Economic inexact restoration for derivative-free expensive function minimization and applications *

E. G. Birgin† N. Krejić‡ J. M. Martínez§

September 18, 2020

Abstract

The Inexact Restoration approach has proved to be an adequate tool for handling the problem of minimizing an expensive function within an arbitrary feasible set by using different degrees of precision in the objective function. The Inexact Restoration framework allows one to obtain suitable convergence and complexity results for an approach that rationally combines low- and high-precision evaluations. In the present research, it is recognized that many problems with expensive objective functions are nonsmooth and, sometimes, even discontinuous. Having this in mind, the Inexact Restoration approach is extended to the nonsmooth or discontinuous case. Although optimization phases that rely on smoothness cannot be used in this case, basic convergence and complexity results are recovered. A derivative-free optimization phase is defined and the subproblems that arise at this phase are solved using a regularization approach that take advantage of different notions of stationarity. The new methodology is applied to the problem of reproducing a controlled experiment that mimics the failure of a dam.

Key words: Inexact Restoration, derivative-free, inexact evaluation, expensive function, global convergence, algorithms.

2010 Mathematics Subject Classification: 65K05, 65K10, 90C30, 90C56, 65Y20, 90C90.

1 Introduction

For many reasons scientists and engineers may need to optimize problems in which the objective function is very expensive to evaluate. In these cases, partial, and obviously inexact, evaluations are useful. The idea is to decrease as much as possible functional values using partial evaluations

---

*This work has been partially supported by the Brazilian agencies FAPESP (grants 2013/07375-0, 2016/01860-1, and 2018/24293-0) and CNPq (grants 302538/2019-4 and 302682/2019-8) and by the Serbian Ministry of Education, Science, and Technological Development.

†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 Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia. e-mail: natasak@uns.ac.rs

§Department of Applied Mathematics, Institute of Mathematics, Statistics, and Scientific Computing (IMECC), University of Campinas, 13083-859 Campinas SP, Brazil. e-mail: martinez@ime.unicamp.br
in such a way that, when we have no chance except to evaluate the function with maximal accuracy, we are already close enough to a solution of the problem. Rational decisions about when to increase accuracy (and evaluation cost) or even when to try more inexact evaluations are hard to make. Roughly speaking, we need a compromise between accuracy of evaluation and functional decrease that is difficult to achieve on a mere heuristic basis.

Most papers on minimization methods with inexact evaluations aim to report the behavior of modifications of standard algorithms in the presence of errors in the computation of the objective function, its derivatives, or the constraints \([5, 6, 10, 33, 21, 31, 32]\). In general, it is assumed that the objective function and, perhaps, its derivatives, can be computed with a given error bound each time the functional value is required at an arbitrary point. In \([5]\), the underlying “exact” algorithm is based on adaptive regularization. Trust-region algorithms inspired in \([23, \S 10.6]\) are addressed in \([34]\). High-order complexity of stochastic regularization is considered in \([6]\). The case of PDE-constrained optimization is studied in \([40]\). Nonsmoothness is addressed in \([33]\). Convex quadratic problems using Krylov methods are studied in \([32]\). Many of these works seem to be influenced by an early paper by Carter \([21]\).

In recent papers \([9, 10, 41]\), a methodology based on the analogy of the Inexact Restoration idea for continuous constrained optimization and the process of increasing accuracy of function evaluations was developed. In this approach, it is not assumed that one is able to compute a functional value (let alone derivatives) within a given required error bound. Instead, it is assumed that accuracy is represented by an abstract function \(h(y)\) such that \(h(y) = 0\) means maximal accuracy and \(y\) is a case-dependent procedure. For example, \(y \in Y\) may represent an algorithm with which one can compute the approximate objective function and \(h(y) \geq 0\) is an accuracy-related function. The connection between the value of \(h(y)\) and a possible error bound is not assumed to be known. For example, \(y\) may represent the maximal number of iterations that are allowed for a numerical algorithm that computes the objective function before obtaining convergence when we know that convergence eventually occurs but an error estimation is not available. Several additional examples may be found in \([10]\).

Inexact Restoration methods for smooth constrained optimization were introduced in \([44]\). Each iteration of an Inexact Restoration method proceeds in two stages. In the Restoration Phase, infeasibility is reduced; and, in the Optimization Phase, the reduction of the objective function or its Lagrangian is addressed with a possible loss of feasibility. Convergence with sharp Lagrangians as merit functions was proved in \([43]\). Applications to bilevel programming were given in \([45, 1]\). In \([19]\), an application to multiobjective optimization was described. The first line-search implementation was introduced in \([27]\). Nonsmooth versions of the main algorithms were defined in \([2, 24, 26]\). The employment of filters associated with Inexact Restoration was explored in \([35, 24]\). Applications to control problems were given in \([37, 12, 36, 3]\). In \([25]\), Inexact Restoration was used to obtain global convergence of a sequential programming method. Large-scale applications were discussed in \([31]\). In \([30]\), Inexact Restoration was used for electronic structure calculations; and problems with a similar mathematical structure were addressed in \([29]\). The reliability of Inexact Restoration for arbitrary nonlinear optimization problems was assessed in \([8]\). In \([20]\), the worst-case functional complexity of Inexact Restoration was analyzed. An application to finite-sum minimization was described in \([7]\). Continuous and discrete variables were considered in \([11]\). Non-monotone alternatives were defined in \([28]\). An application to the demand adjustment problem was given in \([48]\). Local convergence results were
proved in [12].

In [10], an algorithm of Inexact Restoration type applied to minimization with inexact evaluations that exhibits convergence and complexity results, was developed. However, the algorithm proposed in [10], as well as the algorithms previously introduced in [9, 41], employs derivatives of the objective function, a feature that may be inadequate in many cases in which one does not have differentiability at all. This state of facts motivates the present work. Here, the algorithms of [9, 10, 41] are adapted to the case in which derivatives are not available and the main theoretical results are proved. Moreover, we concentrate ourselves in a practical problem related with the prediction and mitigation of the consequences of dam breaking disasters.

