Accelerated derivative-free nonlinear least-squares applied to the estimation of Manning coefficients*

E. G. Birgin† J. M. Martínez‡

April 6, 2021§

Abstract

A general framework for solving nonlinear least squares problems without the employment of derivatives is proposed in the present paper together with a new general global convergence theory. With the aim to cope with the case in which the number of variables is big (for the standards of derivative-free optimization), two dimension-reduction procedures are introduced. One of them is based on iterative subspace minimization and the other one is based on spline interpolation with variable nodes. Each iteration based on those procedures is followed by an acceleration step inspired in the Sequential Secant Method. The practical motivation for this work is the estimation of parameters in Hydraulic models applied to dam breaking problems. Numerical examples of the application of the new method to those problems are given.

Key words: Nonlinear least-squares, derivative-free methods, acceleration, Manning coefficients.

1 Introduction

Many statistical learning problems require fitting models to large data sets. Frequently, the number of unknown parameters is not small. Moreover, for different reasons, derivatives of the functions that define the model may not be available, and the sum of squares of residuals is a natural function to be minimized. These considerations lead to the problem

$$\text{Minimize } \frac{1}{2}||F(x)||_2^2 \text{ subject to } x \in \Omega \subseteq \mathbb{R}^n, \tag{1}$$

where $F : \Omega \to \mathbb{R}^m$.

Let us define $f(x) = \frac{1}{2}\|F(x)\|_2^2$. For obtaining a quadratic approximation of $f(x)$ with the property of being exact if $f(x)$ is quadratic, $1 + n + n(n+1)/2$ evaluations of $f(x)$ are needed. However, if the structure $\frac{1}{2}\|F(x)\|_2^2$ of $f(x)$ is used, the same property can be obtained using only $n + 1$ evaluations of $F(x)$; see [32, 34, 48]. Considering that evaluating $f(x)$ and $F(x)$ has the same cost, this seems to be a strong argument to take advantage of the sum-of-squares structure of $f(x)$, especially if derivatives are not available.

*This work 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, Rua do Matão, 1010, Cidade Universitária, 05508-090, São Paulo, SP, Brazil. e-mail: 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. e-mail: martinez@ime.unicamp.br

§Revision made on August 16, 2021 and December 3, 2021.

Ralston and Jennrich [35] introduced a purely local method that, at each iteration, minimizes the norm of the linear model that interpolates $n+1$ consecutive residuals, providing the first generalization of the Sequential Secant Method [3, 44] to nonlinear least squares. Zhang, Conn, and Scheinberg [48] employed different quadratic models for each component of the residual function in order to define conveniently structured trust-region subproblems. Therefore, as in [31, 32, 33, 34], at least $2n+1$ residual evaluations are computed per iteration. The use of quadratic models allow these authors to prove not only global convergence, but also local quadratic convergence under suitable assumptions [47]. The idea of interpolating a different quadratic for each component of the residual has also been exploited in the POINDERS software [43] with trust-region strategies for obtaining global convergence. Cartis and Roberts [10] introduced a derivative-free Gauss-Newton method for solving nonlinear least squares problems. At each iteration of their method, $n+1$ residuals are used to interpolate a linear model of $F(x)$. The norm of the linear model is approximately minimized over successive trust regions until sufficient decrease of the sum of squares is obtained. The points used for interpolation are updated in order to preserve well-conditioning. With this framework global convergence and complexity results are proved. All mentioned methods suffer from a high linear algebra cost per iteration related to construct the model and find a model's solution; so a natural idea, explored in the present work, is to apply dimensionality-reduction techniques. While developing this work, we became aware of a work of Cartis and Roberts [11] in which this idea is explored. In [11], a method that performs successive minimizations within random subspaces, employing the model-based framework based on the Gauss-Newton method introduced in [10], is introduced.

The present work is motivated by the objectives of CRIAB, a research group of the State of São Paulo, in Brazil, aimed at investigating, understanding, and mitigating the consequences of technological disasters caused by the rupture of dams, which, unfortunately, are occurring both in Brazil and in the rest of the world with increasing frequency. Different hydraulic models are used for such objectives. All of them tend to work correctly if the parameters that determine their behavior (along with initial and boundary conditions) are correctly estimated. The goal of the present work is to offer to the Hydraulic Engineering community an efficient methodology for parameter estimation, exemplified, in this case, by Manning's coefficients. As it is well known, such coefficients determine the level of non-linearity in the evolution of the flow; and their correct estimation can lead to better forecasts and, consequently, better chances of mitigating consequences and allocating resources in different regions affected by the eventual flood. The algorithmic approach presented here has a strong mathematical basis and the experiments carried out indicate its efficiency for forecasting real situations. We have great expectations that the dissemination of these techniques in the Hydraulic Engineering community will have significant effects in practical terms, both in economic and environmental and sanitary terms.

A derivative-free method for large-scale least-squares problems is introduced in the present work. Mathematical models for the estimation of parameters in one-dimensional models that simulate water or mud flow in natural channels consist of partial differential equations with boundary conditions that simulate flood intensity. The initial conditions for this type of models are, in general, well known, but the parameters reflecting density, friction, obstacles, or terrain features must be estimated from data. We are particularly interested in Manning coefficients. Manning's coefficients play a crucial role in the correct modeling of mud or water flow in a natural channel influenced by a flood. In principle, in perfectly straight channels with constant cross-sectional area, these coefficients account for velocity reductions due to friction with the walls or viscosity of the fluid. In real situations, in which the channel is not straight and the cross-sectional area is not constant, Manning's coefficients absorb the information due to these "irregularities", which, in fact, detract the theoretical model from the real situation. The realistic simulation of a natural channel cannot rely on theoretical estimates of Manning's coefficients based on physical considerations linked to ideal situations. Necessarily, such coefficients must be estimated on the basis of (much or little) available data. This is the exercise we

propose in the present work, for which we use an ideal situation that allows us to infer the usefulness of the introduced methods in more realistic situations. Incidentally, data collection in real cases was dramatically interrupted in 2020 by the outbreak of the pandemic we are still suffering from. The use of programs whose source code is not available is frequent in this type of research. For this reason, we are interested in investigating the behavior of derivative-free methods to estimate parameters of the models used. The introduced method combines dimensionality-reduction techniques [42, 45] and acceleration steps based on the sequential secant approach [3, 44].

Acceleration schemes, by means of which, given an iterate x^k and its predecessors, one obtains a possible (accelerated) better approximation to the solution may be applied to any of the algorithms mentioned in the previous paragraph. Let us provide a rough description of the sequential secant idea applied to nonlinear least squares problems. Assume that $p \in \{1, 2, \dots n\}$ is given and $x^0, x^{-1}, \dots, x^{-p} \in \mathbb{R}^n$ are arbitrary. Given $k = 0, 1, 2, \dots$, we define

$$s^{k-1} = x^k - x^{k-1}, \dots, s^{k-p} = x^{k-p+1} - x^{k-p},$$

$$y^{k-1} = F(x^k) - F(x^{k-1}), \dots, y^{k-p} = F(x^{k-p+1}) - F(x^{k-p}),$$

and

$$S_k = (s^{k-1}, \dots, s^{k-p}) \text{ and } Y_k = (y^{k-1}, \dots, y^{k-p}).$$

The Sequential Secant Method for nonlinear least-squares is defined by

$$x^{k+1} = x^k - S_k Y_k^\dagger F(x^k), \tag{2}$$

where $Y_k^\dagger$ denotes the Moore-Penrose pseudo-inverse of Y_k. Its main drawback is that, according to (2), $x^{k+1} - x^k$ always lies in the subspace generated by $\{s^{k-1}, \dots, s^{k-p}\}$. Therefore, all the iterates lie in the affine subspace that passes through x^0 and is spanned by $\{s^{-1}, \dots, s^{-p}\}$. This is not a serious inconvenient if $p = n$ and the increments $s^{k-1}, \dots, s^{k-p}$ remain linearly independent. However, even when $p = n$, the vectors $s^{k-1}, \dots, s^{k-p}$ may become linearly dependent and, consequently, all the iterates x^{k+j} would be condemned to lie in a fixed affine subspace of dimension strictly smaller than n. For these reasons, the pure Sequential Secant Method is not appropriate for solving nonlinear least-squares problems when n is large and, thus, it is required to maintain p reasonable small.

Note, however, that, when $m = n$, under suitable assumptions, the method defined by (2) has Q-superlinearly local converge to a solution of $F(x) = 0$, and its R-rate of convergence is the positive root of $t^{n+1} - t^n - 1 = 0$ [29]. When $m = n$, the problem consists of solving the nonlinear system $F(x) = 0$. This case has been extensively considered in [4]. The drawback pointed out above was overcome in [4] taking auxiliary residual-related directions. The idea of using residuals as search directions for solving nonlinear systems of equations have been introduced and exploited in [23, 24, 27, 40] and analyzed from the point of view of complexity in [12]. Unfortunately, in general nonlinear least-squares problems, in which $m \neq n$, it is not possible to use residuals as search directions, since residuals are in $\mathbb{R}^m$ and search directions are in $\mathbb{R}^n$. Therefore, in the present paper, we suggest different alternatives for choosing the first trial point without residual information at each iteration. This is the place where the dimensionality-reduction techniques place their role – trial points are computed by minimizing the least-squares function in a reduced space. Two alternatives are considered. In one of them, minimizations within small random affine-subspaces are performed. On the other one, the reduced problem has as variables nodes and values of a linear spline from which the values of the original variables are obtained. After the computation of a suitable trial point, we try an acceleration step using sequential secant ideas.

It is worth mentioning that sequential secant acceleration is closely connected with Anderson acceleration [1, 7, 8, 9, 21, 28, 36, 41] and quasi-Newton acceleration [7, 16, 20, 26]. Moreover, the

sequential secant algorithm is a particular case of a family of secant methods described in [29] and [22], whereas related multipoint secant methods for solving nonlinear systems and minimization have been introduced in [5, 6, 18, 19, 38, 39] and others.

This paper is organized as follows. In Section 2, we introduce a general scheme that applies to derivative-free optimization (not only nonlinear least-squares) and has the proposed algorithm for nonlinear least squares as particular case. Global convergence results for the general scheme are included in this section. In Section 3, we define the specific algorithm that we use for derivative-free nonlinear least-problems. In Section 4, we present the problem of estimating Manning coefficients and report numerical experiments. Conclusions and lines for future research are stated in Section 5.

Notation. The symbol $\|\cdot\|$ will denote an arbitrary norm.

2 General optimization framework

In this section, we consider the general problem

$$\text{Minimize } f(x) \text{ subject to } x \in \Omega, \tag{3}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is arbitrary and $\Omega \subset \mathbb{R}^n$ is closed and convex, of which problem (1) is a particular case. In principle, no assumption on the objective function f is made in order to define Algorithm 2.1. However, well-definiteness and global convergence results that follow require continuity (Lemmas 2.1 and 2.2) and continuous differentiability (Lemma 2.3 and 2.4).

Algorithm 2.1 presented below applies to the solution of (3). It resembles the classical Frank-Wolfe algorithm [17], conceived for the minimization of convex objective functions, because, at each iteration, it minimizes a linear function subject to the true constraints of the problem and an additional constraints that guarantees that the problem is solvable. The linear approximation considered at the subproblem of iteration k is given by $\langle v^k, d\rangle$ with an arbitrary v^k such that $\|v^k\| = 1$. (The requirement $\|v^k\| = 1$ is arbitrary and it could be replaced by $\|v^k\| = c$ for any constant $c > 0$.) In its more general form, this subproblem hardly approximates f at all. Thus, the goal of the subproblem is, merely, to find a feasible trial point $x^k + d^k$ in the intersection of the convex domain and a fixed trust region defined by Δ. The practical role of Δ, other than making the subproblem solvable, is to prevent the ocurrence of unreasonably large steps d^k, whose appearance could cause the necessity of many evaluations of the objective function to satisfy the desired descent criterion with a backtracking procedure along the direction d^k. The descent condition is nonmonotone and it can be satisfied by a trial point of the form $x^{\text{trial}} = x^k + \alpha d^k$ provided that $\alpha > 0$ be sufficiently small. The iteration ends defining $x^{k+1} = x^{\text{trial}}$ or any x^{k+1} such that $f(x^{k+1}) \leq f(x^{\text{trial}})$. Despite its generality and theoretical properties, the efficiency of practical versions of the algorithm will depend on specific choices of v^k and ad-hoc strategies for the definition of x^{k+1} that will be shown in the next section.

