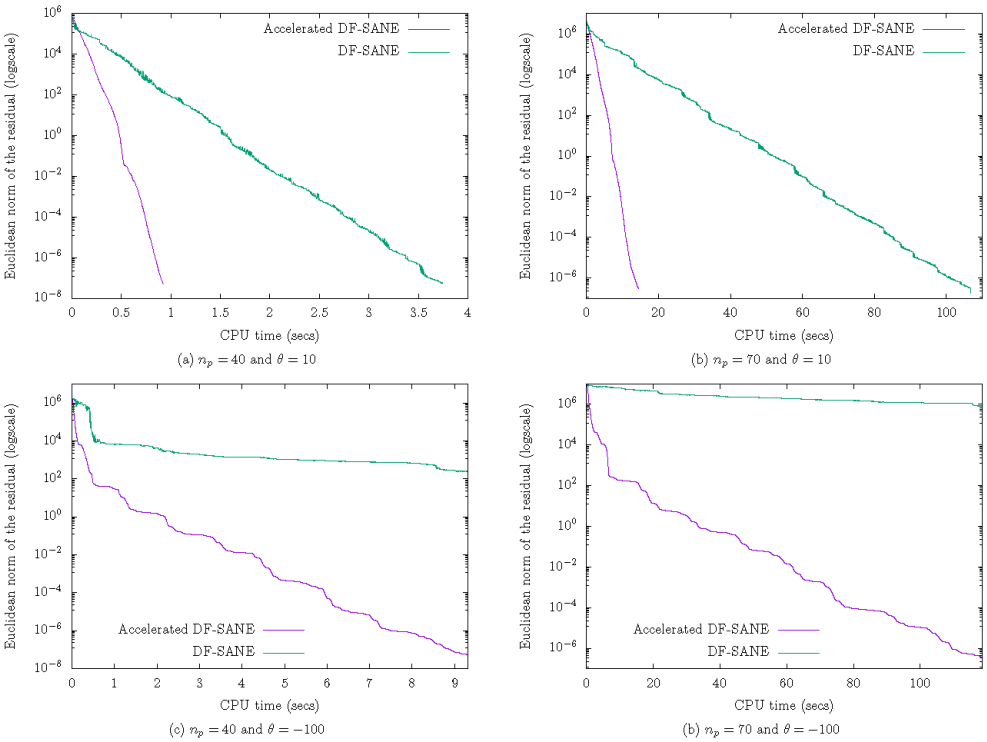

FIG. 3. *Performance of DF-SANE and Accelerated DF-SANE in four instances of the* 3*D Bratu problem with* $\theta \in \{-100, 10\}$ *and* $n_p \in \{40, 70\}$.

DF-SANE in the smaller instances of the generalized 2D Bratu problem (n_p up to 750). In the largest instance in the table ($n_p = 1,500$), Accelerated DF-SANE takes around half of the time required by NITSOL. Finally, Table 5 shows that NITSOL is faster than Accelerated DF-SANE in the smaller instances (n_p up to 400) of the flow in a porous medium problem. In the largest instance in the table ($n_p = 1,500$), Accelerated DF-SANE takes around half of the time required by NITSOL.

6.3. Comparison with Anderson mixing. In a third set of experiments, we compared Accelerated DF-SANE with Anderson mixing, given by (46)–(47) and described in [30, sect. 2.5]. According to [29] and [30], Anderson mixing is equivalent to a generalization of Broyden's second method. Once again, we do not consider preconditioners in the experiments, since they could be used in all problems regardless of the method considered. Anderson mixing was implemented in Fortran 90. In the implementation, the minimum-norm solution to the least-squares problems is computed with the same updated QR decomposition that is used in the Accelerated DF-SANE method, as described at the end of section 5.4.

The experiments in [30] make it clear that the performance of Anderson mixing depends strongly on the choice of the β parameter and that this parameter varies from problem to problem and even when the same problem with different dimensions is solved; see, for example, [30, sect. 5.1], where a variant of the Bratu problem is solved. As a consequence, it would be impracticable to address each of the five problems in

TABLE 3
Performance of NITSOL and Accelerated DF-SANE in the driven cavity problem.