The rest of this paper is organized as follows. The basic Inexact Restoration algorithm for problems with inexact evaluation without error bounds is introduced in Section 2. This algorithm is the one described in [10] with the difference that the domain Ω is an arbitrary metric space here, instead of a subset of \( \mathbb{R}^n \). This extension may be useful for cases in which we have discrete or qualitative variables. The proofs of the complexity and convergence results for this algorithm are similar to the ones reported in [10]. Since the employment of derivatives is essential for the definition of the Optimization Phase in [10], in the present contribution we need a different approach for that purpose. This is the subject of Section 3. The Optimization Phase requires sufficient descent of an iteration-dependent inexact evaluation function. Sufficient descent of this function is equivalent to single descent of an associated forcing function, therefore the required sufficient descent may be obtained applying an arbitrary number of iterations of a monotone minimization algorithm to the forcing associated function. This case-dependent monotone algorithm is called Algorithm A in the rest of the paper. In this phase, we may take advantage of the theoretically justified possibility of relaxing precision. In Section 4 we concentrate ourselves in the definition of a strategy for Algorithm A. Our choice is to solve the problem addressed by Algorithm A by means of the use of suitable surrogate models associated with iterated adaptive regularization. In Section 5 we describe an implementation that is adequate for the practical problem that we have in mind. The idea is to simulate a small-scale controlled physical experiment that mimics the failure of a dam, described in [22]. We rely on an MPM-like (Moving Particles Method) approach in which the dynamic of the particles is governed by the behavior of the Spectral Projected Gradient (SPG) method applied to the minimization of a semi-physical energy. Our optimization problem consists of finding the parameters of the energy function by means of which the trajectory of the SPG method best reproduces physical experiments reported in [22]. In Section 6 we state conclusions and lines for future research.

**Notation.** \( \mathbb{N}_+ \) denotes the non-negative integer numbers; while \( \mathbb{R}_+ \) denotes the non-negative real numbers. The symbol \( \| \cdot \| \) denotes the Euclidean norm of vector and matrices.

### 2 Main algorithm

This section presents the main problem considered in the present work and the algorithm proposed in [10], summarizing its theoretical results.

Consider the problem

\[
\text{Minimize (with respect to } x) \ f(x, y) \text{ subject to } h(y) = 0 \text{ and } x \in \Omega, \tag{1}
\]
where $\Omega$ is an arbitrary metric space, $Y$ is an abstract set, $h : Y \to \mathbb{R}_+$, and $f : \Omega \times Y \to \mathbb{R}$. (The case $\Omega \subseteq \mathbb{R}^n$ was considered in [10].) $Y$ is sometimes interpreted as a set of indices according to which the objective function is computed with different levels of accuracy. The nature of these indices varies with the problem and, for this reason, $Y$ is presented as an arbitrary set.

The function $h(y)$ is assumed to be related, in an unspecified way, with the accuracy in the evaluation of $f$ associated with the index $y \in Y$. The smaller the value of $h(y)$, the higher the accuracy; while maximal accuracy corresponds to $h(y) = 0$. In principle, problem (1) consists of minimizing the objective function with maximal accuracy.

Let us define a merit function $\Phi : \Omega \times Y \times (0, 1) \to \mathbb{R}$ by

$$\Phi(x, y, \theta) = \theta f(x, y) + (1 - \theta) h(y).$$

The merit function $\Phi$ combines objective function evaluation and accuracy level. $f(x, y)$ is the value of the objective function for a feasible $x$ when the accuracy level is determined by $y \in Y$. Therefore, $\Phi(x, y, \theta)$ represents a compromise between optimality and accuracy. The penalty parameter $\theta$ is updated at each iteration of the main algorithm. The algorithm introduced in [10] follows.

**Algorithm 2.1.** Let $x_0 \in \Omega$, $y_0 \in Y$, $\theta_0 \in (0, 1)$, $\nu > 0$, $r \in (0, 1)$, $\alpha > 0$, and $\beta > 0$ be given. Set $k \leftarrow 0$.

**Step 1.** *Restoration phase*

Define $y_k^{re} \in Y$ in such a way that

$$h(y_k^{re}) \leq rh(y_k) \tag{2}$$

and

$$f(x_k, y_k^{re}) \leq f(x_k, y_k) + \beta h(y_k). \tag{3}$$

**Step 2.** *Updating the penalty parameter*

If

$$\Phi(x_k, y_k^{re}, \theta_k) \leq \Phi(x_k, y_k, \theta_k) + \frac{1 - r}{2} \left( h(y_k^{re}) - h(y_k) \right), \tag{4}$$

set $\theta_{k+1} = \theta_k$. Otherwise, set

$$\theta_{k+1} = \frac{(1 + r) \left( h(y_k) - h(y_k^{re}) \right)}{2 \left( f(x_k, y_k^{re}) - f(x_k, y_k) + h(y_k) - h(y_k^{re}) \right)}. \tag{5}$$

**Step 3.** *Optimization phase*

Compute $y_{k+1} \in Y$ and $x_{k+1} \in \Omega$, such that

$$f(x_{k+1}, y_{k+1}) \leq f(x_k, y_k^{re}) - \alpha d(x_k, x_{k+1})^\nu \tag{6}$$

and

$$\Phi(x_{k+1}, y_{k+1}, \theta_{k+1}) \leq \Phi(x_k, y_k, \theta_{k+1}) + \frac{1 - r}{2} \left( h(y_k^{re}) - h(y_k) \right). \tag{7}$$

Update $k \leftarrow k + 1$, and go to Step 1.
Given an iterate \((x_k, y_k)\), at Step 1 we compute the function with an accuracy determined by \(y^\text{re}_k \in Y\), which is better than the accuracy given by \(y_k\). At Step 2 we update the penalty parameter with the aim that the merit function at the restored pair \((x_k, y^\text{re}_k)\) be smaller than the merit function computed at \((x_k, y_k)\). This is typical in Inexact Restoration methods for constrained optimization. At Step 3 (optimization phase) we compute the new iterate, at which the objective function value should decrease with respect to the restored pair in the sense of (6) and the merit function should be improved with respect to \((x_k, y_k)\) in the sense of (7).