Algorithm 2.1. Let $f_{\text{target}} \in \mathbb{R}$, $\Delta > 0$, $\gamma \in (0, 1)$, a sequence $\{\eta_k\}$ of positive numbers such that

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

and the initial guess $x^0 \in \Omega$ be given. Set $k \leftarrow 0$.

Step 1. If $f(x^k) \leq f_{\text{target}}$, then terminate the execution of the algorithm.

Step 2. Choose $v^k \in \mathbb{R}^n$ such that $\|v^k\| = 1$.

Step 3. Compute d^k as a solution to the suproblem given by

$$\text{Minimize}\langle v^k, d\rangle \text{ subject to } \|d\| \leq \Delta \text{ and } x^k + d \in \Omega. \tag{5}$$

Step 4. Set $\alpha \leftarrow 1$.

Step 5. Set $x^{\text{trial}} \leftarrow x^k + \alpha d^k$.

Step 6. Test the descent condition

$$f(x^{\text{trial}}) \leq f(x^k) + \eta_k - \gamma\alpha^2 \left[f(x^k) - f_{\text{target}}\right]. \tag{6}$$

Step 7. If (6) holds define $\alpha_k = \alpha$, compute $x^{k+1} \in \Omega$ such that

$$f(x^{k+1}) \leq f(x^{\text{trial}}), \tag{7}$$

set $k \leftarrow k + 1$, and go to Step 1. Otherwise, update $\alpha \leftarrow \alpha/2$ and go to Step 5.

Remark. In Step 7, the new iterate x^{k+1} can be chosen as the trial point x^{trial} that satisfies (6). However, x^{k+1} may also be chosen as any point that satisfies (7). On the one hand, this freedom does not affect the algorithm's theoretical results. On the other hand, it opens the possibility of defining accelerations of the main procedure that can positively affect the practical behavior of the algorithm.

In Lemma 2.1, we prove that Algorithm 2.1 is well defined, so that given an iterate x^k the next iterate x^{k+1} necessarily exists. In Lemma 2.2, we prove that either $f(x^k)$ approximates a target value f_{target} up to arbitrary precision or the step α_k tends to zero.

Lemma 2.1 *Assume that f is continuous, $x^k \in \mathbb{R}^n$ is an arbitrary iterate of Algorithm 2.1, and $f(x^k) > f_{\text{target}}$. Then, x^{trial} and x^{k+1} satisfying (6) and (7) are well defined.*

Proof: The thesis follows from the continuity of f using that $\eta_k > 0$ and that the successive trials for α tend to zero. □

Lemma 2.2 *Assume that f is continuous and, for all $k \in \mathbb{N}$, we have that $f(x^k) > f_{\text{target}}$. Then,*

$$\lim_{k\to\infty} \alpha_k^2 \left[f(x^k) - f_{\text{target}}\right] = 0. \tag{8}$$

Morever, at least one of the following two possibilities takes place:

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

or there exists an infinite subset of indices $K_1 \subset \mathbb{N}$ such that

$$\lim_{k\in K_1} f(x^k) = f_{\text{target}}. \tag{10}$$

Proof: By Lemma 2.1 and the hypothesis, the algorithm generates an infinite sequence $\{x^k\}$ such that $\{f(x^k)\}$ is bounded below. Assume that (8) is not true. Then, there exists $c > 0$ such that

$$\alpha_k^2[f(x^k) - f_{\text{target}}] \geq c \tag{11}$$

for infinitely many indices $k \in K_2$. By the convergence of $\sum_{k=0}^{\infty} \eta_k$, there exists $k_1 \in \mathbb{N}$ such that

$$\eta_k < \gamma c/2$$

for all $k \geq k_1$. Then, by (6), (7), and (11), for all $k \in K_2$ such that $k \geq k_1$,

$$f(x^{k+1}) \leq f(x^k) + \gamma c/2 - \gamma c = f(x^k) - \gamma c/2. \tag{12}$$

Let $k_2 \geq k_1$ such that

$$\sum_{k=k_2}^{\infty} \eta_k < \gamma c/4.$$

Then, by (6) and (7), for all $k \in \mathbb{N}$, $k > k_2$ we have that

$$\begin{aligned} f(x^k) - f(x^{k_2}) &= [f(x^k) - f(x^{k-1})] + [f(x^{k-1}) - f(x^{k-2})] + \cdots + [f(x^{k_2+1}) - f(x^{k_2})] \\ &\leq \eta_{k-1} + \eta_{k-2} + \ldots + \eta_{k_2} < \gamma c/4. \end{aligned} \tag{13}$$

Thus, between two consecutive terms (not smaller than k_2) of the sequence K_2, by (13), f increases at most $\gamma c/4$; but, by (12), decreases at least $\gamma c/2$. This implies that $\lim_{k\to\infty} f(x^k) = -\infty$, which contradicts the fact that $\{f(x^k)\}$ is bounded below. Therefore, (8) is proved.

Now, if (9) does not hold, there exists an infinite set of indices K_1 such that α_k is bounded away from zero. By (8), we have that (10) must take place. □

By Lemmas 2.1 and 2.2, there are three possibilities for the sequence generated by Algorithm 2.1: **(i)** The sequence terminates at some x^k where $f(x^k) \leq f_{\text{target}}$; **(ii)** The sequence terminates at some x^k where $f(x^k) \leq f_{\text{target}} + \varepsilon_f$ for a given tolerance $\varepsilon_f > 0$; and **(iii)** The sequence $\{\alpha_k\}$ tends to zero. Possibilities (i) and (ii) are symptoms of success of the algorithm. Possibility (iii) cannot be discarded since, given $\varepsilon_f > 0$, $f(x)$ may be bigger than $f_{\text{target}} + \varepsilon_f$ for every $x \in \Omega$. Therefore, the implications of $\alpha_k \to 0$ need to be analyzed.

In Lemma 2.3 it is proved that, in the case that $f(x^k)$ is bounded away from f_{target}, if the sequence of iterates x^k admits a limit point x^*, then the set of search directions generated in Step 3 is bounded and every limit point of this sequence of directions *is not* a descent direction of $f(x)$ emanating from x^*. This fact has a positive consequence in terms of optimality if the choice of v^k in Step 2 is, in some sense, gradient-related. (Note that v^k is the gradient of the linear function that, implicitly, is taken as a linear approximation of f at Step 3.) The required gradient-relatedness of v^k is given by assumption (19) in Lemma 2.4, whose plausibility is discused after the lemma. The consequence, proved in Lemma 2.4, is that at every limit point of $\{x^k\}$, every feasible direction d is not a descent direction. The final theorem in this section is a consequence of this fact.

Lemma 2.3 *Assume that f admits continuous derivatives for all x in an open set that contains Ω and $\{x^k\}$ is generated by Algorithm 2.1. Assume that $\{f(x^k) - f_{\text{target}}\}$ is bounded away from zero, $x_* \in \mathbb{R}^n$, and there exists $K_1 \subset \mathbb{N}$ such that $\lim_{k\in K_1} x^k = x_*$. Then, the sequence $\{d^k\}_{k\in K_1}$ admits at least one limit point and, for every limit point d of $\{d^k\}_{k\in K_1}$, we have that*

$$\langle \nabla f(x_*), d \rangle \geq 0. \tag{14}$$

Proof: By Lemma 2.2,

$$\lim_{k\to\infty} \alpha_k = 0. \tag{15}$$

Since the first trial value for α_k at each iteration is 1, (15) implies that

$$\lim_{k\to\infty} \alpha_{k,+} = 0,$$

and, for all k large enough,

$$f(x^k + \alpha_{k,+} d^k) > f(x^k) + \eta_k - \gamma \alpha_{k,+}^2 \left[f(x^k) - f_{\text{target}} \right].$$

So, since $\eta_k > 0$,

$$\frac{f(x^k + \alpha_{k,+} d^k) - f(x^k)}{\alpha_{k,+}} > -\gamma \alpha_{k,+} \left[f(x^k) - f_{\text{target}} \right]$$

for all k large enough. Thus, by the Mean Value Theorem, there exists $\xi_{k,+} \in [0, \alpha_{k,+}]$ such that

$$\langle \nabla f(x^k + \xi_{k,+} d^k), d^k \rangle > -\gamma \alpha_{k,+} \left[f(x^k) - f_{\text{target}} \right] \tag{16}$$

for all k large enough.

Since $\|d^k\| \leq \Delta$ for all k, we have that $\{d^k\}_{k \in K_1}$ admits at least one limit point. Let d be an arbitrary limit point of $\{d^k\}_{k \in K_1}$ and let $K_2 \subseteq K_1$ such that

$$\lim_{k \in K_2} d^k = d \tag{17}$$

and $\|d\| \leq \Delta$. By continuity, since $\lim_{k \in K_2} x^k = x_*$ we have that

$$\lim_{k \in K_2} f(x^k) = f(x_*). \tag{18}$$

Then, taking limits for $k \in K_2$ in both sides of (16), by (15), (17), (18), and the fact that (15) implies $\lim_{k \in K_2} \xi_{k,+} = 0$, we get

$$\langle \nabla f(x_*), d \rangle \geq 0$$

as we wanted to prove. □

Lemma 2.4 *Assume that f admits continuous derivatives for all x in an open set that contains Ω and $\{x^k\}$ is generated by Algorithm 2.1. Assume that $\{f(x^k) - f_{\text{target}}\}$ is bounded away from zero, $x_* \in \mathbb{R}^n$, and there exists $K_1 \subset \mathbb{N}$ such that $\lim_{k \in K_1} x^k = x_*$ and*

$$\lim_{k \in K_1} \left\| v^k - \frac{\nabla f(x^k)}{\|\nabla f(x^k)\|} \right\| = 0. \tag{19}$$

Then, for all $d \in \mathbb{R}^n$ such that $\|d\| \leq \Delta$ and $x_* + d \in \Omega$ we have that

$$\langle \nabla f(x_*), d \rangle \geq 0.$$

Proof: If $\nabla f(x_*) = 0$, we are done; so we assume $\nabla f(x_*) \neq 0$ from now on. By Lemma 2.3, there exists $K_2 \subseteq K_1$ and $\bar{d} \in \mathbb{R}^n$ such that

$$\lim_{k \in K_2} d^k = \bar{d} \tag{20}$$

and

$$\langle \nabla f(x_*), \bar{d} \rangle \geq 0; \tag{21}$$

and, since $\nabla f(x_*) \neq 0$, (21) implies

$$\left\langle \frac{\nabla f(x_*)}{\|\nabla f(x_*)\|}, \bar{d} \right\rangle \geq 0. \tag{22}$$

Given $\varepsilon > 0$, by (19), (20), and the continuity of ∇f, (22) implies that

$$\langle v^k, d^k \rangle \geq -\varepsilon \tag{23}$$

for all $k \in K_2$ large enough. Since, by the definition of Algorithm 2.1, d^k is a solution to (5), (23) implies that

$$\langle v^k, d \rangle \geq -\varepsilon \tag{24}$$

for all $d \in \mathbb{R}^n$ such that $x^k + d \in \Omega$ and $\|d\| \leq \Delta$ and all $k \in K_2$ large enough.