n_p	n	NITSOL (Newton-GMRES)						Accelerated DF-SANE			
		SC	$\|F(x_*)\|_2$	#it$_1$	#it$_2$	fcnt	Time	$\|F(x_*)\|_2$	#it	fcnt	Time
10	100	0	7.7e−06	5	194	200	0.00	9.6e−06	96	200	0.00
15	225	0	1.0e−05	6	988	995	0.00	1.4e−05	190	392	0.00
20	400	0	1.6e−05	6	2,739	2,746	0.02	1.9e−05	278	582	0.01
25	625	0	2.0e−05	10	6,273	6,284	0.09	2.4e−05	473	1,044	0.03
30	900	0	2.4e−05	15	11,939	11,955	0.24	2.9e−05	663	1,519	0.06
35	1,225	0	3.1e−05	23	20,058	20,082	0.57	3.4e−05	889	2,118	0.12
40	1,600	0	3.8e−05	36	32,819	32,856	1.19	3.9e−05	1,141	2,954	0.20
45	2,025	0	4.4e−05	56	52,200	52,257	2.52	4.4e−05	1,431	3,706	0.32
50	2,500	0	5.0e−05	80	74,981	75,062	4.68	5.0e−05	1,747	4,653	0.49
55	3,025	0	5.2e−05	114	106,481	106,596	8.27	5.5e−05	2,093	6,010	0.75
60	3,600	0	5.8e−05	154	142,222	142,377	13.09	6.0e−05	2,470	7,346	1.04
65	4,225	0	6.3e−05	217	195,849	196,067	21.29	6.5e−05	2,880	8,824	1.46
70	4,900	0	6.8e−05	278	244,149	244,428	30.60	7.0e−05	3,297	10,533	1.96
75	5,625	0	7.4e−05	392	328,630	329,023	48.01	7.5e−05	3,781	12,557	2.69
80	6,400	0	7.8e−05	568	442,630	443,199	72.66	8.0e−05	4,274	14,564	3.42
85	7,225	0	8.4e−05	790	563,030	563,821	104.84	8.5e−05	4,811	16,905	4.45
90	8,100	0	9.0e−05	1,135	706,031	707,167	147.14	9.0e−05	5,369	19,423	5.64
95	9,025	0	9.5e−05	1,661	856,631	858,293	200.43	9.5e−05	5,949	22,052	6.99
100	10,000	0	1.0e−04	3,051	1040,211	1,043,263	269.07	1.0e−04	6,563	24,901	8.64
105	11,025	0	1.0e−04	13,222	1233,251	1,246,474	357.23	1.0e−04	7,208	27,977	10.53
110	12,100	0	1.1e−04	20,017	1415,812	1,435,830	452.47	1.1e−04	7,593	30,394	12.36
115	13,225	0	1.1e−04	18,717	1290,772	1,309,495	458.66	1.1e−04	8,543	34,375	15.36
120	14,400	0	1.2e−04	18,350	1461,992	1,480,349	568.50	1.2e−04	9,265	37,904	18.14
125	15,625	0	1.2e−04	26,341	1628,992	1,655,343	696.52	1.2e−04	10,036	41,724	21.39
130	16,900	0	1.3e−04	13,114	1895,392	1,908,981	903.98	1.3e−04	10,824	45,718	25.66
135	18,225	6	1.0e−01	6,604	370,973	377,706	190.84	1.3e−04	11,639	49,969	29.41
140	19,600	6	1.3e−01	6,616	302,593	309,288	171.16	1.4e−04	12,406	53,796	34.15
145	21,025	6	1.5e−01	6,661	278,673	285,397	170.61	1.4e−04	12,814	56,423	38.12
150	22,500	6	1.7e−01	7,048	269,053	276,151	178.26	1.5e−04	13,741	61,042	43.95
155	24,025	6	1.9e−01	6,531	273,073	279,666	197.21	1.5e−04	14,651	65,802	50.28
160	25,600	6	2.0e−01	6,204	261,354	267,616	203.83	1.6e−04	15,509	70,613	57.77
165	27,225	6	2.1e−01	6,897	265,594	272,543	221.86	1.6e−04	15,310	69,910	60.14
170	28,900	6	2.2e−01	5,214	286,714	292,023	257.46	1.7e−04	17,338	80,401	74.03
175	30,625	6	2.6e−01	5,618	255,294	260,978	244.98	1.7e−04	18,015	83,984	81.03
180	32,400	6	2.9e−01	5,260	237,754	243,072	242.00	1.8e−04	18,510	86,308	89.89
185	34,225	6	3.1e−01	5,460	233,394	238,906	253.32	1.8e−04	19,457	91,745	101.76
190	36,100	6	3.4e−01	5,741	217,814	223,590	259.29	1.9e−04	20,846	99,200	119.78
195	38,025	6	3.5e−01	5,539	226,995	232,580	280.39	1.9e−04	20,135	97,440	121.89
200	40,000	6	3.5e−01	4,969	241,335	246,370	311.54	2.0e−04	21,875	107,288	140.57

sections 6.1 and 6.2 by varying the values of n_p as we did in Tables 1–5. In fact we tried to do that but, for example, if in the 2D Bratu problem with $n_p = 125$ we use the best value of p and β that we found for the case $n_p = 100$, the method diverges. That is an issue when one wants to do experiments like the ones in the previous sections, but it is not a serious problem if one has a practical problem to solve. Therefore, we considered in this comparison only the first problem of each of Tables 1–5. For each of the problems, we made an exhaustive search for the best values of p and β which, based on the values of β reported in [30], comprised testing all combinations of $p \in \{5, 6, \ldots, 10\}$ and $\beta \in \{\beta_a 10^{-\beta_b} \mid \beta_a \in \{0.5, 1\} \text{ and } \beta_b = 1, \ldots, 8\}$, totaling 96 combinations. Figure 4 shows, for each of the five problems considered, the evolution of the residual norm over time obtained with the three best (fastest in terms of CPU