**Assumption A1** At Step 1 of Algorithm 2.1, for all \(k \in \mathbb{N}_+\) it is possible to compute, in finite time, \(y^\text{re}_k\) satisfying (2) and (3).

**Lemma 2.1** Suppose that Assumption A1 holds. Then, Algorithm 2.1 is well defined.

Proof: See Lemma 2.3 of [10].

**Assumption A2** There exist \(h_{\text{max}} > 0\) and \(f_{\text{min}} \in \mathbb{R}\) such that, for all \(y \in Y\) and \(x \in \Omega\) we have that \(h(y) \leq h_{\text{max}}\) and \(f(x, y) \geq f_{\text{min}}\).

**Theorem 2.1** Suppose that Assumptions A1 and A2 hold. Given \(\varepsilon_{\text{feas}} > 0\), the number of indices \(k\) such that \(h(y_k) > \varepsilon_{\text{feas}}\) is bounded above by

\[
\frac{c_{\text{feas}}}{\varepsilon_{\text{feas}}} \tag{8}
\]

where \(c_{\text{feas}}\) only depends on \(x_0, y_0, r, \theta_0, \beta, h_{\text{max}},\) and \(f_{\text{min}}\).

Proof: See Corollary 2.2 of [10].

**Theorem 2.2** Suppose that Assumptions A1 and A2 hold. Then, the series \(\sum_{k=0}^{\infty} d(x_k, x_{k+1})^\nu\) is convergent. Moreover, given \(\varepsilon_{\text{opt}} > 0\), the number of iterates \(k\) at which \(d(x_k, x_{k+1}) > \varepsilon_{\text{opt}}\) is not bigger than

\[
\frac{c_{\text{opt}}}{\varepsilon_{\text{opt}}} \tag{9}
\]

where \(c_{\text{opt}}\) only depends on \(\alpha, x_0, y_0, r, \theta_0, \beta, h_{\text{max}},\) and \(f_{\text{min}}\).

Proof: See Theorem 2.2 and Corollary 2.3 of [10].

### 3 A general framework for the Optimization Phase

For the implementation of Step 3 of Algorithm 2.1 we may choose an arbitrary monotone (derivative-free) optimization Algorithm A and apply it to the minimization, with respect to \(x\), of \(f(x, y_{k+1}) + \alpha d(x, x_k)^\nu\). Since the value of this function for \(x = x_k\) is \(f(x_k, y_{k+1})\), it turns out that, due to monotonicity, Algorithm A will find \(x \in \Omega\) such that \(f(x, y_{k+1}) + \alpha d(x, x_k)^\nu \leq f(x_k, y_{k+1})\). Thus, condition (6) will take place, at least, when \(y_{k+1} = y^\text{re}_k\). Moreover, if we run Algorithm A up to the fulfillment of some reasonable approximate convergence criterion, this criterion will be fulfilled by \(x_{k+1}\), which, by Theorem 2.2, is such that \(d(x_k, x_{k+1}) \to 0\). In other words,
$x_{k+1}$ will satisfy an optimality criterion that corresponds to the minimization of the function $f(x, y_{k+1}) + \alpha d(x, x_k)^\nu$. This was the idea used in [17] that led to the choice of GSS [39, 38] as a subalgorithm for an Inexact Restoration method for derivative-free optimization with smooth constraints.

The following algorithm summarizes the procedure sketched above.

**Algorithm 3.1.**

**Step 1.** Choose $y_{k+1} \in Y$.

**Step 2.** Define, for all $x \in \Omega$,

$$F(x) = f(x, y_{k+1}) + \alpha d(x, x_k)^\nu. \quad (10)$$

**Step 3.** Consider the subproblem

Minimize $F(x)$ subject to $x \in \Omega$. \quad (11)

Using an appropriate monotone iterative (derivative-free) minimization Algorithm A, starting with $x = x_k$, compute $x_{\text{trial}} \in \Omega$ such that

$$F(x_{\text{trial}}) \leq F(x_k) \quad (12)$$

and $x_{\text{trial}}$ satisfies a stopping criterion related to Algorithm A to be specified later.

**Step 4.** If $y_{k+1} = y_k^e$ or

$$f(x_{\text{trial}}, y_{k+1}) \leq f(x_k, y_k^e) - \alpha d(x_k, x_{\text{trial}})^\nu \quad \text{and}$$

$$\Phi(x_{\text{trial}}, y_{k+1}, \theta_{k+1}) \leq \Phi(x_k, y_k, \theta_{k+1}) + \frac{1-r}{2} (h(y_k^e) - h(y_k)),$$ \quad (13)

then **return** $y_{k+1}$ and $x_{k+1} = x_{\text{trial}}$.

**Step 5.** Re-define $y_{k+1} = y_k^e$, and go to Step 2.

**Remark 3.1.** At Step 1 of Algorithm 3.1 we have the chance of improving the approximate solution already obtained using a looser precision than the one employed at the previous iteration. By means of a judicious choice of $y_{k+1}$ here we may save a lot of computer time obtaining, simultaneously, smaller function values and, consequently, substantial progress in terms of distance to the true solution of the original problem.