Consider the problem

$$\text{Minimize } \left\langle \frac{\nabla f(x_*)}{\|\nabla f(x_*)\|}, d \right\rangle \text{ subject to } \|d\| \leq \Delta \text{ and } x_* + d \in \Omega \tag{25}$$

that, by compactness, admits a solution d_*; and suppose, by contradiction, that

$$\left\langle \frac{\nabla f(x_*)}{\|\nabla f(x_*)\|}, d_* \right\rangle = -c < 0. \tag{26}$$

Therefore,

$$\left\langle \frac{\nabla f(x_*)}{\|\nabla f(x_*)\|}, x_* + d_* - x_* \right\rangle = -c < 0.$$

This implies, by (19), that

$$\left\langle v^k, x_* + d_* - x^k \right\rangle \leq -c/2 < 0 \tag{27}$$

for $k \in K_2$ large enough. Let us write $\tilde{d}^k = x_* + d_* - x^k$. Since $x_* + d_* \in \Omega$, we have that $x^k + \tilde{d}^k \in \Omega$. If, for some $k \in K_2$ large enough, we have that $\|\tilde{d}^k\| \leq \Delta$, taking $\varepsilon < -c/2$, we get a contradiction between (27) and (24). This contradiction comes from the assumption (26), which, as a consequence, is false, completing the proof.

We now consider the case in which $\|\tilde{d}^k\| > \Delta$ for all $k \in K_2$ large enough. Since

$$\|\tilde{d}^k\| = \|x_* + d_* - x^k\| \leq \|d_*\| + \|x_* - x^k\| \leq \Delta + \|x_* - x^k\|,$$

defining

$$\hat{d}^k = \frac{\tilde{d}^k}{1 + \|x^k - x_*\|/\Delta}, \tag{28}$$

we have that $\|\hat{d}^k\| \leq \Delta$ and, by the convexity of Ω, $x^k + \hat{d}^k \in \Omega$. Moreover, by (27), since $\lim_{k \in K_2} x^k = x_*$,

$$\left\langle v^k, \hat{d}^k \right\rangle \leq -c/4 < 0 \tag{29}$$

for $k \in K_2$ large enough. Taking $\varepsilon < -c/4$, we get the contradiction between (29) and (24) and the proof is complete. □

Remark. Let us show that Assumption (19) is plausible. With this purpose, assume that it does not hold. Then, there exists $\varepsilon > 0$ such that for all $k \in \mathbb{N}$,

$$\left\| v^k - \frac{\nabla f(x^k)}{\|\nabla f(x^k)\|} \right\| > \varepsilon.$$

Clearly, if we choose randomly the vectors v^k in the unitary sphere, the probability of this event is zero. Therefore, assumption (19) holds with probability 1. In a deterministic setting, consider the

often used assumption that states that $\{v^k\}$ is dense on the unitary sphere of $\mathbb{R}^n$; see, for example, [2]. If $\|\nabla f(x^k)\|$ does not tend to zero, then there exists a subsequence such that $\|\nabla f(x^k)\|$ is bounded away from zero. Then, the sequence $\{\nabla f(x^k)/\|\nabla f(x^k)\|\}$ admits a limit point w such that $\|w\| = 1$. By the density of $\{v^k\}$, there exists a subsequence of $\{v^k\}$ that converges to w. By continuity of $\nabla f(x)$, this implies that Assumption (19) holds for that subsequence. It is worth noting, the authors make no claims about the practical implications of this fact.

Theorem 2.1 *Assume that f admits continuous derivatives for all x in an open set that contains Ω and $\{x^k\}$ is generated by Algorithm 2.1. Assume that the level set defined by $f(x^0) + \eta$ is bounded, where $\eta = \sum_{k=0}^{\infty} \eta_k$, and there exists an infinite sequence of indices K_1 such that (19) holds. Then, given $\varepsilon > 0$, either exists an iterate x^k such that $f(x^k) \leq f_{\text{target}} + \varepsilon$ or there exists a limit point x_* of $\{x^k\}$ such that $\langle \nabla f(x_*), d\rangle \geq 0$ for all d such that $x_* + d \in \Omega$.*

Proof: Since the level set defined by $f(x^0)+\eta$ is bounded, the sequence $\{x^k\}_{k\in K_1}$ admits a limit point. Therefore, the thesis follows from Lemma 2.4. □

3 Practical algorithm for nonlinear least-squares

In this section, we are interested in the application of Algorithm 2.1 to large scale nonlinear least squares problems of the form

$$\underset{x\in\mathbb{R}^n}{\text{Minimize}}\ \frac{1}{2}\|F(x)\|_2^2, \tag{30}$$

where $F : \mathbb{R}^n \to \mathbb{R}^m$ and $\|\cdot\|_2$ is the Euclidean norm. Consequently, we define

$$f(x) = \frac{1}{2}\|F(x)\|_2^2. \tag{31}$$

The proposed algorithm for solving (30) is a particular case of Algorithm 2.1 for the case $\Omega = \mathbb{R}^n$, but it includes two additional features: minimizations in reduced spaces and acceleration steps. Two different ways of minimizing within reduced spaces are described in Sections 3.1 and 3.2; while the acceleration strategy is described in Section 3.3. Only unconstrained problems are considered in this section because, up to the present, we are unable to recommend efficient acceleration methods in constrained cases.

Algorithm 3.1. Let $f_{\text{target}} \in \mathbb{R}$, $\Delta > 0$, $\gamma \in (0, 1)$, a sequence $\{\eta_k\}$ of positive numbers such that

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

and the initial guess $x^0 \in \mathbb{R}^n$ be given. Set $k \leftarrow 0$.

Step 1. If $f(x^k) \leq f_{\text{target}}$, then terminate the execution of the algorithm.

Step 2. Compute $x^{\text{trial}} \in \mathbb{R}^n$ by means of a Reduction Algorithm.

Step 3. Test the descent condition

$$f(x^{\text{trial}}) \leq f(x^k) + \eta_k - \gamma\left[f(x^k) - f_{\text{target}}\right]. \tag{32}$$

If $x^{\text{trial}} \neq x^k$ and (32) holds, set $d^k = x^{\text{trial}} - x^k$, $\alpha_k = 1$, $v^k = d^k/\|d^k\|$, and go to Step 9.

Step 4. Choose $v^k \in \mathbb{R}^n$ such that $\|v^k\| = 1$.

Step 5. Compute d^k as a solution to the subproblem given by

$$\text{Minimize} \langle v^k, d \rangle \text{ subject to } \|d\| \leq \Delta.$$

Step 6. Set $\alpha \leftarrow 1$.

Step 7. Set $x^{\text{trial}} \leftarrow x^k + \alpha d^k$.

Step 8. Test the descent condition

$$f(x^{\text{trial}}) \leq f(x^k) + \eta_k - \gamma \alpha^2 \left[f(x^k) - f_{\text{target}} \right]. \tag{33}$$

If (33) does not hold, update $\alpha \leftarrow \alpha/2$ and go to Step 7. Otherwise, set $\alpha_k = \alpha$.

Step 9. Compute, by means of the Acceleration Algorithm, $x^{k+1} \in \mathbb{R}^n$ such that

$$f(x^{k+1}) \leq f(x^{\text{trial}}).$$

Set $k \leftarrow k + 1$ and go to Step 1.

Lemmas 2.1, 2.2, 2.3, 2.4, and Theorem 2.1 hold for Algorithm 3.1 exactly in the same way as they do for Algorithm 2.1. In order to complete the definition of Algorithm 3.1, we now introduce two possible Reduction Algorithm and an Acceleration Algorithm. Both Reduction algorithms employ BOBYQA [34] for minimizing $f(x)$ over manifolds of moderate dimension.

3.1 Affine-subspaces-based Reduction Algorithm

In this Reduction Algorithm the manifold over which we minimize $f(x)$ at each iteration is an affine subspace. At iteration k, we consider an affine transformation $\mathcal{T}_k : \mathbb{R}^{n_{\text{red}}} \to \mathbb{R}^n$, with $n_{\text{red}} \leq n$, given by $\mathcal{T}_k(d) := x^k + M_k d$, where $M_k \in \mathbb{R}^{n \times n_{\text{red}}}$ is a matrix with random (with uniform distribution) elements $m_{ij} \in [-1, 1]$. So the problem to be solved at iteration k is given by

$$\underset{d \in \mathbb{R}^{n_{\text{red}}}}{\text{Minimize}} \frac{1}{2} \|F(\mathcal{T}_k(d))\|_2^2. \tag{34}$$

The natural initial guess for this problem is given by $d = 0$. Since (34) is expected to be small, its (approximate) solution x^{trial} may be computed with any derivative-free method capable of dealing with small unconstrained problems. The minimization on small-dimensional subspaces has been used in [42, 45]. Moreover, in the context of derivative-free optimization, it has been recently employed in [11].

3.2 Linear-interpolation-based Reduction Algorithm

The Reduction Algorithm described in this section is suitable for problems in which the variables $x_1, \ldots, x_n$ exhibit some continuity with respect to $i \in \{1, \ldots n\}$. For example, these variables may represent discrete realizations of a continuous function, discretized by the indices i.

At iteration k, we consider a linear-spline-based transformation $\mathcal{S}_k : \mathbb{R}^{n_{\text{red}}} \to \mathbb{R}^n$, with $n_{\text{red}} \leq n$, where $n_{\text{red}} = 2\kappa + 2$ for some $\kappa \geq 0$. Variables of the reduced model are the κ knots $p_1, \ldots, p_\kappa$, and $\kappa + 2$ values $v_0, v_1, \ldots, v_\kappa, v_{\kappa+1}$, with $0 \leq p_j \leq 1$ for $j = 1, \ldots, \kappa$. We now describe how $\mathcal{S}_k$ transforms $(v_0, \ldots, v_{\kappa+1}, p_1, \ldots, p_\kappa) \in \mathbb{R}^{n_{\text{red}}}$ into $(x_1, \ldots, x_n) \in \mathbb{R}^n$. Define two additional knots $p_0 = 0$ and

$p_{\kappa+1} = 1$. Roughly speaking, each knot p_j is associated with the value v_j and $x_1, \ldots, x_n$ are computed by linear interpolation. The detail that must be taken into considerations is that the *variable* knots $p_j \in [0,1]$, for $j = 1, \ldots, \kappa$, *do not* satisfy $p_0 < p_1 < \cdots < p_\kappa < p_{\kappa+1}$ and they can even be such that $p_{j_1} = p_{j_2} = \ldots$ for $j_1 \neq j_2 \neq \ldots$ If $p_{j_1} = p_{j_2} = \ldots$ for $j_1 \neq j_2 \neq \ldots$, then redefine $v_{j_1}, v_{j_2}, \ldots$ as their average. Let $\bar{p}_0 < \cdots < \bar{p}_{\bar{\kappa}+1}$ (with $\bar{\kappa} \leq \kappa$) be a permutation of $p_0, \ldots, p_{\kappa+1}$ in which repeated values were eliminated and let $\bar{v}_0, \ldots, \bar{v}_{\bar{\kappa}+1}$ be the corresponding (reordered) values. We define a piecewise linear function $L : [0,1] \to \mathbb{R}$ such that $L(\bar{p}_j) = \bar{v}_j$ for $j = 0, \ldots, \bar{\kappa}+1$. The transformation $\mathcal{S}_k$ is given by $\mathcal{S}_k(v_0, \ldots, v_{\kappa+1}, p_1, \ldots, p_\kappa) := x^k + d(v,p)$, where $[d(v,p)]_i = L((i-1)/(n-1))$ for $i = 1, \ldots, n$. So the *bound constrained* problem to be solved at iteration k is given by

$$\underset{(v,p)\in\mathbb{R}^{n_{\text{red}}}}{\text{Minimize}} \frac{1}{2}\|F(\mathcal{S}_k(v,p))\|_2^2 \text{ subject to } 0 \leq p_j \leq 1 \text{ for } j = 1, \ldots, \kappa. \tag{35}$$

As initial guess, we consider $v = 0$ and p with random (with uniform distribution) components $p_j \in [0,1]$ for $j = 1, \ldots, \kappa$. Once again, since (35) is expected to be small, it can be approximately solved by any derivative-free method capable of dealing with small unconstrained problems. In the present work we intend to use BOBYQA [34].