TABLE 4
Performance of NITSOL and Accelerated DF-SANE in the generalized 2D Bratu problem.

n_p	n	NITSOL (Newton-GMRES)					Accelerated DF-SANE			
		$\|F(x_*)\|_2$	#it$_1$	#it$_2$	fcnt	Time	$\|F(x_*)\|_2$	#it	fcnt	Time
100	10,000	7.9e−05	4	212	217	0.03	9.9e−05	174	349	0.16
150	22,500	1.1e−04	4	321	326	0.14	1.5e−04	257	515	0.54
200	40,000	1.6e−04	4	464	469	0.40	2.0e−04	335	671	1.34
250	62,500	1.9e−04	4	655	660	0.98	2.5e−04	405	811	2.63
300	90,000	2.4e−04	4	741	746	1.59	3.0e−04	473	947	4.58
350	122,500	2.8e−04	4	1,054	1,059	3.15	3.4e−04	543	1,087	7.38
400	160,000	3.2e−04	4	1,315	1,320	5.17	4.0e−04	613	1,227	11.31
450	202,500	3.4e−04	4	1,518	1,523	7.54	4.3e−04	684	1,369	17.58
500	250,000	4.1e−04	3	1,607	1,611	10.02	4.9e−04	754	1,509	23.88
550	302,500	4.5e−04	3	2,447	2,451	19.94	5.5e−04	823	1,647	32.47
600	360,000	5.0e−04	3	2,793	2,797	29.30	5.9e−04	893	1,787	41.98
650	422,500	5.2e−04	4	3,502	3,507	45.42	6.3e−04	962	1,925	53.68
700	490,000	5.6e−04	4	3,691	3,696	59.89	7.0e−04	1,030	2,061	68.96
750	562,500	6.2e−04	4	3,702	3,707	72.01	7.4e−04	1,098	2,197	87.03
800	640,000	6.4e−04	5	4,871	4,877	113.52	8.0e−04	1,164	2,329	106.73
850	722,500	6.9e−04	5	4,826	4,832	132.01	8.5e−04	1,229	2,459	124.76
900	810,000	7.2e−04	6	5,965	5,972	187.16	9.0e−04	1,292	2,585	148.10
950	902,500	7.6e−04	6	5,695	5,702	200.12	9.5e−04	1,353	2,707	174.09
1,000	1,000,000	8.0e−04	8	7,386	7,395	292.89	1.0e−03	1,408	2,817	201.93
1,050	1,102,500	8.4e−04	7	6,901	6,909	304.10	1.0e−03	1,465	2,931	241.54
1,100	1,210,000	8.8e−04	9	8,399	8,409	405.89	1.1e−03	1,528	3,057	279.02
1,150	1,322,500	9.2e−04	8	7,977	7,986	423.56	1.1e−03	1,592	3,185	323.85
1,200	1,440,000	1.0e−03	10	9,980	9,991	578.89	1.2e−03	1,656	3,313	368.41
1,250	1,562,500	1.0e−03	10	9,678	9,689	608.95	1.2e−03	1,720	3,441	417.97
1,300	1,690,000	1.3e−03	11	10,960	10,972	747.45	1.3e−03	1,783	3,567	459.53
1,350	1,822,500	1.1e−03	11	10,935	10,947	803.39	1.3e−03	1,846	3,693	517.30
1,400	1,960,000	1.1e−03	13	12,932	12,946	1021.67	1.4e−03	1,909	3,819	585.59
1,450	2,102,500	1.2e−03	13	12,636	12,650	1074.93	1.4e−03	1,971	3,943	648.91
1,500	2,250,000	1.4e−03	14	13,920	13,935	1267.96	1.5e−03	2,032	4,065	718.74

time) combinations of parameters. (Combinations are different for each problem. The values of the parameters in each of the chosen combinations are shown in the figures.) The figures also include the results obtained with Accelerated DF-SANE. The result shown for Accelerated DF-SANE is the one already reported in the previous sections, without any extra tuning of its parameters. (This means that DF-SANE used memory $p = 5$ in the five problems, as well as $h_{\text{small}} = h_{\text{large}} = 0.1$ and $h_{\text{init}} = 1$ in the 3D Bratu problem and $h_{\text{small}} = 10^{-4}$, $h_{\text{large}} = 0.1$, and $h_{\text{init}} = 0.01$ in the other four problems.) The figure shows that Accelerated DF-SANE used approximately 5 to 10 times less CPU time than that used by Anderson mixing in the considered problems. The comparison yields similar results if we use the number of functional evaluations as a measure of performance. Using the number of iterations would not be adequate since the linear algebra cost per iteration depends on p, which varies from method to method and problem to problem (in the case of Anderson mixing).