**Remark 3.2.** Observe that (12) is equivalent to

$$f(x_{\text{trial}}, y_{k+1}) \leq f(x_k, y_{k+1}) - \alpha d(x_k, x_{\text{trial}})^\nu$$ \quad (14)

and that when $y_{k+1} = y_k^e$ the fulfillment of (14) implies trivially the fulfillment of (6) and (7) taking $x_{k+1} = x_{\text{trial}}$. This is the reason why the test of (6) and (7) is not necessary at Step 4 when $y_{k+1} = y_k^e$. 
Under the assumptions of Theorems 2.1 and 2.2, given $\varepsilon_{\text{feas}} > 0$ and $\varepsilon_{\text{opt}} > 0$, at an iteration $k_{\text{end}}$ not larger that
\[
\frac{c_{\text{feas}}}{\varepsilon_{\text{feas}}} + \frac{c_{\text{opt}}}{\varepsilon_{\text{opt}}},
\]
where $c_{\text{feas}}$ and $c_{\text{opt}}$ are constants that only depends on problem and algorithmic constants, Algorithm 2.1 computes $y_{k_{\text{end}+1}}$ and $x_{k_{\text{end}+1}}$ such that
\[
h(y_{k_{\text{end}+1}}) \leq \varepsilon_{\text{feas}} \quad \text{and} \quad d(x_{k_{\text{end}}}, x_{k_{\text{end}+1}}) \leq \varepsilon_{\text{opt}}.
\]
It is worth noticing that the objective function of (11) at $x_{k_{\text{end}+1}}$ differs from $f(x_{k_{\text{end}+1}}, y_{k+1})$ in at most $\alpha \varepsilon_{\text{opt}}$. In addition, if at iteration $k_{\text{end}}$ Step 2 of Algorithm 2.1 uses Algorithm 3.1 plus Algorithm A to compute $y_{k_{\text{end}+1}}$ and $x_{k_{\text{end}+1}}$, then we can request to Algorithm A to stop with $x_{k_{\text{end}+1}}$ approximately fulfilling an optimality condition $C$ for problem (11) with precision $\eta > 0$ (in short, $(C, \eta)$). This means that the final iterate of Algorithm 2.1 will approximately satisfy an optimality condition for a problem with the same feasible set and a very similar objective function.

In the paragraph above we suggested that we need to run the auxiliary Algorithm A up to the fulfillment of $(C, \eta)$ at the “final” iteration only. Of course, except in very special cases, we do not know, in advance, if the current iteration is the “final” one or not. In order to ensure the fulfillment of $(C, \eta)$ at the final iteration of Algorithm 2.1, we could simply run Algorithm A up to its fulfillment at every iteration. This could be a very inefficient alternative since, as a matter of fact, one hopes that only one iteration of Algorithm A would be enough at each call at Step 2 of Algorithm 3.1 in order to get the desired final precision. Our hope rests on the fact that a sequence of A-iterations with different precisions $y_k$ probably have a similar effect as a single sequence of A-iterations with maximal precision. Therefore, many different strategies are possible for deciding how to employ Algorithm A. The choice of the best strategy is problem-dependent and depends also on the very nature of Algorithm A. In the following sections we will discuss different alternatives to be used as Algorithm A.

4 Using a surrogate regularized model

In this section we define a surrogate model regularization algorithm for solving problem (11). The specific form (10) will not be used in this context. (Recall that we only need to apply this algorithm up to high convergence precision at the “last” iteration of Algorithm 2.1.) We consider $F: \Omega \to \mathbb{R}$ where $\Omega$ is arbitrary. No conditions are imposed on $F$ except, of course, to be defined for all $x \in \Omega$. The main assumption to be used in this section is given below.

**Assumption A3** For each $\bar{x} \in \Omega$, $M(\bar{x}, x)$ will be a surrogate model of $F(x)$ such that:
1. For all $\bar{x} \in \Omega$, $M(\bar{x}, \bar{x}) = F(\bar{x})$.
2. There exists $L \geq 0$ and $p + 1 > 0$ such that, for all $\bar{x}, x \in \Omega$,
\[
F(x) \leq M(\bar{x}, x) + Ld(\bar{x}, x)^{p+1}.
\]
3. There exists $\sigma \geq 0$ such that $M(\bar{x}, x) + \sigma d(\bar{x}, x)^{p+1}$ is bounded below onto $\Omega$. 

7
4. For all $\bar{x} \in \Omega$ and $\sigma > 0$, it is affordable to solve the problem

$$\text{Minimize } M(\bar{x}, x) + \sigma d(\bar{x}, x)^{p+1} \text{ subject to } x \in \Omega.$$  \hspace{1cm} (16)

Note that the third condition in Assumption A3 implies that $M(\bar{x}, x) + \sigma' d(\bar{x}, x)^{p+1}$ is bounded below onto $\Omega$ for all $\sigma' \geq \sigma$. The algorithm for minimizing $F$ onto $\Omega$ based on the regularized model is defined as follows.

**Algorithm 4.1.** Let $\sigma_{\text{min}} > 0$, $\gamma > 0$, and $x_0 \in \Omega$ be given. Initialize $j \leftarrow 0$.

**Step 1.** Set $\ell \leftarrow 1$ and choose $\sigma_{j,1} \in [0, 1]$ such that $M(x_j, x) + \sigma_{j,1} d(x_j, x)^{p+1}$ is bounded below onto $\Omega$.

**Step 2.** Compute $x_{j,\ell} \in \Omega$ a solution of (16) with $\bar{x} = x_j$ and $\sigma = \sigma_{j,\ell}$.

**Step 3.** Test the condition

$$F(x_{j,\ell}) \leq F(x_j) - \gamma d(x_j, x_{j,\ell})^{p+1}.$$  \hspace{1cm} (17)

If (17) does not hold, then set $\sigma_{j,\ell+1} = \max\{\sigma_{\text{min}}, 2\sigma_{j,\ell}\}$, $\ell \leftarrow \ell + 1$, and go to Step 2.