3.3 Acceleration Algorithm

We adopt a Sequential Secant approach for defining the acceleration. The scheme, that generalizes the one adopted in [4] for solving nonlinear sytems of equations, is as follows.

1. If $k = 0$, then define $x^{k+1} = x^{\text{trial}}$.
2. If $k > 0$, then choose $k_{\text{old}} \in \{0, 1, \ldots, k-1\}$,

$$s^j = x^{j+1} - x^j \text{ for all } j = k_{\text{old}}, \ldots, k-1,$$
$$s^k = x^{\text{trial}} - x^k,$$
$$y^j = F(x^j + s^j) - F(x^j) \text{ for all } j = k_{\text{old}}, \ldots, k.$$
$$S_k = (s^{k_{\text{old}}}, \ldots, s^k),$$
$$Y_k = (y^{k_{\text{old}}}, \ldots, y^k),$$
$$x^k_{\text{accel}} = x^k - S_k Y_k^\dagger F(x^k).$$

3. If $f(x^k_{\text{accel}}) \leq f(x^{\text{trial}})$, then define $x^{k+1} = x^k_{\text{accel}}$. Otherwise, define $x^{k+1} = x^{\text{trial}}$.

This algorithm differs from the plain acceleration scheme defined in (2) in a very substantial way. In (2), the definition of x^{k+1} depends only on the previous iterates and lies in the affine subspace determined by them. Therefore, in (2), the successive iterates do not escape from a fixed p-dimensional affine subspace, where p is the number of previous iterates that contribute to the acceleration process. On the contrary, here, we define x^{k+1} as the possible result of an acceleration that includes the trial point x^{trial} which, in principle, does not belong to any pre-determined affine subspace. As a consequence, the accelerated point has the chance of exploring the whole domain in a more efficient way.

4 Estimation of Manning coefficients in the Saint-Venant equation

In the present work, we are interested in the estimation of parameters in one-dimensional models that simulate water or mud flow in natural channels. The presence of extreme boundary conditions can be a consequence of upstream levee breakage, a subject that is studied in the context of the interdisciplinary research and action group CRIAB (acronym for "Dams Conflicts, Risks and Impacts" in Portuguese) at the University of Campinas. The initial conditions for this type of models are, in general, well known, but the parameters reflecting density, friction, obstacles, or terrain features must be estimated from data. Mathematical models for this type of phenomena consist of partial differential equations with boundary conditions that simulate flood intensity. The use of programs whose source code is not available is frequent in this type of research. For this reason we are interested in investigating the behavior of derivative-free methods to estimate parameters of the models used.

For simplicity, in this study we assume that the phenomenon we are interested in is well represented by the Saint-Venant equations [37]. More sophisticated tools are beyond the scope of the present work. The Saint-Venant equations

$$A_t + Q_x = 0 \tag{36}$$

and

$$Q_t + (Q\,V)_x + g\,A\,\widehat{z} + \frac{\xi\,P\,V\,|V|}{8} = 0. \tag{37}$$

simulate the evolution of mean velocity, wetted cross-sectional area, depth, and flow in a one-dimensional channel. In (36) and (37), $A = A(x,t)$ is the wetted cross sectional area at position x and time t; $V = V(x,t)$ is the mean velocity; $Q = Q(x,t) = A(x,t)\,V(x,t)$ is the flow rate; $P = P(x,t)$ is the wetted perimeter, that is, the perimeter enclosing the wetted area taking away the air contact surface; g is the acceleration of gravity, approximately $9.8 m/s^2$; $\widehat{z} = \widehat{z}(x,t) = z_x/(1+(z_x)^2)$, where $z = z(x,t) = h(x,t) + z_b(x)$, $h(x,t)$ is the maximum channel depth at point x and time t, and $z_b(x)$ is the vertical coordinate of the channel bottom at point x (therefore, $z_x = h_x + (z_b)_x$); and $\xi = \xi(x)$ is the adimensional Manning coefficient whose estimation using data is the subject of the present study. The estimation of Manning coefficients is a very hard problem related with the simulation of floods in natural channels [14]. In the present work, we adopt that (a) Manning coefficients vary at different points of the channel but are invariant in time and (b) the best estimation of Manning coefficients is the one that provides the best predictions of streams in a period of time.

We assume that the channel under consideration extends one-dimensionally from $x = x_{\min}$ to $x = x_{\max}$. The boundary condition on the left ($x_{\min}$) simulates a flow rate that grows linearly from 8.245 m^3/s to 200 m^3/s in 1,200 seconds a decreases to the initial flow rate between 1,200 seconds and 3,600 seconds, remaining stationary thereafter. The initial depth is 1.2 meters. The second derivatives of other state variables are assumed to be zero both in $x = x_{\min}$ and in $x = x_{\max}$. We consider that wetted cross-sectional areas and velocities are measured between times $t = t_{\min}$ and $t = t^{\text{obs}}_{\max}$, at equally spaced points in the interval $[x_{\min}, x_{\max}]$. The physical characteristics of this channel were taken from [30] and [15]. Synthetic data were created with $x_{\min} = 0$ meters, $\Delta x = 6$ meters, $x_{max} \in \{3{,}000, 3{,}600, \ldots, 9{,}000\}$ meters, $t_{\min} = 0$ seconds, and $\Delta t = 0.1$ seconds. The value of $t^{\text{obs}}_{\max}$, maximum observation time, was subject to experimentation; while $t^{\text{pred}}_{\max}$, maximum time for prediction, was set to $t^{\text{pred}}_{\max} = 3{,}600$ seconds. The transversal area was considered to be rectangular with a width of 5 meters. We assumed that the true value for the adimensional Manning coefficients at the discretizated space points is 0.0366 plus a random uniform perturbation of up to 1%. We set $(z_b)_x = 0.001$. For the purposes of this research, we found it satisfactory to solve the Saint-Venant equations by finite differences using a Lax-Friedrichs type scheme [25] with artificial diffusion coefficient equal to 0.9.

The considerations above lead to a problem of the form

$$\underset{\xi\in\mathbb{R}^{n_x}}{\text{Minimize}} \sum_{i=1}^{n_t}\sum_{j=0}^{n_x}\sum_{k=1}^{2}\left(y(\xi,t_i,x_j,k)-y_{ijk}^{\text{obs}}\right)^2, \tag{38}$$

where $x_j = x_{\min} + j\Delta x$ for $j = 0,\ldots,n_x$, $t_i = t_{\min} + i\Delta t$ for $i = 1,\ldots,n_t$ and n_x, $x_{\min}$, Δx, n_t, $t_{\min}$, and Δt are given. When $k = 1$, y_{ijk}^{obs} ($i = 1,\ldots,n_t$, $j = 0,\ldots,n_x$) corresponds to a given observation of transversal area; while, when $k = 2$, it corresponds to a given observed velocity. The problem has n_x unknowns and $2n_t(n_x+1)$ terms in the summation. $y(\xi,t_{\min},x_j,k)$ does not depend on ξ and it assumed to be known for $k = 1,2$ and $j = 0,\ldots,n_x$; while the given values of Δx and Δt are such that, if ξ and $y(\xi,t_i,x_j,k)$ for $j = 0,\ldots,n_x$ are known, then $y(\xi,t_{i+1},x_j,k)$, for $j = 0,\ldots,n_x$, may be computed in finite time. In a generalization to (38), it is assumed that most of the observations are not available and, then, (38) is substituted with

$$\underset{\xi\in\mathbb{R}^{n_x}}{\text{Minimize}} \sum_{\{(i,j,k)\in S\}}\left(y(\xi,t_i,x_j,k)-y_{ijk}^{\text{obs}}\right)^2, \tag{39}$$

where $S \subseteq \widehat{S}$ is given, $\widehat{S} = \{(i,j,k) \mid i = 1,\ldots,n_t,\ j = 0,\ldots,n_x,\ k = 1,2\}$, and $|S| \ll |\widehat{S}|$. For further reference, we denote by $n_o = |S|$ the number of available observations. Note that, if $S = \widehat{S}$, then $n_o = 2n_t(n_x+1)$; while if, for example, only 10% of the observations are available, then we have $n_o = 0.2n_t(n_x+1)$.

Approximately solving (39) provides a value $\bar{\xi}$; and this value $\bar{\xi}$ is then used to predict that

$$y_{ijk}^{\text{obs}} \approx y(\bar{\xi},t_i,x_j,k) \text{ for } (i,j,k)\in\widehat{S}^{+},$$

where $t_{\max}^{\text{obs}} = t_{\min} + n_t\Delta t$ is the largest time instant at which observations considered in (39) were collected, $t_{\max}^{\text{pred}} > t_{\max}^{\text{obs}}$, and $\widehat{S}^{+} = \{(i,j,k) \mid t_{\max}^{\text{obs}} < t_i \le t_{\max}^{\text{pred}},\ j = 0,\ldots,n_x,\ k = 1,2\}$ represents the set of indices of the y_{ijk}^{obs}, *not yet observed*, whose predicted value is given by $y(\bar{\xi},t_i,x_j,k)$.

4.1 Numerical results

We implemented Algorithm 3.1, together with the two Reduction Algorithms (Sections 3.1 and 3.2) and the Acceleration Algorithm (Section 3.3) in Fortran 90. In the Reduction Algorithms, subproblems are solved using BOBYQA [34]. All tests were conducted on a computer with a 3.4 GHz Intel Core i5 processor and 8GB 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. Source codes are freely available for download at `https://www.ime.usp.br/~egbirgin/`. In the rest of this section, mainly in figures and tables, Algorithm 3.1 is sometimes referred to as SESEM, that stands for "Sequential Secant Method". Based on [4] and on preliminar numerical experiments, we set $\gamma = 10^{-4}$, $\eta_k = 2^{-k}$ for $k = 0,1,\ldots,$ and $\Delta = 10$ in Algorithm 3.1, and $p = 1000$, i.e. $k_{\text{old}} = \max\{0, k-p\}$, in the Acceleration Algorithm.

In section 4.1.1, we aim to determine (i) the amount of observations (starting at $t_{\min} = 0$ and at intervals $\Delta t = 0.1$ seconds), determined by the maximum observation time $t_{\max}^{\text{obs}}$, and (b) the precision of the optimization process that are required to recover Manning coefficients ξ suitable for making predictions up to $t_{\max}^{\text{pred}} = 3{,}600$ seconds. Sections 4.1.2 and 4.1.3 are related to the calibration and analysis of the proposed method. In Section 4.1.2, the dimension of the subproblem solved at each iteration is determined; while in Section 4.1.3 the influence of the Acceleration Algorithm in the overall process is observed. In Section 4.1.4, a set of instances of increasing size, mimic the the size of real-life instances, is solved. Section 4.1.5 presents the behavior of the solvers BOBYQA [34] and DFBOLS [48] in the set of considered instances.

4.1.1 Choice of a tolerance that leads to acceptable solutions

Given data coming from observations, we seek to estimate the Manning coefficients by means of which the Saint-Venant equations produce the best reproduction of data. In real cases, we are tempted to believe that an accuracy of around 10% in the prediction of depths and velocity is sufficiently good and that more accurate reproduction is not justified since observation and modeling errors may be, many times, of that order. However, we have no guarantees about the quality of predictions for data that are not available yet; and it can be argued that, although an excessive precision in the available data has no effect in the reproduction of these data, it may have a significant effect in the reproduction of observations that are not available yet. Therefore, it is sensible to test our inversion procedure not only up to the precision compatible with observation and modeling errors but also with moderate higher precisions.