7. Conclusions. The SSM, which is the most obvious multidimensional generalization of the secant method for solving nonlinear equations, seems to have been introduced by Wolfe in 1959 [64]; see also [2]. This method has been analyzed in classical books and papers [38, 51, 59], where, under suitable conditions, local con-

TABLE 5
Performance of NITSOL and Accelerated DF-SANE in the flow in a porous medium problem.

n_p	n	NITSOL (Newton-GMRES)					Accelerated DF-SANE			
		$\|F(x_*)\|_2$	#it$_1$	#it$_2$	fcnt	Time	$\|F(x_*)\|_2$	#it	fcnt	Time
100	10,000	7.9e−05	11	1,806	1,818	0.22	1.0e−04	684	1,376	0.54
150	22,500	1.2e−04	10	2,666	2,677	0.92	1.5e−04	845	1,698	1.54
200	40,000	1.7e−04	10	3,634	3,645	2.48	2.0e−04	1,255	2,521	4.52
250	62,500	2.4e−04	12	5,817	5,830	6.83	2.5e−04	1,378	2,798	9.51
300	90,000	2.6e−04	13	7,272	7,286	12.84	3.0e−04	1,595	3,212	16.95
350	122,500	3.4e−04	13	7,890	7,904	19.29	3.5e−04	1,634	3,310	19.49
400	160,000	3.9e−04	13	8,655	8,669	28.19	4.0e−04	1,616	3,349	27.38
450	202,500	4.2e−04	15	10,911	10,927	45.61	4.5e−04	1,723	3,642	39.99
500	250,000	4.7e−04	15	10,999	11,015	61.56	5.0e−04	1,858	3,916	52.95
550	302,500	4.7e−04	18	14,065	14,084	103.30	5.5e−04	1,954	3,997	71.96
600	360,000	5.3e−04	18	14,140	14,159	132.62	6.0e−04	2,043	4,145	87.22
650	422,500	6.3e−04	17	13,152	13,170	151.83	6.5e−04	2,171	4,508	107.60
700	490,000	6.4e−04	20	16,127	16,148	229.96	7.0e−04	2,271	4,874	132.76
750	562,500	7.3e−04	18	14,164	14,183	243.82	7.5e−04	2,377	5,263	167.42
800	640,000	7.2e−04	20	16,127	16,148	344.09	8.0e−04	2,502	5,630	204.92
850	722,500	7.6e−04	20	16,147	16,168	396.81	8.5e−04	2,629	6,037	242.25
900	810,000	8.8e−04	20	16,129	16,150	431.85	9.0e−04	2,750	6,398	290.42
950	902,500	9.3e−04	21	17,094	17,116	521.95	9.5e−04	2,909	6,856	346.35
1,000	1,000,000	9.7e−04	20	16,144	16,165	549.77	1.0e−03	2,986	7,227	397.73
1,050	1,102,500	9.9e−04	22	18,098	18,121	679.61	1.0e−03	3,048	7,581	473.76
1,100	1,210,000	1.0e−03	22	18,111	18,134	753.17	1.1e−03	3,125	8,017	546.52
1,150	1,322,500	1.1e−03	22	18,026	18,049	892.35	1.1e−03	3,264	8,448	627.86
1,200	1,440,000	1.2e−03	23	19,006	19,030	1056.51	1.2e−03	3,434	9,051	710.48
1,250	1,562,500	1.1e−03	24	19,983	20,008	1183.49	1.2e−03	3,371	9,061	767.28
1,300	1,690,000	1.3e−03	24	20,010	20,035	1207.67	1.3e−03	3,377	9,249	839.12
1,350	1,822,500	1.3e−03	26	21,969	21,996	1440.00	1.3e−03	3,302	8,892	891.25
1,400	1,960,000	1.3e−03	26	21,947	21,974	1538.56	1.4e−03	3,234	8,477	942.91
1,450	2,102,500	1.4e−03	27	22,938	22,966	2118.16	1.4e−03	3,284	8,566	1021.00
1,500	2,250,000	1.4e−03	27	22,934	22,962	2108.73	1.5e−03	3,314	8,569	1105.85

vergence with R-order equal to the unique solution of $t^{n+1} - t^n - 1 = 0$ was proved. Given the consecutive iterates $x^{k-n}, \ldots, x^{k-1}, x^k$, the SSM computes

$$x^{k+1} = x^k - (s^{k-n}, \ldots, s^{k-1})(y^{k-1}, \ldots, y^{k-1})^{-1} F(x^k). \tag{56}$$

This iteration is well defined if the matrix $(y^{k-n}, \ldots, y^{k-1})$ is nonsingular. Moreover, the good local convergence properties need uniformly linear independence of the increments $s^{k-n}, \ldots, s^{k-1}$. The method has been updated in several ways in order to fix these drawbacks while maintaining its convergence properties. The natural limited memory version of (56) is defined by

$$x^{k+1} = x^k - (s^{k-p}, \ldots, s^{k-1})(y^{k-p}, \ldots, y^{k-1})^{\dagger} F(x^k), \tag{57}$$