**Step 3.** Set $x_{j+1} = x_{j,\ell}$, $\sigma_j = \sigma_{j,\ell}$, $j \leftarrow j + 1$, and go to Step 1.

It is worth noting that Algorithm 4.1 may be seen as a generalization of the projected gradient method for the convex constrained minimization of a smooth function, in which a trial point $x_{j,\ell}$ of the form $x_{j,\ell} = P_\Omega(x_j - \sigma_{j,\ell}^{-1} \nabla F(x_j))$ can be seen as the solution to subproblem (16) with $M(\bar{x}, x) := \nabla F(\bar{x})^T (x - \bar{x}) + \sigma \|x - \bar{x}\|^2$, $\bar{x} = x_j$, and $\sigma = \sigma_{j,\ell}$. Algorithm 4.1 has been defined without a stopping criterion. If $\eta > 0$ is a small tolerance, condition

$$d(x_{j+1} - x_j) \leq \eta,$$  \hspace{1cm} (18)

tested at Step 3 right before going back to Step 1, would correspond to the well-known stopping criterion

$$\|P_\Omega(x_j - \sigma_{j}^{-1} \nabla F(x_j)) - x_j\| \leq \eta,$$  \hspace{1cm} (19)

associated with the norm of the so called scaled continuous projected gradient (evaluated at $x_j$). Note that criterion (19) is satisfied by $x_j$ and that, by (17), $F(x_{j+1}) \leq F(x_j) - \gamma \eta^{p+1} < F(x_j)$.

The theorem below shows Algorithm 4.1 is well-defined and, additionally, it gives an evaluation complexity result for each iteration.

**Theorem 4.1** Assume that $M(\cdot, \cdot)$, $L \geq 0$, and $p + 1 > 0$ are such that Assumption A3 holds. Then, the $j$th iteration of Algorithm 4.1 is well defined and finishes with the fulfillment of (17) after at most $O(\log(L + \gamma))$ evaluations of $F$. 

8
Corollary 4.1 Assume that\( 2 \) The proof follows straightforwardly from this inequality.

Proof: By (15), and the fact that, by the definition of \( x_{j,\ell} \), \( M(x_j, x_{j,\ell}) + \sigma_{j,\ell}d(x_j - x_{j,\ell})^{p+1} \leq M(x_j, x_j) \), we have that

\[
F(x_{j,\ell}) \leq M(x_{j,\ell}, x_j) + Ld(x_j, x_{j,\ell})^{p+1} = M(x_j, x_{j,\ell}) + \sigma_{j,\ell}d(x_j, -x_{j,\ell})^{p+1} - \sigma_{j,\ell}\|x_{j,\ell} - x_j\|^{p+1} + Ld(x_j, -x_{j,\ell})^{p+1} = F(x_j) + (L - \sigma_{j,\ell})d(x_j, -x_{j,\ell})^{p+1}.
\]

Therefore, (17) holds if \( \sigma_{j,\ell} \geq L + \gamma \) that, by construction, occurs in the worst case when \( \ell \geq \log_2((L + \gamma)/\sigma_{\min}) + 2 \). Since \( F \) is evaluated only at points \( x_{j,\ell} \) to test condition (17), this completes the proof. \( \square \)

Theorem 4.2 Assume that \( F(x) \geq F_{\text{low}} \) for all \( x \in \Omega \), \( M(\cdot, \cdot), L \geq 0 \), and \( p + 1 > 0 \) are such that Assumption \( A3 \) holds and the sequence \( \{x_j\} \) is generated by Algorithm 4.1. Then, given \( \eta > 0 \), the number of iterations such that

\[
d(x_j, x_{j+1}) > \eta \tag{20}
\]

is bounded above by

\[
\left[ \frac{F(x_0) - F_{\text{low}}}{\gamma\eta} \right]^\frac{1}{p+1}. \tag{21}
\]

Proof: By (17) we have that, for all \( j = 0, 1, 2, \ldots \),

\[
F(x_{j+1}) \leq F(x_j) - \gamma d(x_j, x_{j+1})^{p+1}.
\]

The proof follows straightforwardly from this inequality. \( \square \)

Corollary 4.1 Assume that \( F(x) \geq F_{\text{low}} \) for all \( x \in \Omega \), \( M(\cdot, \cdot), L \geq 0 \), and \( p + 1 > 0 \) are such that Assumption \( A3 \) holds and the sequence \( \{x_j\} \) is generated by Algorithm 4.1. Then,

\[
\lim_{j \to \infty} d(x_j, x_{j+1}) = 0. \tag{22}
\]

Proof: The corollary follows immediately from Theorem 4.2. \( \square \)

We say that \( x_* \in \Omega \) is a critical point, of minimizing \( F \) over \( \Omega \), related to model \( M(\cdot, \cdot) \) and \( p + 1 > 0 \) (in short, \( (M,p)\)-critical) if there exists \( \sigma \in [0, 2L] \) such that \( x_* \) is a local minimizer of \( M(x_*, x) + \sigma d(x_*, x)^{p+1} \) subject to \( x \in \Omega \). In the following lemma we prove any local minimizer of \( F \) over \( \Omega \) is \((M,p)\)-critical.

Lemma 4.1 Suppose that \( x_* \in \Omega \) is a local minimizer of \( F(x) \) subject to \( x \in \Omega \). Assume, moreover, that \( M(\cdot, \cdot), L \geq 0 \), and \( p + 1 > 0 \) are such that Assumption \( A3 \) holds. Then, for all \( \sigma \geq L \), \( x_* \) is a local minimizer \( M(x_*, x) + \sigma d(x_*, x)^{p+1} \) subject to \( x \in \Omega \).
Proof: Assume that \( \sigma \geq L \) and that there exists a sequence \( \{x_j\} \) contained in \( \Omega \) that converges to \( x_* \) and \( M(x_*,x_j) + \sigma d(x_*,x_j)^{p+1} < M(x_*,x_*) = F(x_*) \) for all \( j \). Then, since \( \sigma \geq L \), \( M(x_*,x_j) + Ld(x_*,x_j)^{p+1} < F(x_*) \) for all \( j \). Therefore, by (15), \( F(x_j) < F(x_*) \) for all \( j \) and, thus, \( x_* \) is not a local minimizer of \( F(x) \).

