Inexact Restoration approach for minimization with inexact evaluation of the objective function

Nataša Krejić [*] J. M. Martínez[†]

December 8, 2014

Abstract

A new method is introduced for minimizing a function that can be computed only inexactly, with different levels of accuracy. The challenge is to evaluate the (potentially very expensive) objective function with low accuracy as far as this does not interfere with the goal of getting high accuracy minimization at the end. For achieving this goal the problem is reformulated in terms of constrained optimization and handled with an Inexact Restoration technique. Convergence is proved and numerical experiments motivated by Electronic Structure Calculations are presented, which indicate that the new method overcomes current approaches for solving large-scale problems.

Key words: Inexact Restoration, inexact evaluations, global convergence, numerical experiments.

[*]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. Research supported by the Serbian Ministry of Education, Science, and Technological Development, Grant no. 174030

[†]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. Research supported by FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo under projects CEPID-Cemeai on Industrial Mathematics 2013/07375-0, and PT 2006/53768-0, and CNPq under projects 300933-2009-6 and 400926-2013-0.

1 Introduction

We wish to minimize a function $f(x)$ that can be computed with different levels of accuracy $\{1, 2, \ldots, N\}$. The N-th level corresponds to the maximal accuracy that we are able to achieve, although we do not have an error bound for the inexact evaluation at each level. For all $J = 1, 2, \ldots, N$ we denote by $f_J(x)$ the function value when f is computed at the J-th accuracy level. Accuracy levels admit different interpretations. If ξ is a random variable and $f(x)$ is the mathematical expectation $E[F(x, \xi)]$, we may define

$$f_J(x) = \frac{1}{J} \sum_{i=1}^{J} F(x, \xi^i),$$

where $\xi^1, \ldots, \xi^N$ is a sample of ξ. Typically F is a result of some simulation or measurement performed in a noisy environment. Another typical example comes from data fitting problems, where one seeks for the values of $x \in \mathbb{R}^n$ which minimize the function

$$f_N(x) = \sum_{i=1}^{N} F(x, y_i)$$

for a given data set $y_1, \ldots, y_N$.

In some problems involving Electronic Structure Calculations [5, 21] the objective function can be computed only by means of an iterative procedure. In this case $f_J(x)$ represents the inexact function value when one employs a maximum of J iterations for computing $f(x)$.

The problem of minimizing $f(x)$ is, for all practical effects, equivalent to

$$\text{Minimize } f_N(x). \tag{1}$$

Therefore, Problem (1) could be handled as an ordinary optimization problem but, since approximating $f(x)$ with high accuracy may be very expensive, the challenge is to solve (1) employing cheaper evaluations $f_J(x)$, with $J < N$.

For all $k = 0, 1, 2, \ldots$, we define $N_k \in \{1, \ldots, N\}$ as the accuracy level employed at iteration k. The sequence $\{N_0, N_1, N_2, \ldots\}$ is said to be the *schedule sequence* associated with the method.

The dynamics of the schedule sequence is the topic of many studies dealing with the minimization of the expected value within the framework of Sample Average Approximation (SAA) [28, 30]. The main idea is to use a variable sample size strategy, starting with a small sample and increasing the sample size during the iterative process. The case of N large but finite, as well as the case in which the sample size tends to infinity, are considered in several papers. The almost sure convergence towards a minimizer of the objective function defined as the mathematical expectation is attainable if the growth of the schedule sequence is fast enough, for example $N_k \geq \sqrt{k}$, where the schedule sequence is predetermined and tends to infinity [14]. Asymptotic convergence results for gradient methods with increasing schedule sequence and for the case in which the function is nonsmooth may be found in [29]. A refinement of SAA methods based on a careful analysis of the schedule sequence and the error tolerance of the optimization method was given in [24]. An approach that offers a quantitative measure of SAA solutions is presented in [26], where optimality functions for general stochastic programs (expected value objective and constraint functions) are considered. A method presented in [25] states an auxiliary problem that is solved before the optimization process is started. The solution of the auxiliary problem provides an efficient and strictly increasing schedule sequence.

The objective function considered in [3] and [4] comes from the SAA method assuming that N is large and finite. The problem is solved by means of a trust–region approach including a variable schedule strategy that makes it possible to decrease the schedule sequence as well as to increase it during the optimization process. The idea is to use a measure of decrease of the function value and a measure of the width of confidence interval to determine the change in the schedule sequence. These ideas were adapted to the line search framework in [19] and [20], resulting in a schedule sequence strategy that balances the precision and the decrease of the approximate objective function f_{N_k}.

In the context of data fitting problems, the dynamics of the schedule sequence is considered in [9]. Approximate gradients of the form ∇f_{N_k} are employed to solve the problem with several different dynamics for the schedule sequence. Nevertheless, all considered dynamics involve strict increase of the schedule sequence. In [7] the inexactness of the functional evaluation considered in this paper is not necessarily related to sampling strategies. In [7] (Section 10.6) a basic trust-region method is adapted to unconstrained minimization problems in which a bound on the accuracy level is available.

The idea developed in the present paper is originated in a trivial observation: The problem

$$\text{Minimize } f(x) \tag{2}$$

is equivalent to the constrained optimization problem

$$\text{Minimize } z \text{ subject to } z = f(x). \tag{3}$$

As a consequence, minimizing $f(x)$ with inexact evaluations of f corresponds to solving (3) through a sequence of non-feasible iterates. Most constrained optimization methods employ infeasible iterates and achieve feasibility at the end of the process. However, the naive application of a constrained optimization method to (3) is not admissible because the exact evaluation of $f(x)$ would not be avoided at all. Fortunately, Inexact Restoration (IR) ideas allow us to exploit the equivalence (2)–(3) without exact evaluations of f and producing, in a natural way, a non-monotone schedule sequence. Roughly speaking, the idea is to use the accuracy levels as infeasibility measures associated with the constrained problem (3).