Assume that an iterative optimization process is applied to (39) to compute $\bar{\xi}$; and that this process stops when it finds $\bar{\xi}$ satisfying

$$\left[\sum_{\{(i,j,k)\in S\}} \left(y(\bar{\xi},t_i,x_j,k) - y_{ijk}^{\text{obs}}\right)^2\right] \leq \epsilon \left[\sum_{\{(i,j,k)\in S\}} \left(y_{ijk}^{\text{obs}}\right)^2\right], \tag{40}$$

where $\epsilon > 0$ is a given tolerance. Of course, $\bar{\xi}$ depends on ϵ and on the problem data. In particular, $\bar{\xi}$ depends on the set of available observations S, that depends on $t_{\max}^{\text{obs}}$. Assume that, after computing $\bar{\xi}$, $t_{\max}^{\text{pred}} > t_{\max}^{\text{obs}}$ is chosen and observations y_{ijk}^{obs} with $(i,j,k) \in S^+ \subseteq \widehat{S}^+$ become available. We define that, for the given S^+ and $t_{\max}^{\text{pred}}$, $\bar{\xi}$ is *acceptable* if we have that

$$\eta(\bar{\xi}) := \frac{\sum_{\{(i,j,k)\in S\cup S^+\}} \left(y(\bar{c},t_i,x_j,k) - y_{ijk}^{\text{obs}}\right)^2}{\sum_{\{(i,j,k)\in S\cup S^+\}} \left(y_{ijk}^{\text{obs}}\right)^2} \leq 10^{-4}. \tag{41}$$

Let the problem data n_x, $x_{\min}$, Δx, $t_{\min}$, and Δt (note that n_t is missing here) be given and assume that an instant $t_{\max}^{\text{pred}}$ is chosen. The question is: Which are the number of observations n_o and the optimization tolerance ϵ that make the computed $\bar{\xi}$ to be acceptable? We aim to answer this question empirically considering a typical instance of (39) with $n_x = 500$, $x_{\min} = 0$, $\Delta x = 6$, $t_{\min} = 0$, $\Delta t = 0.1$, and S randomly chossen in such a way that $n_o = |S| \approx 0.1(2n_t(n_x+1))$, i.e. assuming that approximately 90% of the observations are not available. (Units of measure are meters for space and seconds for time.) Setting $t_{\max}^{\text{pred}} = 3{,}600$ and varying a constant $\nu \in \{10^{-4}, 2\times10^{-4}, \ldots, 40\times10^{-4}\}$, used to define $n_t(\nu)$ such that $t_{\max}^{\text{obs}} \approx \nu\, t_{\max}^{\text{pred}}$, we defined 40 instances of problem (39). (The number of observations is $n_o \approx 0.2n_t(\nu)(n_x+1)$; $\nu = 10^{-4}$ corresponds to $n_o = 427$, while $\nu = 4\times10^{-3}$ corresponds to $n_o = 14{,}460$.) For each instance, a solution satisfying (40) was computed considering 36 different tolerances $\epsilon \in \{7.5\times10^{-13}, 5\times10^{-13}, 2.5\times10^{-13}, \ldots, 10^{-4}\}$. For each combination (ν,ϵ), we obtained a solution $\bar{\xi}(\nu,\epsilon)$, that is said to be acceptable if (41) holds. Figure 1 displays, as a function of ν and ϵ, the value of the prediction error $\eta(\bar{\xi}(\nu,\epsilon))$ defined in (41). In the figure, cold colors (blue, cyan, and green) correspond to solutions that are not acceptable; while hot colors (yellow, orange, red, and dark red) correspond to acceptable solutions. The figure shows (on the left) that acceptable solutions were not found when the number of observations $n_o(\nu)$ was smaller than the number of unknowns $n_x = 500$. When the number of observations is larger than the number of unknowns, acceptable solutions are only found when $\epsilon \leq 10^{-9}$.

4.1.2 Choice of the subproblems' dimension

We now consider an instance of problem (39) with $n_x = 500$, $x_{\min} = 0$, $\Delta x = 6$, $n_t = 10$, $t_{\min} = 0$, $\Delta t = 0.1$, and S randomly chossen in such a way that $n_o = |S| \approx 0.1(2n_t(n_x+1))$, i.e. assuming

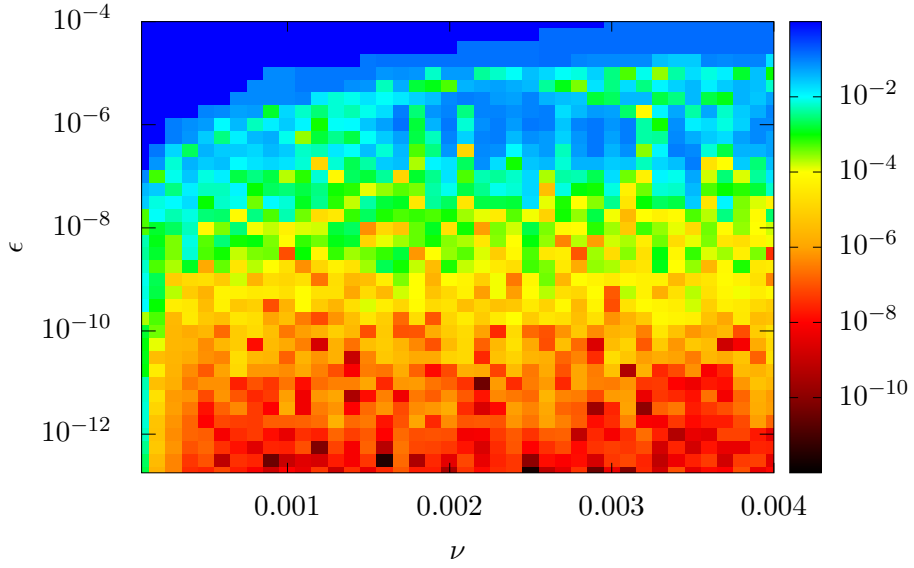

Figure 1: Acceptability of solutions $\bar{\xi}(\nu, \epsilon)$ to instances with varying number of observations $n_t(\nu)$ solved with varying tolerances ϵ. Hot colors show that acceptable solutions can be computed when the number of observations is larger than the number of unknowns and the tolerance to stop the optimization process is tight (smaller than 10^{-9}).

that approximately 90% of the observations are not available. The choice $n_t = 10$ combined with $\Delta t = 0.1$ means that observations are collected at intervals of 0.1 seconds during 1 second; and since we are assuming that 90% of the observations will not be available, this means that there will be $n_o \approx 0.1 \times 2 \times 10 \times (n_x + 1) = 2(n_x + 1) > n_x$ observations available. We aim to find solutions to this instance satisfying (40) with $\epsilon = 10^{-9}$ that, for this instance, corresponds to $f_{\text{target}} \approx 1.9633 \times 10^{-5}$. Due to analysis in the previous paragraph, it is expected the computed solution to be acceptable according to (41); so the solution can be used to make predictions for the next 3,559 seconds.

The instance in the previous paragraph will be used to observe the behavior of two variants of Algorithm 3.1, with affine-subspaces-based and with linear-interpolation-based subproblems, under variations of the subproblems' dimension n_{red}. Each variation of Algorithm 3.1 uses, at every iteration, the same reduction strategy and the same subproblem's dimension. As mentioned in the previous paragraph, $f_{\text{target}} \approx 1.9633 \times 10^{-5}$; while the initial guess is always $x^0 = 0$. Figures 2 and 3 show the results. Since both reduction strategies have a random component, the instance was solved ten times for each considered value of n_{red}. Figure 2 shows boxplots of two performance measures (CPU time and number of functional evaluations) of Algorithm 3.1 with affine-subspace-based subproblems and $n_{\text{red}} \in \{4, 5, \ldots, 9\} \cup \{10, 15, \ldots, 50\}$. (It is worth noting that in all cases the method stopped satisfying the stopping criterion (40) with $\epsilon = 10^{-9}$ as desired.) The boxplots show that the efficiency of the method is inversely proportional to the size of the subproblems. It must be observed that with $n_{\text{red}} \in \{2, 3\}$ (that are not being shown in the picture) the performance measures present a large standard deviation and some outliers, while the method fails a few times, characterizing a situation in which the method has difficulties in improving the current approximation to a solution by inspecting a very small search space. Figure 3 shows boxplots of two performance measures (CPU time and

number of functional evaluations) of Algorithm 3.1 with linear-interpolation-based subproblems and $n_{\text{red}} \in \{8, 10, 12, 14, 16, 18, 20, 30, 40, 50\}$. The boxplots show a uniform performance of the method for $n_{\text{red}} \leq 20$; while, for $n_{\text{red}} > 20$, the efficiency decreases when n_{red} increases.

Comparing Figures 2 and 3 and disregarding the difference in scale, it can be observed a difference in the standard deviation (over each batch of ten runs) when affine-subspaces-based and linear-interpolation-based subproblems are considered. The interpolation by splines is clearly problem oriented whereas the minimization in affine subspaces is not. Different Manning functions generated by interpolation are all meaningful, which is not the case when the Manning functions lie in more or less random affine subspaces. The subspace idea "ignores" the fact that the Manning is a continuous one-dimensional function. In this sense, it would be expected to observe smaller standard deviations in the case in which linear-interpolation-based subproblems are considered which, unfortunately, is not the case. Therefore, we presume that the observed behavior of both strategies is related to their sensitivity with respect to the initial guess used to solve the subproblems (described in Sections 3.1 and 3.2).

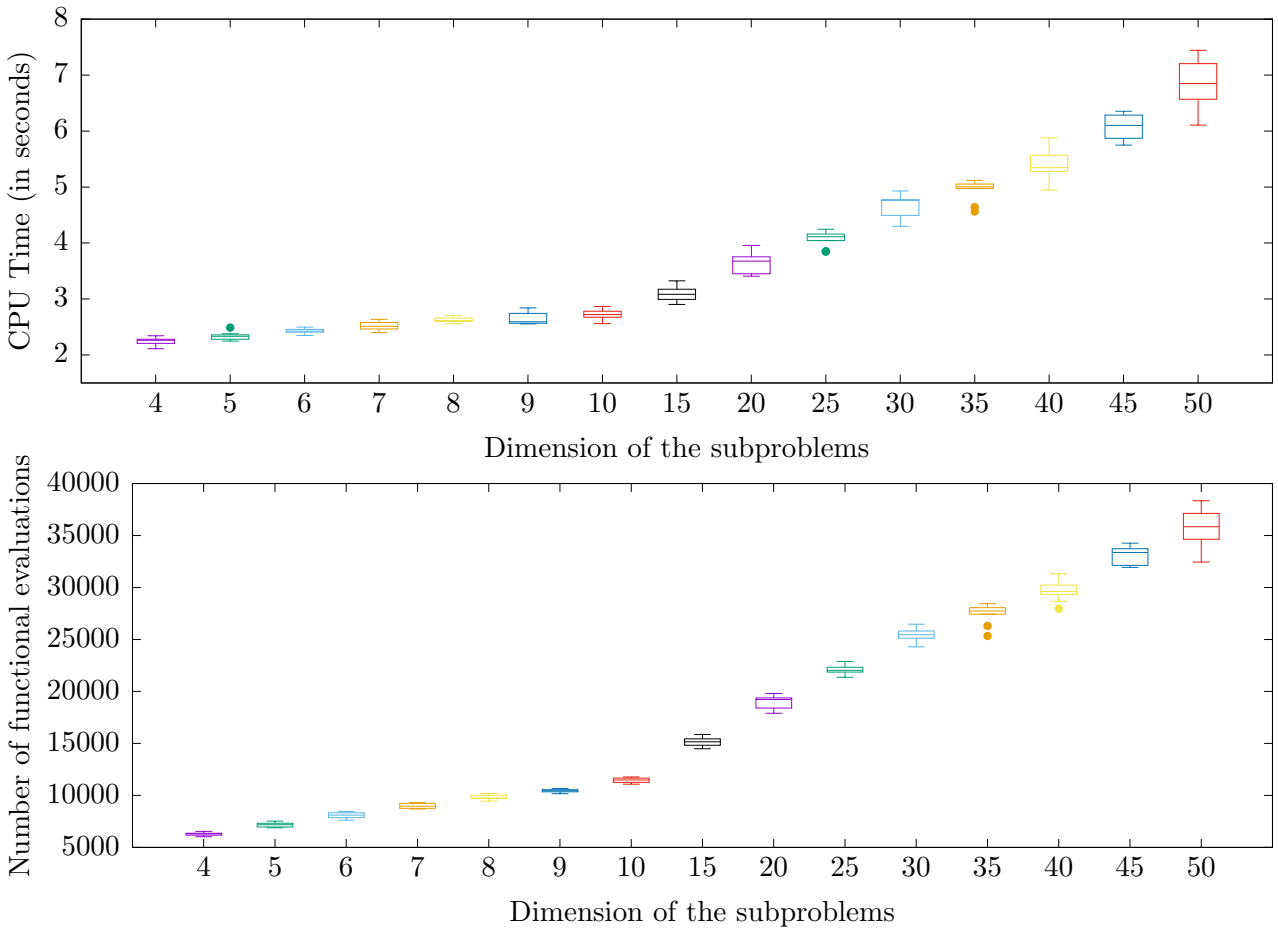

Figure 2: Boxplots of performance metrics of Algorithm 3.1 with the affine-subspaces-based reduction strategy applied to the instance with $n_x = 500$ varying the subproblems' dimension n_{red}.