We say that \( \bar{x} \in \Omega \) is a \( \eta \)-critical point, of minimizing \( F \) over \( \Omega \), related to model \( M(\cdot, \cdot) \) and \( p + 1 > 0 \) (in short, \( \eta-(M,p) \)-critical) if there exists \( \sigma \in [0,2L] \) and \( z \in \Omega \) such that \( d(\bar{x},z) \leq \eta \) and \( z \) is a local minimizer of \( M(\bar{x},x) + \sigma d(\bar{x},x)^{p+1} \) subject to \( x \in \Omega \).

Theorem 4.3 Assume that \( F(x) \geq F_{\text{low}} \) for all \( x \in \Omega \), \( M(\cdot, \cdot), L \geq 0 \), and \( p + 1 > 0 \) are such that Assumption A3 holds and the sequence \( \{x_j\} \) is generated by Algorithm 4.1. Then, given \( \eta > 0 \), the number of iterations such that \( x_j \) is not \( \eta-(M,p) \)-critical is bounded above by

\[
\left[ \frac{F(x_0) - F_{\text{low}}}{\gamma \eta} \right]^\frac{1}{p+1}.
\]

Proof: This theorem merely rephrases Theorem 4.2.

Remark 4.1. The \( \eta-(M,p) \)-criticality relies on a point being a stationary point of a regularized model. We now show an example that illustrates that regularization is an essential ingredient in this condition. Suppose that \( \Omega = \mathbb{R}^n \), \( F \) is \( p \) times continuously differentiable, \( x_* \) is an unconstrained minimizer of \( F \), \( M(x_*, x) \) is the \( p \)th Taylor polynomial of \( F \) at \( x_* \), and the \( p \)th derivatives of \( F \) at \( x_* \) are Lipschitz continuous, so that Assumption A3 holds. It is well known that \( M(x_*, x) \) has a local minimizer at \( x = x_* \) if \( n = 1 \) or \( p \in \{0,1,2\} \). However, a local minimizer of \( F \) may fail to exhibit that property if \( n \neq 1 \) and \( p > 2 \). As an example, take \( F : \mathbb{R}^2 \to \mathbb{R} \) given by

\[
F(z_1, z_2) = z_2^3 - z_1^3 z_2 + z_1^6.
\]

The origin is a global minimizer of \( F \). However, its Taylor polynomial of order \( p \in \{4, 5\} \) is

\[
M(z_1, z_2) = z_2^3 - z_1^3 z_2,
\]

for which the origin is not a local minimizer. For a similar example with \( p = 3 \), take

\[
F(z_1, z_2) = z_2^3 - z_1^2 z_2 + z_1^4.
\]

5 Numerical experiments

We begin this section by tackling the choice of an adequate surrogate model and describing a particular case of Algorithm 2.1 with Algorithm 3.1 in the Optimization Phase and Algorithm 4.1 for solving (11). Then, an application to the simulation of a dam failure is presented.

The choice of an adequate surrogate model is highly problem-dependent. Whereas in the smooth case Taylor-like models should be interesting, this is not the case when differentiability is out of question. A default approach in the context of this paper consists of choosing as model for \( f(x, y_{k+1}) + ad(x_k, x)\) (see (10)) the function

\[
M(x_k, x) = [f(x, y_k) + ad(x_k, x\) - f(x_k, y_k\) + f(x_k, y_{k+1}) \quad (23)
\]
where $k' \ll k + 1$. In this way, $M(x_k, x_k) = F(x_k)$ as required at Assumption $A3$ and solving (16) is, very likely, affordable, as the evaluation of $M(x_k, x)$ should be much cheaper than the evaluation of $f(x, y_{k+1})$ due to the requirement $k' \ll k$. We cannot ensure the fulfillment of the whole Assumption $A3$ and, in fact, it does not make sense to employ any effort in this verification since the lack of fulfillment of Assumption $A3$ would cause the lack of fulfillment of the descent condition (12). The particular case of Algorithms 2.1–4.1 that employs (23) as a surrogate model follows.

Algorithm 5.1. Let $x_0 \in \Omega$, $y_0 \in Y$, $\theta_0 \in (0, 1)$, $\nu > 0$, $r \in (0, 1)$, $\alpha > 0$, $\beta > 0$, $\varepsilon_{\text{feas}} > 0$, and $\varepsilon_{\text{opt}} > 0$ be given. Set $k \leftarrow 0$.

**Step 1. Restoration phase**

Define $y_{k}^{\text{re}} \in Y$ satisfying (2) and (3).

**Step 2. Updating the penalty parameter**

Compute $\theta_{k+1}$ by means of (4) and (5).

**Step 3. Optimization phase**

**Step 3.1.** Set $y_{k+1} = y_k$.

**Step 3.2.** If $h(y_{k+1}) \leq \varepsilon_{\text{feas}}$, then call Algorithm 4.1 with the surrogate model defined by (23) returning when the optimality condition $C(\varepsilon_{\text{opt}})$ is fulfilled. On return, stop.

**Step 3.3.** Execute one iteration of Algorithm 4.1 with the surrogate model defined by (23).

**Step 3.4.** If (6) or (7) is not fulfilled, then set $y_{k+1} = y_{k}^{\text{re}}$ and find $x_{k+1}$ satisfying (6) by means of Algorithm 4.1 with the surrogate model defined by (23).

**Step 4.** Set $k \leftarrow k + 1$ and go to Step 1.

**Remark.** If Algorithm 4.1 fails in any of the tasks required at Steps 3.2–3.4 in Algorithm 5.1, this reveals that Assumption $A3$ does not hold. In this case, stop with an appropriate failure message.