where $1 \leq p \ll n$. This formula is inconvenient for solving nonlinear systems because s^k necessarily belongs to the subspace generated by $s^{k-p}, \ldots, s^{k-1}$, which implies that x^{k+j} is in the affine subspace determined by $x^{k-p}, \ldots, x^{k-1}, x^k$ for all j, and so convergence to a solution cannot occur unless the solution belongs to the same affine subspace. This is the reason why, in the present paper, we do not use the method defined by (57). Instead, we compute x^{k+1}_{trial} using the sequential residual approach, we define $s^k = x^{k+1}_{\text{trial}} - x^k$, $y^k = F(x^{k+1}_{\text{trial}}) - F(x^k)$,

$$x^{k+1}_{\text{accel}} = x^{k+1}_{\text{trial}} - (s^{k-p+1}, \ldots, s^{k-1}, s^k)(y^{k-p+1}, \ldots, y^{k-1}, y^k)^{\dagger} F(x^{k+1}_{\text{trial}}),$$

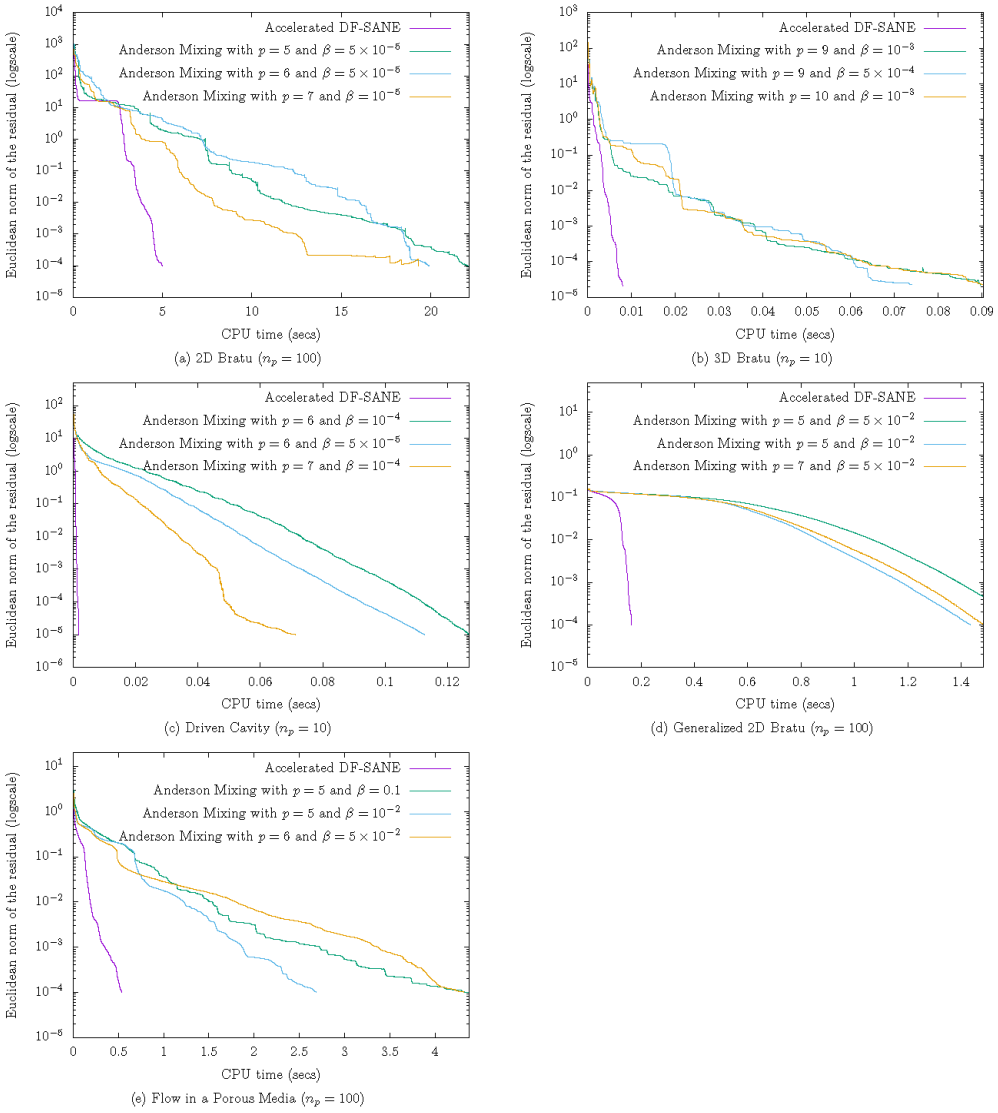

FIG. 4. *Comparison between Accelerated DF-SANE and Anderson mixing.*

and we choose x^{k+1} as the best of the trials x^{k+1}_{trial} and x^{k+1}_{accel}. In this way, we preserve the good properties of the SSM associated with the good global behavior of sequential residual approaches. The freedom in the choice of the residual step favors the employment of preconditioners when they are available.