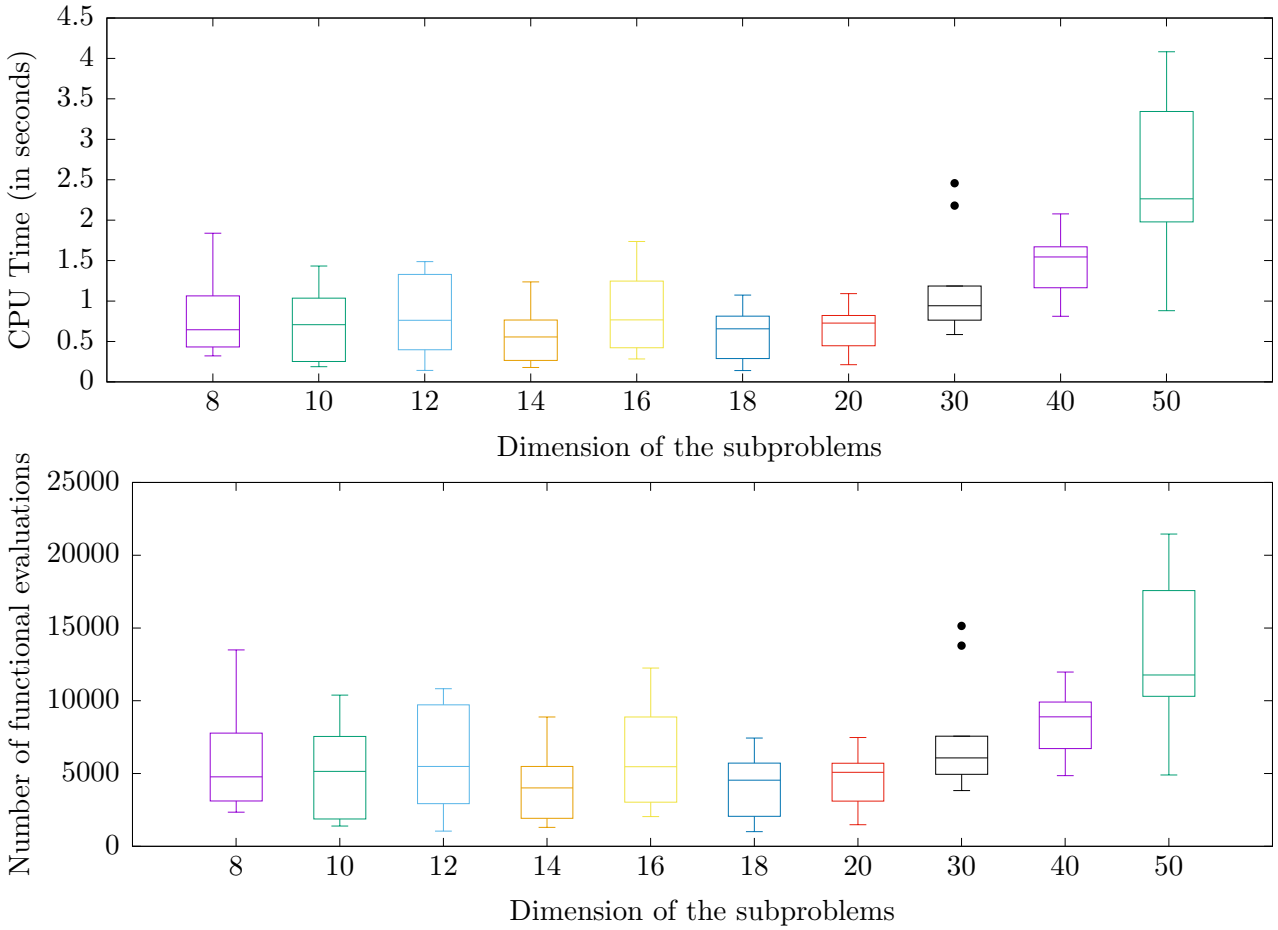

Figure 3: Boxplots of performance metrics of Algorithm 3.1 with the linear-interpolation-based reduction strategy applied to the instance with $n_x = 500$ varying the subproblems' dimension n_{red}.

4.1.3 Influence of the acceleration scheme

Still considering the same instance, we now analyze the influence of the acceleration in the performance of Algorithm 3.1 with affine-subspaces-based subproblems ($n_{\text{red}} = 4$) and with linear-interpolation-based subproblems ($n_{\text{red}} = 20$). Figure 4 shows the results. The figure shows that, when the affine-subspaces reduction strategy is considered, the acceleration improves the efficiency of the method in approximately two orders of magnitude; while it appears to have no relevant effect in combination with the linear-interpolation-based reduction strategy; although it appears to speed up the convergence of the method in its final iterations.

4.1.4 Solving larger instances

We now consider a set of instances exactly as the one already described but with $n_x \in \{500, 600, \ldots, 1{,}500\}$. (These values correspond to $x_{\max} = 3{,}000, 3{,}600, 4{,}200, \ldots, 9{,}000$, respectively.) Table 1 presents the performance of Algorithm 3.1 with affine-subspaces-based subproblems ($n_{\text{red}} = 4$) and with linear-interpolation-based subproblems ($n_{\text{red}} = 20$). As before, the initial guess x^0 is always the origin, f_{target} corresponds to the value of the right-hand-side in (40) with $\epsilon = 10^{-9}$. In the table,

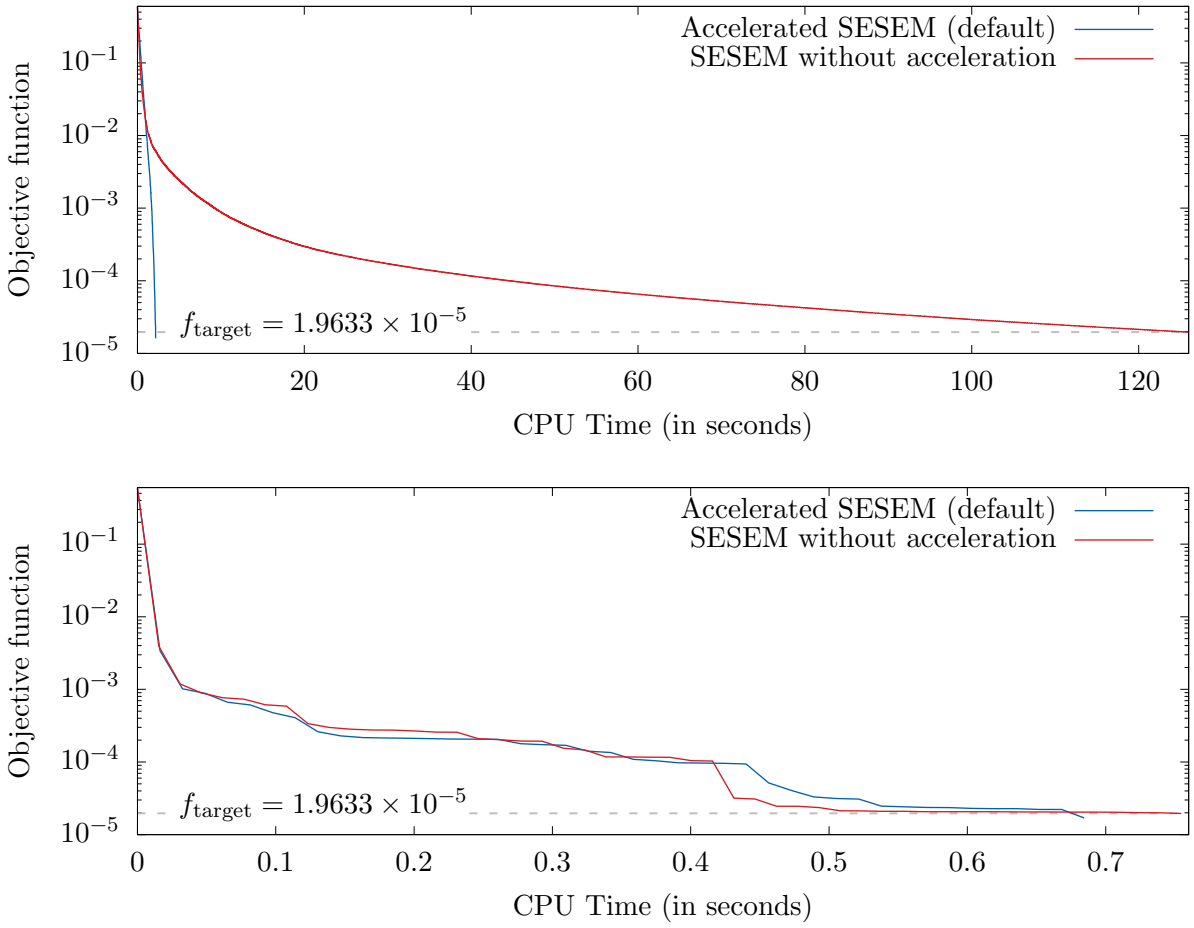

Figure 4: Influence of the acceleration in the performance of Algorithm 3.1 with affine-subspaces-based subproblems (top) and with linear-interpolation-based subproblems (bottom) when applied to the instance with $n_x = 500$.

$\|F(\bar{\xi})\|_2^2$ corresponds to the left-hand-side in (40), i.e.,

$$\|F(\bar{\xi})\|^2 = \sum_{\{(i,j,k)\in S\}} \left(y(\bar{\xi}, t_i, x_j, k) - y_{ijk}^{\text{obs}}\right)^2,$$

#it stands for the number of iterations, #fcnt stands for the number of functional evaluations, and Time stands for the CPU time in seconds. Since the method is run ten times per instance, values in the table correspond to averages. In addition, for the CPU time, the standard deviation is also presented in the table; and boxplots are given in Figures 5 and 6. A comparison between Figures 5 and 6 makes it clear that the cost of the affine-subspaces strategy grows together with the size of the instances; while the linear-interpolation strategy appears to absorve the cost of increasing sizes by incorporation some knowledge of the problem's solution. It is worth noticing that the trial point x^{trial} computed by the Reduction Algorithm at Step 2 satisfied the descent condition (32) 100% of the times; while the accelerated point x^k_{accel} at the Acceleleration Algorithm of Step 9 improved x^{trial} 99.16% of the times, in average.

n_x	n_o	SESEM with affine subspaces ($n_{\text{red}} = 4$)					SESEM with linear splines ($n_{\text{red}} = 20$)				
		$\|F(\bar{\xi})\|^2$	#it	#fcnt	Time		$\|F(\bar{\xi})\|^2$	#it	#fcnt	Time	
					avg	stdev				avg	stdev
500	1,058	1.86e-05	464	6,293	2.18	0.06	1.92e-05	38	4,598	0.64	0.29
600	1,257	2.10e-05	568	7,765	4.26	2.23	2.12e-05	39	5,094	0.80	0.45
700	1,471	2.64e-05	669	9,265	5.72	1.14	2.63e-05	42	5,109	0.92	0.54
800	1,669	3.06e-05	740	10,051	7.76	0.17	2.97e-05	43	4,984	1.02	0.49
900	1,908	3.53e-05	829	11,192	10.63	0.17	3.30e-05	45	5,513	1.26	0.51
1,000	2,106	3.83e-05	921	12,538	14.43	0.27	3.75e-05	45	6,021	1.62	0.66
1,100	2,312	4.26e-05	1,024	14,089	19.12	0.75	4.02e-05	41	5,304	1.47	0.77
1,200	2,484	4.67e-05	1,360	19,924	35.85	2.19	4.58e-05	45	5,372	1.62	0.81
1,300	2,677	5.05e-05	1,713	26,404	55.81	3.39	4.94e-05	46	5,802	1.89	1.06
1,400	2,885	5.45e-05	2,124	33,796	84.39	9.25	5.10e-05	43	5,227	1.84	0.94
1,500	3,090	5.84e-05	2,725	44,896	137.36	7.85	5.35e-05	48	5,916	2.39	1.04

Table 1: Performance of Algorithm 3.1 with affine-subspaces-based subproblems ($n_{\text{red}} = 4$) and with linear-interpolation-based subproblems ($n_{\text{red}} = 20$) applied to instances of increasing size with $x_{\max} \in \{3{,}000, 3{,}600, \ldots, 9{,}000\}$.