In what follows we describe the application of Algorithm 4.1 to the problem of fitting a descriptive model for a controlled physical experiment that aims to simulate and record the failure of a dam in [22].

As it is well known, hydro-geological hazardous natural phenomena like debris flows, avalanches and submerged landslides can cause significant damage and loss of lives and properties. A number of tragic incidents all over the world is well documented and analyzed in scientific literature but these events are extremely complex and still challenging in terms of mathematical modelling and numerical simulations. The key interaction that yields fast flow events is the one between solid grains and interstitial fluid. We briefly review here the results reported in [22] which are based on a small scale experiment and two numerical simulations with different methods.
Figure 1: Graphical representation of the four frames in [22, Fig.2, p.685]. Gray boxes represent boxes that are completely or partially filled with sand; while white boxes represent empty boxes.

The experiment [22] is a standard small-scale model of column collapse with saturated material that allows propagation in air in order to achieve similarity with natural flow-like landslide. A glass fume cuboid with fixed length and width is closed at one end while the second end has a movable vertical gate (see figure 1 of [22]). The cuboid is filled with a saturated granular mixture. The column height is varied to achieve different aspect ratios between height and length. The solid grains exhibit uniform granular distribution and density. The gate and opening mechanism is designed in such a way that partial desaturation prior to the collapse is avoided and the fast uplift of the gate triggers the propagation of the saturated mixture in a way that resembles natural disasters. The results of this physical experiment are documented by high resolution photos and essentially demonstrate similar behaviour for all aspect ratios. However, the time evolution of the normalized front position is different depending on the aspect ratio. Initially, when the gate is lifted both grain and water start moving forward at the open end of column. Then the upper parts head toward the bottom with the failing surface evolving in time. At the end of the process the granular front stops while the water filters through the solid phase.

Two conceptually different approaches are addressed in [22] for numerical simulations of column collapse. The first one, based on a Discrete Element Method (DEM), assumes that the granular material is represented by an assembly of particles interacting at contact points. The key issue, both conceptually and computationally, in this approach is the definition of a contact...
problem where the "radius" \( r \) > weight \( x \) through the iterates \( p \) construct a two-dimensional simulation of the dynamics of the dam-failure physical experiment where \( P \) starting from \( p \) the application of the Spectral Projected Gradient (SPG) method \([13, 14, 15, 16]\) to problem (25) identical balls with radius \( a \) solution to problem (25) consists in Cartesian space. 

d) show, for each of the four frames in \([22, \text{Fig.2, p.685}]\), the boxes completely or partially filled presented. (See \([22, \text{Fig.2, p.685}]\).) Pictures have a squared paper on background. Figures 1(a–d) show, for each of the four frames in \([22, \text{Fig.2, p.685}]\), the boxes completely or partially filled with sand.

Let \( p_j = ([p_{j1}], [p_{j2}])^T \in \mathbb{R}^2 \) for \( j = 1, \ldots, n_p \), \( p = (p_1^T, \ldots, p_{n_p}^T)^T \in \mathbb{R}^{2n_p} \), and

\[
\Psi_x(p) = x \left( \sum_{j=1}^{n_p} \sum_{i=j+1}^{n_p} \max \{0, (2r)^2 - \|p_j - p_i\|^2_2\} \right) + (1 - x) \sum_{j=1}^{n_p} [p_j]_2, \tag{24}
\]

where the “radius” \( r > 0 \) and the “weight” \( x \in (0, 1) \subset \mathbb{R} \) are given constant; and consider the problem

Minimize \( \Psi_x(p_1, \ldots, p_{n_p}) \) subject to \( p \in D \),

where \( D = \{ p \in \mathbb{R}^{2n_p} \mid p_j \geq 0 \text{ for } j = 1, \ldots, n_p \} \). Independently of the value of \( x \in (0, 1) \), a solution to problem \([25]\) consists in \( n_p \) points representing the centers of \( n_p \) non-overlapping identical balls with radius \( r \) “resting on the floor” of the positive orthant of the two-dimensional Cartesian space.

Assume that \( p^0 \in D \) corresponds to the configuration represented in Figure 2. Given a maximum of iterations \( y \in \mathbb{N}_+ \), let \( p^0, p^1, p^2, \ldots, p^\vartheta \in D \) be the \( \bar{y} \leq y \) iterates that result from the application of the Spectral Projected Gradient (SPG) method \([13, 14, 15, 16]\) to problem \([25]\) starting from \( p^0 \) and using as a stopping criteria a maximum of \( y \) iterations or finding an iterate \( p^\vartheta \) such that

\[\|P_D(p^\vartheta - \nabla \Psi_x(p^\vartheta)) - p^\vartheta\|_\infty \leq \varepsilon_{\text{opt}}^\text{SPG} := 10^{-8},\]

where \( P_D \) represents the projector operator onto \( D \). We aim to verify whether, by adjusting the weight \( x \) in \([24]\) and the maximum number of iterations \( y \) of the SPG method, it is possible to construct a two-dimensional simulation of the dynamics of the dam-failure physical experiment through the iterates \( p^0, p^1, p^2, \ldots, p^\vartheta \).
We consider solving problem (25) with \( r = 0.125 \) and \( n_p = 419 \) using SPG with the initial guess \( p^0 \) depicted on Figure 1. We associate with the four frames depicted in Figures 1(a–d) binary matrices \( M_1, M_2, M_3, M_4 \in \{ 0, 1 \}^{8 \times 20} \) that represent whether there is sand in each box of the frame or not. In an analogous way, we define \( M(p) \in \{ 0, 1 \}^{8 \times 20} \) as the matrix associated with the point \( p = (p_1^T, ..., p_{np}^T)^T \in D \subset \mathbb{R}^{2np} \), that indicates whether each box contains at least a point \( p_j \) or not. Given two \( A = (a_{ij}) \), \( B = (b_{ij}) \in \{ 0, 1 \}^{8 \times 20} \), we also define the fitness function

\[
\Pi(A, B) = \sum_{j=1}^{20} \sum_{i=1}^{8} |a_{ij} - b_{ij}|
\]