The most popular methods for solving nonlinear systems of equations in which the application of Newton's method is impossible or extremely expensive are based on the inexact Newton approach with Krylov subspace methods (as GMRES) for solving approximately the Newtonian linear systems at each iteration. These methods have a long tradition and, very likely, they deserve to be the preferred ones by practitioners in the numerical PDE community. Nevertheless, in this paper we showed that, for some very large interesting problems, an approach based on sequential-secant-like

accelerations of a residual method is more efficient than a standard implementation of Newton-GMRES with its default algorithmic parameters. This indicates that those problems possess characteristics that favor the application of the secant paradigm over the inexact Newton one. Of course, the opposite situation probably occurs in many cases. This means, therefore, that efficiency for solving practical problems will be increased if practitioners have easy access to both types of methods.

The numerical experiments presented in the manuscript correspond to the case $k_{\text{mon}} = \infty$, in which random directions play no role and only residual directions are considered. The reason is that in the considered problems (which represent an important class of problems coming from the discretization of PDE equations), the residuals are always gradient related in the sense that they very likely satisfy the angle assumption of Theorem 3.1. Since these directions are generally better than random directions, the practical consequence of this fact is that the performance of the case $k_{\text{mon}} = \infty$ was better than the performance of the case $k_{\text{mon}} < \infty$. However, we think that there are good practical reasons (other than the theoretical reasons already presented) to have incorporated the option $k_{\text{mon}} < \infty$ in Algorithm 2.1. Roughly speaking, DF-SANE finds a solution to $F(x) = 0$ or finds a point x at which $F(x) \perp J(x)^T F(x)$. The algorithm presented in this work either finds a solution to $F(x) = 0$ or finds a point x at which $v^T J(x)^T F(x) = 0$ for all $v \in \mathbb{R}^n$. Since $v^T J(x)^T F(x) = 0$ for all v implies that $J(x)^T F(x) = 0$, it turns out that the point annihilates the gradient of $\frac{1}{2}\|F(x)\|^2$, which is certainly preferable. It is, for example, the best that could be done, by an affordable algorithm (i.e., excluding global optimization techniques), in problems for which a point x such that $F(x) = 0$ does not exist.

Consider the example

$$F(x_1, x_2) = (x_1 x_2, x_1^2 + x_2^2)^T.$$

The unique solution of $F(x_1, x_2) = 0$ is $(0, 0)^T$. However, $F(x_1, x_2)$ is orthogonal to the gradient $J(x_1, x_2)^T F(x_1, x_2)$ of the squared residual $\frac{1}{2}\|F(x)\|^2$ for all $x_1 \in \mathbb{R}$ if $x_2 = 0$. More generally, defining

$$F(x_1, \ldots, x_{n-1}, x_n) = (x_1 \psi(x_n), \ldots, x_{n-1}\psi(x_n), \sum_{i=1}^{n-1} x_i^2 + \psi(x_n)^2)^T,$$

where ψ is an arbitrary smooth function, we have that $F(x)^T J(x)^T F(x) = 0$ for all $x \in \mathbb{R}^n$ such that $\psi(x_n) = 0$. These examples suggest that the residual could define poor search directions in rather simple problems, opening up space for alternatives. In their seminal paper [20] concerning the application of Newton's method to nonlinear complementarity problems using the Fischer–Burmeister function, De Luca, Facchinei, and Kanzow considered the case in which even Newton's directions are poorly connected with the gradient of the objective function and, so, gradient directions should be used in a safeguarding context. The fact that in the present work, gradient directions are not assumed to be computed motivated the random choice.

Future work will include the employment of the new methods to the acceleration of KKT solvers that are necessary in the augmented Lagrangian approach for constrained optimization [5]. Accelerating sequential residual methods with other variants of Anderson's method and embedding Anderson's method in a globally convergent scheme such as Algorithm 2.1 would also be the subject of future research.

Acknowledgment. The authors would like to thank the referees for their careful reading and the many suggestions they contributed to improve the quality of this work.