4.1.5 Comparison with BOBYQA and DFBOLS

This section ends presenting the performance of BOBYQA and DFBOLS[1] applied to the same instances of Table 1. Aiming a fair comparison, both methods were modified to stop as soon as they reach a solution ξ satisfying $\|F(\xi)\|_2^2 \leq f_{\text{target}}$. Table 2 shows the results. Figures in the table show that both variants of Algorithm 3.1 outperforms BOBYQA and DFBOLS by several orders of magnitude when the CPU time is considered as performance measure. While DFBOLS is the most time consuming method, it is the most efficient if the number of functional evaluations is considered. It is worth noticing that the comparison between the behaviors of the considered methods is restricted to their application to the problem under consideration.

5 Final remarks

In this paper, we presented a general scheme under which globally convergent derivative-free algorithms for nonlinear least squares with sequential secant acceleration can be defined. Our main motivation was the estimation of parameters in hydraulic models governed by partial differential equations. The non-availability of derivatives come from the fact that these models may be computed by "partially black-box" codes and the possible uncertainty of function evaluations motivated by the lack of precise geometrical parameters during the estimation process.

Algorithms based on interpolating quadratic models like BOBYQA [34] (see, also, [13]) use to be effective for this type of optimization problems. However, big costs associated with model building and its minimization make it necessary to employ schemes in which the number of variables is not very large. Partial minimization over random affine subspaces is an adequate dimension-reduction procedure [11, 42, 45]; and we showed that acceleration based on the sequential secant framework is effective to increase the performance of that approach. In addition, we developed a new reduction procedure based on variable linear interpolation in which the variables of subproblems are a set of

[1]Provided by Hongchao Zhang on January 11th, 2021.

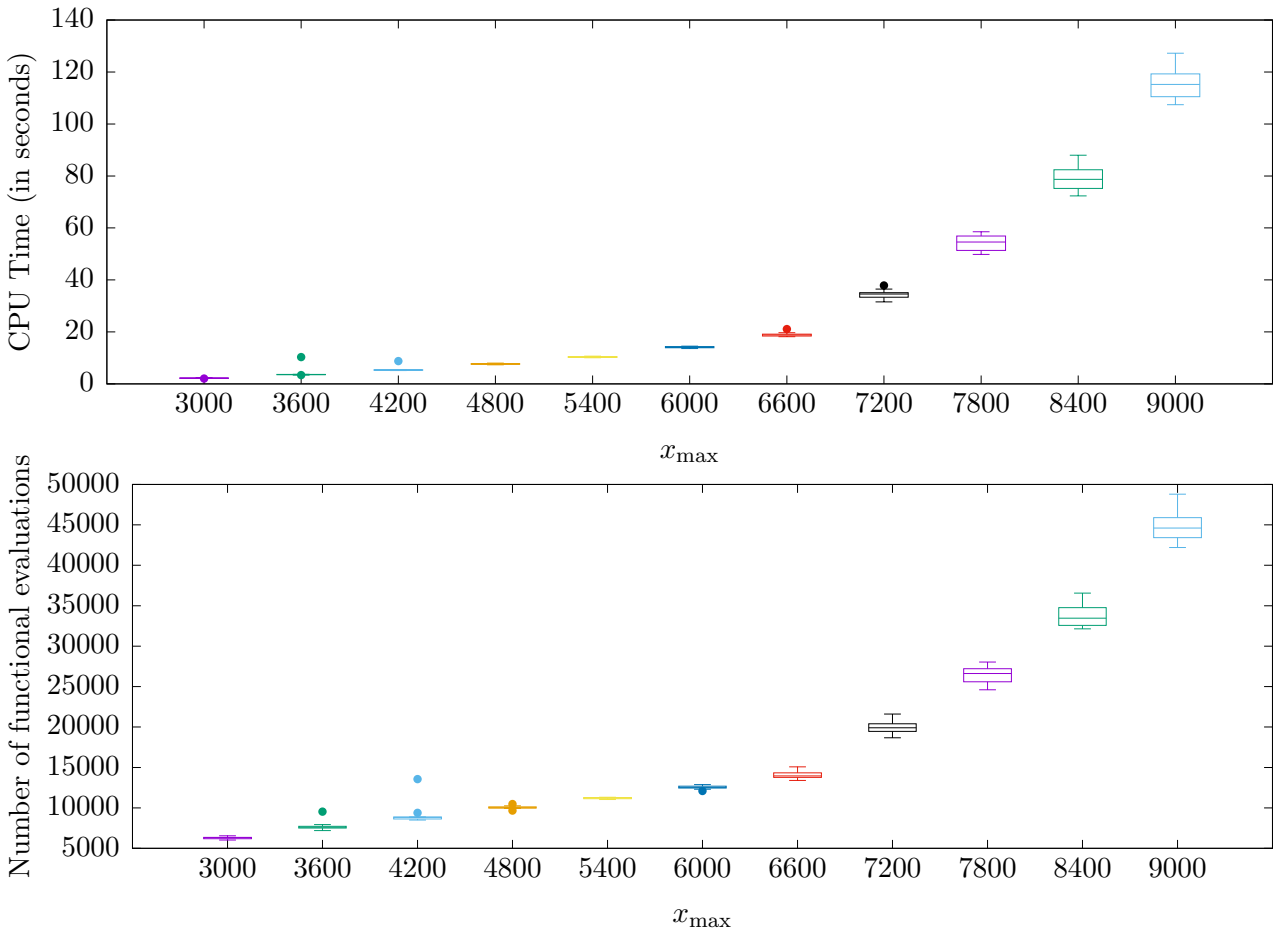

Figure 5: Boxplots of performance metrics of Algorithm 3.1 with affine-subspaces-based subproblems ($n_{\text{red}} = 4$) applied to instances of increasing size with $x_{\max} \in \{3{,}000, 3{,}600, \ldots, 9{,}000\}$.

independent variables with coordinates corresponding to (also) variable nodes. The effectiveness of this new approach is associated with the structure of the variables of the problem. If, in the original underlying problem, the unknown is a continuous function that depends on a single variable, the one-dimensional interpolatory scheme tends to be quite effective. This is the case of our problem of estimating the Manning coefficients, which, as a consequence, does not need acceleration to obtain the best possible results. In more complicated cases, the "true" unknown of the problem may be a continuous function of 2, 3, or more variables. In this case, our variable-node interpolation scheme should be conveniently adapted by means of incorporation of multi-dimensional interpolation devices. In the present work, the implementation of the proposed method, which combines a dimensionality reduction scheme and an acceleration process, made use of the general purpose derivative-free solver BOBYQA [34] to solve the small-sized subproblems arising at each iteration. For the latter purpose, any of the derivative-free least-squares methods such as those introduced in [10, 11, 48] could also be used instead.

The proposed method was tailored to solve a specific Engineering problem; and the developed framework includes steps whose implementation is dependent on the problem addressed. For the Manning coefficients problem, the linear-interpolation-based reduction technique had a crucial effect. In a different context, if $F(x) = (f_1(x), \ldots, f_m(x))^T$, in a given iteration k we could choose v^k consid-

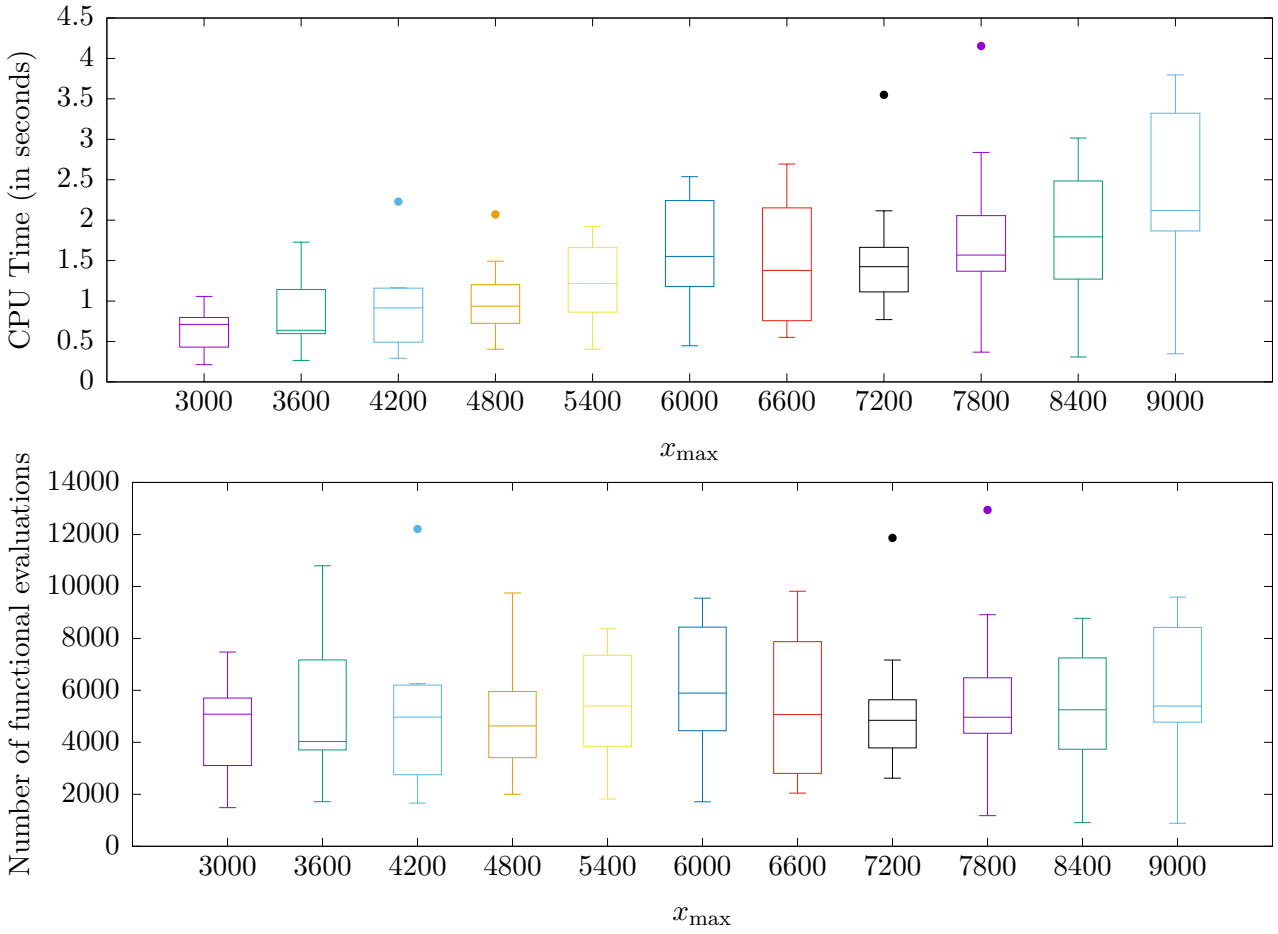

Figure 6: Boxplots of performance metrics of Algorithm 3.1 with linear-interpolation-based subproblems ($n_{\text{red}} = 20$) applied to instances of increasing size with $x_{\max} \in \{3{,}000, 3{,}600, \ldots, 9{,}000\}$.

ering the separability of the f_j or trying to minimize some specific subset of f_j, for exemple the ones with a larger contribution to $\|F(x^k)\|_2$, with a block coordinate descent flavour. Another problem-dependent factor is the parameter f_{target}, which affects the sufficient descent criterion – knowing whether the problem has null residue or not, influences the performance of the method. Thus, the proposed method can be seen as a general framework, and a deep study of the many possibilities it embeds is out of the scope of the present work.