For a full sequence of iterates \( p^0, p^1, p^2, ..., p^\bar{y} \in \Omega \) of the SPG method, that depends on the weight \( x \) and the maximum of iterations \( y \), we define

\[
f(x, y) = 1 - \frac{1}{640} \max_{c \geq 0} \left\{ \sum_{\kappa=1}^{4} \Pi(M(p^{[c \kappa]}), M_\kappa) \right\},
\]

where, if \( [c \kappa] \geq \bar{y} \) and, thus, \( p^{[c \kappa]} \) does not exist, then we consider \( \Pi(M(p^{[c \kappa]}), M_\kappa) = 0 \). The value of \( f \) can be computed by inspection on \( c \). Finally, we define \( h(y) = 1/y \) and \( \Omega = \{ x \in \mathbb{R} \mid 0 \leq x \leq 1 \} \).

We implemented Algorithms 5.1 in Fortran. In the numerical experiments, we considered, \( \alpha = 10^{-4}, \beta = 100, \theta_0 = 0.5, \nu = 2, r = 2, p = 4, \varepsilon_{\text{feas}} = 1/12,800, \) and \( \varepsilon_{\text{opt}} = 10^{-4} \). The optimality condition defined by \( \varepsilon_{\text{opt}} \) is that the objective function to which this criterion is applied is not bigger, at the approximate optimizer \( z \), than the values at the feasible points of the form \( z = \varepsilon_{\text{opt}} \). The model \( M(x_k, x) \) is given by (23) with \( k' = \max\{0, k - 1\} \). For the approximate minimization of the model we employ standard global one-dimensional search. As initial guess, we considered \( (x^0, y^0) = (0.5, 100) \). 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. Table 1 shows the results; while Figure 3 illustrates the obtained solution. A rough comparison with Figure 9 of [22] indicates that our results are similar to the ones obtained by DEM-LBM and seem to be closer to the experimental results.
than the ones reported for MPM. As in the case of DEM-LBM and MPM, the discrepancies with respect to the experimental results are due to the behavior of the top part of the column. The whole process was completed in less than 1 minute of CPU time. The objective function equal to $1 - 618/640 \approx 0.03$ at the final iterate means that at 97% of the “pixels” the measured data matched the prediction of the model.

$$k \quad x^k \quad y^k \quad f(x^k, y^k)$$

<table>
<thead>
<tr>
<th>$k$</th>
<th>$x^k$</th>
<th>$y^k$</th>
<th>$f(x^k, y^k)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>100</td>
<td>$1 - 556/640$</td>
</tr>
<tr>
<td>1</td>
<td>0.9</td>
<td>200</td>
<td>$1 - 561/640$</td>
</tr>
<tr>
<td>2</td>
<td>0.99</td>
<td>400</td>
<td>$1 - 604/640$</td>
</tr>
<tr>
<td>3</td>
<td>0.99</td>
<td>800</td>
<td>$1 - 604/640$</td>
</tr>
<tr>
<td>4</td>
<td>0.999</td>
<td>1,600</td>
<td>$1 - 613/640$</td>
</tr>
<tr>
<td>5</td>
<td>0.99925</td>
<td>3,200</td>
<td>$1 - 617/640$</td>
</tr>
<tr>
<td>6</td>
<td>0.999275</td>
<td>6,400</td>
<td>$1 - 618/640$</td>
</tr>
<tr>
<td>7</td>
<td>0.999275</td>
<td>12,800</td>
<td>$1 - 618/640$</td>
</tr>
</tbody>
</table>

Table 1: Details of the application of Algorithm 5.1.

Figure 3: Graphical representation of $p^{384}$, $p^{961}$, $p^{1922}$, and $p^{4369}$ (corresponding to $c = 873.2$) of a run of SPG with $x = 0.999275$ and $y = 12,800$. The red line is a rough representation of the physical experiment profile according to Fig. 2 of [22].
Many natural phenomena tend to converge to equilibrium states that are characterized as minimizers of “energy functions”. This is the reason why physical laws sometimes inspire optimization algorithms. For example, in [49] a method for solving nonlinear equations is linked to a system of second-order ordinary differential equations inspired in classical mechanics. In this paper, we suggested a movement in the opposite direction: Simulating a physical phenomenon by means of the behavior of an optimization algorithm. Two facts are common to physical phenomena and optimization algorithms: On the one hand, energy should decrease in both cases and, on the other hand, the inconveniences for sudden decrease are, in both cases, local in nature. Of course this does not mean that physical events that are usually modelled by means of complex systems of differential equations may be always mimicked by the sequence of iterations of a simple minimization algorithm. However, the possibility that a minimization algorithm could give a first general approach to a complex phenomenon in which the basic principle is energy minimization cannot be excluded, especially when many physical parameters are unknown and, in practice, need to be estimated using available data.

We are optimistic that the techniques described in this paper may help understanding, describing and, at some point, predicting technological disasters due to dam breaking. Many qualitative and partially quantitative description of dam disasters exist in the scientific literature. For example, in [42] we find a useful report about the Brumadinho tailings dam disaster in Brazil in 2019. The paper [42] contains many references about this event, other dam disasters, and the application of mathematical models to their analysis. See, for example, [46]. We plan to apply the techniques introduced in the present paper to the support of models of the Brumadinho event in the context of the activity of CRIAB (acronym for “Conflicts, Risks and Impacts Associated with Dam” in Portugesse), an interdisciplinary research group created at the University of Campinas for prediction and mitigation of the consequences of dam disasters.

References