REFERENCES

[1] D. G. Anderson, *Iterative procedures for nonlinear integral equations*, J. Assoc. Comput. Mach., 12 (1965), pp. 547–560.
[2] J. G. P. Barnes, *An algorithm for solving nonlinear equations based on the secant method*, Comput. J., 8 (1965), pp. 66–72.
[3] J. Barzilai and J. M. Borwein, *Two point step size gradient methods*, IMA J. Numer. Anal., 8 (1988), pp. 141–148.
[4] W. Bian, X. Chen, and C. T. Kelley, *Anderson acceleration for a class of nonsmooth fixed-point problems*, SIAM J. Sci. Comput., 43 (2021), pp. S1–S20.
[5] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*, SIAM, Philadelphia, 2014.
[6] E. G. Birgin, J. M. Martínez, and M. Raydan, *Nonmonotone spectral projected gradient methods on convex sets*, SIAM J. Optim., 10 (2000), pp. 1196–1211.
[7] E. G. Birgin, J. M. Martínez, and M. Raydan, *Algorithm* 813*: SPG – software for convex-constrained optimization*, ACM Trans. Math. Software, 27 (2001), pp. 340–349.
[8] E. G. Birgin, J. M. Martínez, and M. Raydan, *Spectral projected gradient methods*, in Encyclopedia of Optimization, 2nd ed., C. A. Floudas and P. M. Pardalos, eds., Springer, New York, 2009, pp. 3652–3659.
[9] E. G. Birgin, J. M. Martínez, and M. Raydan, *Spectral projected gradient methods: Review and perspectives*, J. Stat. Softw., 60 (2014), https://doi.org/10.18637/jss.v060.i03.
[10] C. Brezinski, *Convergence acceleration during the* 20*th century*, J. Comput. Appl. Math., 122 (2000), pp. 1–21.
[11] C. Brezinski and M. Redivo-Zaglia, *Extrapolation Methods: Theory and Practice*, North-Holland, Amsterdam, 1991.
[12] C. Brezinski, M. Redivo-Zaglia, and Y. Saad, *Shanks sequence transformations and anderson acceleration*, SIAM Rev., 60 (2018), pp. 646–669.
[13] P. N. Brown, H. F. Walker, R. Wasyk, and C. S. Woodward, *On using approximate finite-differences in matrix-free Newton-Krylov methods*, SIAM J. Numer. Anal., 46 (2008), pp. 1892–1911.
[14] O. Burdakov and A. Kamandi, *Multipoint secant and interpolation methods with nonmonotone line search for solving systems of nonlinear equations*, Appl. Math. Comput., 338 (2018), pp. 421–431.
[15] F. P. Cantelli, *Sulla probabilità come limite della frequenza*, Atti Accad. Naz. Lincei, 26 (1917), pp. 39–45.
[16] X. Chen, *Superlinear convergence of smoothing quasi-Newton methods for nonsmooth equations*, J. Comput. Appl. Math., 80 (1997), pp. 105–126.
[17] X. Chen, Z. Nashed, and L. Qi, *Smoothing methods and semismooth methods for nondifferentiable operator equations*, SIAM J. Numer. Anal., 38 (2000), pp. 1200–1216.
[18] X. Chen and C. T. Kelley, *Convergence of the EDIIS algorithm for nonlinear equations*, SIAM J. Sci. Comput., 4 (2019), pp. A365–A379.
[19] M. Chupin, M.-S. Dupuy, G. Legendre, and É. Séré, *Convergence analysis of adaptive DIIS algorithms with application to electronic ground state calculations*, ESAIM Math. Model. Numer. Anal., 55 (2021), pp. 2785–2825.
[20] T. De Luca, F. Facchinei, and C. Kanzow, *A semismooth equation approach to the solution of nonlinear complementarity problems*, Math. Program., 75 (1996), pp. 407–439.
[21] J. Degroote, K.-J. Bathe, and J. Vierendeels, *Performance of a new partitioned procedure versus a monolithic procedure in fluid-structure interaction*, Comput. Struct., 97 (2009), pp. 793–801.
[22] R. S. Dembo, S. C. Eisenstat, and T. S. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.
[23] J. E. Dennis and J. J. Moré, *Quasi-Newton methods: Motivation and theory*, SIAM Rev., 19 (1977), pp. 46–89.
[24] J. E. Dennis, Jr., and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations,* SIAM, Philadelphia, 1996.
[25] S. C. Eisenstat and H. F. Walker, *Globally convergent inexact Newton methods*, SIAM J. Optim., 4 (1994), pp. 393–422.