In small problems, reduction techniques are meaningless. On the other hand, acceleration might work well, but it cannot be combined with independent calls to BOBYQA, seen as a black box, to compute each v^k. (v^k could be obtained by solving the original problem with a tolerance that goes to zero when k goes to infinity.) That way, in each call, BOBYQA is prevented from using the memory of the problems solved in the previous iterations, and the method as a whole ends up being inefficient. The correct way to use acceleration in BOBYQA would be to modify the BOBYQA source code. Each "standard BOBYQA new iterate" would play the role of x^{trial}. Using that x^{trial} and the previous iterates, an accelerated point x^k_{accel} would be calculated with the sequential secant approach and the best of the two would be taken as the new iterate x^{k+1}. However, modifying the BOBYQA source code is difficult and that task might be the subject of future work.

Considerations above lead to the claim that the implementation of Algorithm 3.1 introduced in

n_x	n_o	BOBYQA			DFBOLS		
		$\|F(\xi)\|^2$	#fcnt	Time	$\|F(\xi)\|^2$	#fcnt	Time
500	1,058	1.96e-05	7,593	278.68	2.63e-07	1,006	1,907.91
600	1,257	2.34e-05	8,176	511.07	2.39e-07	1,206	5,679.65
700	1,471	2.73e-05	10,601	1,057.75	5.16e-07	1,406	13,413.81
800	1,669	3.14e-05	15,353	1,948.76	1.16e-07	1,606	31,103.48
900	1,908	3.58e-05	12,901	2,133.89	–	–	>10h
1,000	2,106	3.92e-05	21,674	4,376.88	–	–	–
1,100	2,312	4.32e-05	18,916	5,967.14	–	–	–
1,200	2,484	4.67e-05	22,419	11,584.94	–	–	–
1,300	2,677	5.06e-05	26,074	16,742.44	–	–	–
1,400	2,885	5.46e-05	34,286	26,692.62	–	–	–
1,500	3,090	5.84e-05	29,473	26,325.37	–	–	–

Table 2: Performance of BOBYQA and DFBOLS applied to instances of increasing size with $x_{\max} \in \{3{,}000, 3{,}600, \ldots, 9{,}000\}$.

the present work performs well in large-scale least-squares problems in which variables of the problem correspond to the discretization of a continuous one-dimensional function. The present work makes no claims about the performance of the proposed method in other families of large scale problems, where it performance remains to be assessed.

References

[1] D. G. Anderson, Iterative procedures for nonlinear integral equations, *Journal of the Association for Computing Machinery* 12, pp. 547–560, 1965.

[2] Ch. Audet and J. E. Dennis, Jr., Mesh adaptive direct search algorithms for constrained optimization*SIAM Journal on Optimization* 17, pp. 188–217, 2006.

Read More: https://epubs.siam.org/doi/abs/10.1137/040603371

[3] J. G. P. Barnes, An algorithm for solving nonlinear equations based on the secant method, *Computer Journal* 8, pp. 66–72, 1965.

[4] E. G. Birgin and J. M. Martínez, Secant acceleration of sequential residual methods for large scale nonlinear systems of equations, arXiv:2012.13251v1.

[5] N. Boutet, R. Haelterman, and J. Degroote, Secant update version of quasi-Newton PSB with weighted multisecant equations, *Computational Optimization and Applications* pp. 1–26, 2020.

[6] N. Boutet, R. Haelterman, and J. Degroote, Secant update generalized version of PSB: a new approach, *Computational Optimization and Applications* 78, pp. 953–982, 2021.

[7] C. Brezinski, Convergence acceleration during the 20th century, *Journal of Computational and Applied Mathematics* 122, pp. 1–21, 2000.

[8] C. Brezinski and M. Redivo-Zaglia, *Extrapolation Methods Theory and Practice*, North-Holland, Amsterdam, 1991.

[9] C. Brezinski, M. Redivo-Zaglia, and Y. Saad, Shanks sequence transformations and Anderson acceleration, *SIAM Review* 60, pp. 646–669, 2018.

[10] C. Cartis and L. Roberts, A derivative-free Gauss-Newton method, *Mathematical Programming Computations* 11, pp. 631–674, 2019.

[11] C. Cartis and L. Roberts, Scalable subspace methods for derivative-free nonlinear least-squares optimization, arXiv:2102.12016.

[12] F. Chorobura, *Worst-case complexity analysis of derivative-free nonmonotone methods for solving nonlinear systems of equations*, Master Dissertation, Federal University of Paraná, Curitiba, PR, Brazil, 2020.

[13] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to Derivative-Free Optimization*, MPS-SIAM Series on Optimization, 2009.

[14] Y. Ding, Y. Jia, S. S. Y. Wang, Identification of Manning's roughness coefficients in shallow water flows, *Journal of Hydraulic Engineering*, pp. 501–510, 2004.

[15] W. H. Graf and M. S. Altinakar, *Hydraulique Fluviale - Tome 1: Ecoulement permanent uniforme et non uniforme*, Presses Polytechniques e Universitaires Romandes, Lausanne, 1993.

[16] H. R. Fang and Y. Saad, Two classes of multisecant methods for nonlinear acceleration, *Numerical Linear Algebra and Applications* 16, pp. 197–221, 2009.

[17] M. Frank and P. Wolfe, An algorithm for quadratic programming, *Naval Research Logistics Quarterly* 3, pp. 95–110, 1956.

[18] S. Gratton and Ph. L. Toint, Multi-secant equations, approximate invariant subspaces and multigrid optimization, *Technical Report 07/11*, Department of Mathematics, University of Namur – FUNDP, Namur, Belgium, 2007.

[19] S. Gratton, V. Malmedy, and Ph. L. Toint, Quasi-Newton updates with weighted secant equations, *Optimization Methods and Software* 30, pp. 748–755, 2015.

[20] R. Haelterman, A. Bogaers, J. Degroote, and N. Boutet, Quasi-Newton methods for the acceleration of multi-physics codes, *IAENG International Journal of Applied Mathematics* 47, pp. 352–360, 2017.

[21] N. Ho, S. D. Olson, and H. F. Walker, Accelerating the Uzawa algorithm, *SIAM Journal on Scientific Computing* 39, pp. 461–476, 2017.

[22] J. Jankowska, Theory of Multivariate Secant Methods, *SIAM Journal on Numerical Analysis* 16, pp. 547–562, 1979.

[23] W. La Cruz, J. M. Martínez, and M. Raydan, Spectral residual method without gradient information for solving large-scale nonlinear systems of equations, *Mathematics of Computation* 75, pp. 1429–1448, 2006.

[24] W. La Cruz and M. Raydan, Nonmonotone Spectral Methods for Large-Scale Nonlinear Systems, *Optimization Methods and Software* 18, pp. 583–599, 2003.

[25] R. J. LeVeque, *Numerical Methods for Conservation Laws*, Lectures in Mathematics, ETH Zürich, Birkäuser, 1992.

[26] T. Martini dos Santos, L. Reips, and J. M. Martínez, Under-relaxed quasi-Newton acceleration for an inverse fixed-point problem coming from positron-emission tomography, *Journal of Inverse and Ill-Posed Problems* 26, pp. 755–770, 2018.

[27] E. Meli, B. Morini, M. Porcelli, and C. Sgattoni, Solving nonlinear systems of equations via spectral residual methods: stepsize selection and applications, arXiv:2005.05851v2.

[28] P. Ni and H. F. Walker, Anderson acceleration for fixed-point iterations, *SIAM Journal on Numerical Analysis* 49, pp. 1715–1735, 2011.

[29] J. M. Ortega and W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic Press, 1970.

[30] R. M. Porto, *Hidráulica Básica*, EESC-USP, São Carlos, SP, Brazil, 2004.

[31] M. J. D. Powell, UOBYQA, Unconstrained optimization by quadratic approximation, *Mathematical Programming* 92, pp. 555–582, 2002.

[32] M. J. D. Powell, Least Frobenius norm updating of quadratic models that satisfy interpolation conditions, *Mathematical Programming* 100, pp. 183–215, 2004.

[33] M. J. D. Powell, Beyond symmetric Broyden for updating quadratic models in minimization without derivatives, *Mathematical Programming* 138, pp. 475–500, 2013.

[34] M. J. D. Powell, The BOBYQA algorithm for bound constrained optimization without derivatives, Report No. DAMTP 2009/NA06, Centre for Mathematical Sciences, University of Cambridge, 2009.

[35] M. L. Ralston and R. I. Jennrich, Dud, a derivative free algorithm for nonlinear least squares, *Technometrics* 20, pp. 7–14, 1978.

[36] T. Rohwedder and R. Schneider, An analysis for the DIIS acceleration method used in quantum chemistry calculations, *Journal of Mathematical Chemistry* 49, article number 1889, 2011.

[37] A. J. C. Saint-Venant, Théorie du mouvement non-permanent des eaux, avec application aux crues des rivière at à l'introduction des marées dans leur lit, *Comptes Rendus des Séances de Académie des Sciences* 73, pp. 147–154, 1871.

[38] K. Scheufele and M. Mell, Robust multisecant Quasi-Newton variants for parallel fluid-structure simulations–and other multiphysics applications, *SIAM Journal on Scientific Computing* 39, pp. 404–433, 2017.

[39] R. B. Schnabel, Quasi-Newton methods using multiple secant equations, *Technical Report CU-CS-247-83*, Deptartment of Computer Science, University of Colorado, Boulder, CO, USA, 1983.

[40] R. Varadhan and P. D. Gilbert, BB: An R package for solving a large system of nonlinear equations and for optimizing a high-dimensional nonlinear objective function, *Journal of Statistical Software* 32, article number 4, 2009.

[41] 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 XVIII International Conference on Water Resources, CMWR 2010*, J. Carrera, ed., CIMNE, Barcelona, 2010 (available online at `http://congress.cimne.com/CMWR2010/Proceedings/Start.html`).

[42] Z. Wang, Z. Wen, and Y.-X. Yuan, A subspace trust region method for large scale unconstrained optimization, in *Numerical Linear Algebra and Optimization*, Ya-Xiang Yuan ed., Science Press, 2004, pp. 264–274.

[43] S. M. Wild, Solving derivative-free nonlinear least squares problems with POUNDERS, in *Advances and Trends in Optimization with Engineering Applications*, T. Terlaky, M. F. Anjos, and S. Ahmed (eds.), SIAM, Philadephia, PA, USA, 2017, pp. 529–540.

[44] P. Wolfe, The secant method for simultaneous nonlinear equations, *Communications of ACM* 2, pp. 12–13, 1959.

[45] Y.-X. Yuan, Subspace methods for large scale nonlinear equations and nonlinear least squares, *Optimization and Engineering* 10, pp. 207–218, 2009.

[46] N. Zeev, O. Savasta and D. Cores, Nonmonotone Spectral Projected Gradient method applied to full waveform inversion, *Geophysical Prospecting* 54, pp. 525–534, 2006.

[47] H. Zhang and A. R. Conn, On the local convergence of a derivative-free algorithm for least-squares minimization, *Computational Optimization and Applications* 51, pp. 481–507, 2012.

[48] H. Zhang, A. R. Conn, and K. Scheinberg, A derivative-free algorithm for least-squares minimization, *SIAM Journal on Optimization* 20, pp. 3555–3576, 2010.