[26] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), pp. 16–32.
[27] M. É. Borel, *Les probabilités dénombrables et leurs applications arithmetiques*, Rend. Circ. Mat. Palermo, 27 (1909), pp. 247–271.
[28] 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.
[29] V. Eyert, *A comparative study on methods for convergence acceleration of iterative vector sequences*, J. Comput. Phys., 124 (1996), pp. 271–285.
[30] H. Fang and Y. Saad, *Two classes of multisecant methods for nonlinear acceleration*, Numer. Linear Algebra Appl., 16 (2009), pp. 197–221.
[31] L. Frérot, M. Bonnet, J.-F. Molinari, and G. Anciaux, *A Fourier-accelerated volume integral method for elastoplastic contact*, Comput. Methods Appl. Mech. Engrg., 351 (2019), pp. 951–976.
[32] L. Frérot, G. Anciaux, and J.-F. Molinari, *Crack nucleation in the adhesive wear of an elastic-plastic half-space*, J. Mech. Phys. Solids, 145 (2020), 104100.
[33] A. Friedlander, J. M. Martínez, B. Molina, and M. Raydan, *Gradient methods with retards and generalizations*, SIAM J. Numer. Anal., 36 (1998), pp. 275–289.
[34] W. B. Gragg and G. W. Stewart, *A stable variant of the secant method for solving nonlinear equations*, SIAM J. Numer. Anal., 13 (1976), pp. 889–903.
[35] L. Grippo, F. Lampariello, and S. Lucidi, *A nonmonotone line search technique for Newton's method*, SIAM J. Numer. Anal., 23 (1986), pp. 707–716.
[36] R. Haelterman, J. Degroote, D. van Heule, and J. Vierendeels, *The quasi-Newton least squares method: A new and fast secant method analyzed for linear systems*, SIAM J. Numer. Anal., 47 (2009), pp. 2347–2368.
[37] N. Ho, S. D. Olson, and H. F. Walker, *Accelerating the Uzawa algorithm*, SIAM J. Sci. Comput., 39 (2017), pp. 461–476.
[38] J. Jankowska, *Theory of multivariate secant methods*, SIAM J. Numer. Anal., 16 (1979), pp. 547–562.
[39] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations,* SIAM, Philadelphia, 1995.
[40] W. La Cruz, *A spectral algorithm for large-scale systems of nonlinear monotone equations*, Numer. Algorithms, 76 (2017), pp. 1109–1130.
[41] W. La Cruz, J. M. Martínez, and M. Raydan, *Spectral residual method without gradient information for solving large-scale nonlinear systems of equations*, Math. Comput., 75 (2006), pp. 1429–1448.
[42] W. La Cruz and M. Raydan, *Nonmonotone spectral methods for large-scale nonlinear systems*, Optim. Methods Softw., 18 (2003), pp. 583–599.
[43] C. Le Bris, *Computational chemistry from the perspective of numerical analysis*, Acta Numer., 14 (2005), pp. 363–444.
[44] D. H. Li and M. Fukushima, *A derivative-free line search and global convergence of Broyden-like method for nonlinear equations*, Optim. Methods Softw., 13 (2000), pp. 181–201.
[45] F. Lindner, M. Mehl, K. Scheufele, and B. Uekermann, *A comparison of various quasi-Newton schemes for partitioned fluid-structure interaction*, in ECCOMAS Coupled Problems in Science and Engineering, B. A. Schrefler Venice, E. Oñate, and M. Papadrakakis, eds., DIMNE, Barcelona, 2015, pp. 177–488.
[46] J. M. Martínez, *Three new algorithms based on the sequential secant method*, BIT, 19 (1979), pp. 236–243.
[47] E. Meli, B. Morini, M. Porcelli, and C. Sgattoni, *Solving nonlinear systems of equations via spectral residual methods: Stepsize selection and applications*, J. Sci. Comput., 90 (2022).
[48] N. H. Miller and M. Osborne, *Spatial differentiation and price discrimination in the cement industry: Evidence from a structural model*, RAND J. Econ., 45 (2014), pp. 221–247.
[49] L. N. H. G. Oliveira, *private communication*, 2019.
[50] S. S. Oren, *Self-scaling variable metric algorithms without line search for unconstrained minimization*, Math. Comp., 27 (1973), pp. 873–885.
[51] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.
[52] W. Ouyang, Y. Peng, Y. Yao, J. Zhang, and B. Deng, *Anderson acceleration for nonconvex ADMM based on Douglas-Rachford splitting*, Comput. Graph. Forum, 39 (2020), pp. 221–239.

[53] M. Pernice and H. F. Walker, *NITSOL: A Newton iterative solver for nonlinear systems*, SIAM J. Sci. Comput., 19 (1998), pp. 302–318.
[54] P. Pulay, *Convergence acceleration of iterative sequences: The case of SCF iteration*, Chem. Phys. Lett., 73 (1980), pp. 393–398.
[55] M. Raydan, *On the Barzilai and Borwein choice of steplength for the gradient method*, IMA J. Numer. Anal., 13 (1993), pp. 321–326.
[56] M. Raydan, *The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*, SIAM J. Optim., 7 (1997), pp. 26–33.
[57] T. Rohwedder and R. Schneider, *An analysis for the DIIS acceleration method used in quantum chemistry calculations*, J. Math. Chem., 49 (2011), pp. 1889–1914.
[58] M. Rypdal and O. Løvsletten, *Modeling electricity spot prices using mean-reverting multifractal processes*, Phys. A, 392 (2013), pp. 194–207.
[59] H. Schwetlick, *Numerische Lösung Nichtlinearer Gleichungen*, Deutscher Verlag, Berlin, 1978.
[60] D. Scieur, A. d'Aspremont, and F. Bach, *Regularized nonlinear acceleration*, Math. Program., 179 (2020), pp. 47–83.
[61] A. Toth and C. T. Kelley, *Convergence analysis for Anderson acceleration*, SIAM J. Numer. Anal., 53 (2015), pp. 805–819.
[62] H. F. Walker and P. Ni, *Anderson acceleration for fixed-point iterations*, SIAM J. Numer. Anal., 49 (2011), pp. 1715–1735.
[63] H. F. Walker, C. S. Woodward, and U. M. Yang, *An accelerated fixed-point iteration for solution of variably saturated flow*, in Proceedings of the 18th International Conference on Water Resources, J. Carrera, ed., CIMNE, Barcelona, 2010.
[64] P. Wolfe, *The secant method for simultaneous nonlinear equations*, Commun. ACM, 2 (1959), pp. 12–13.
[65] J. Zhang, B. O'Donoghue, and S. Boyd, *Globally convergent type-*I *Anderson acceleration for nonsmooth fixed-point iterations*, SIAM J. Optim., 30 (2020), pp. 3170–3197.