Modern Inexact Restoration methods are developed for constrained problems and are particularly useful for problems where the objective function and/or constraints are computationally expensive and one can take advantage from separate evaluation of function and constraints. See, among others, [1, 2, 6, 11, 17, 18, 22], and [23]. A typical Inexact Restoration includes two phases - the restoration phase and the optimization phase. Within the restoration phase a new point with improved feasibility is obtained without evaluations of the objective function. During the optimality phase the objective function value at a trial point is improved with respect to the point obtained in the restoration phase. The trial point is then accepted or rejected according to a specific rule. If the trial point is rejected, a new trial point is taken closer to the restored point obtained at the end of the restoration phase. The specific rule for acceptance of the trial point couples the restoration and the optimization phases and can be formulated within line search [8], trust–region ([22], [23]) or filter methods [12, 15, 16]. In this paper we use as merit function a convex combination of the objective function and the infeasibility. The measure of infeasibility is defined taking into account specific properties of the problem and including the schedule sequence and the problem variables z and x.

This paper is organized as follows. In Section 2 we state the general Inexact Restoration Method for minimizing inexact expensive functions. Section 3 is devoted to large scale Electronic Structure Calculations involving huge nonlinear eigenvalue problems. A set of numerical examples that demonstrate the efficiency of the algorithm stated in Section 3 is presented in Section 4. Some conclusions are drawn in Section 5.

2 Inexact Restoration algorithm

In order to address problem (3) using constrained optimization tools we need to define a measure of infeasibility. The natural infeasibility measure would be $|z - f(x)|$ which, in general, cannot be computed. It is not admissible to use $|z - f_N(x)|$ either, because we do not want to compute the expensive approximation f_N at every iteration. The solution adopted in this work is to consider that the accuracy level is an additional variable of the problem and to define infeasibility with respect to a triplet (z, x, M) (instead of (z, x)) in the following way:

$$h(z, x, M) = |z - f_M(x)| + g(M), \tag{4}$$

where g is a strictly decreasing function with $g(M) > 0$ for $M < N$ and $g(N) = 0$. In this way, feasible points (z, x, M) are such that $M = N$ and $z = f_N(x)$. On the other hand, the merit function employed in the algorithm will combine optimality and feasibility measures in a classical way: For all $\theta \in [0, 1]$ this function is defined by:

$$\phi(z, x, M, \theta) = \theta z + (1 - \theta)h(z, x, M). \tag{5}$$

The Inexact Restoration algorithm described below will produce iterates $(z_k, x_k, N_k) \in \mathbb{R} \times \mathbb{R}^n \times \mathbb{N}$ that approximate the solution of problem (3). Throughout the rest of this paper $\| \cdot \|$ will denote an arbitrary norm.

Algorithm 2.1 Given $z_0 \in \mathbb{R}, x_0 \in \mathbb{R}^n$, $N_0 \in \{1, 2, \dots, N\}, r \in (0, 1), \tau, \theta_0 \in (0, 1)$, and $\beta, \gamma, \bar{\gamma} > 0$, set $k \leftarrow 0$.

Step 1. (Restoration phase)

If $N_k < N$ find $\tilde{N}_{k+1} > N_k$ and $(u_k, y_k) \in \mathbb{R}^{n+1}$ such that

$$\tilde{N}_{k+1} \leq N, \ h(u_k, y_k, \tilde{N}_{k+1}) \leq rh(z_k, x_k, N_k), \tag{6}$$

and

$$\|(u_k, y_k) - (z_k, x_k)\| \leq \beta h(z_k, x_k, N_k). \tag{7}$$

If $N_k = N$ set $\tilde{N}_{k+1} = N$ and find (u_k, y_k) such that (6) and (7) hold.

Step 2. (Updating the penalty parameter)

If

$$\phi(u_k, y_k, \tilde{N}_{k+1}, \theta_k) - \phi(z_k, x_k, N_k, \theta_k) \le \frac{1-r}{2}\Big(h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k)\Big) \tag{8}$$

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

Else compute

$$\theta_{k+1} = \frac{(1+r)\Big(h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1}\Big)}{2\Big[u_k - z_k + h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1})\Big]} \tag{9}$$

Step 3 (Optimization Phase)

Step 3.1 Choose $p_k \in \mathbb{R}^n$ and an integer valued function $N_{k+1}(\alpha)$ such that, for all $\alpha \in (0, \tau]$, we have that $N_{k+1}(\alpha) \le \tilde{N}_{k+1}$,

$$f_{N_{k+1}(\alpha)}(y_k + \alpha p_k) - f_{\tilde{N}_{k+1}}(y_k) \le -\gamma\alpha\|p_k\|^2, \tag{10}$$

and

$$h(u_k + d_k(\alpha), y_k + \alpha p_k, N_{k+1}(\alpha)) \le h(u_k, y_k, \tilde{N}_{k+1}) + \bar{\gamma}\alpha^2\|p_k\|^2, \tag{11}$$

where

$$d_k(\alpha) = [-f_{\tilde{N}_{k+1}}(y_k) + f_{N_{k+1}(\alpha)}(y_k + \alpha p_k)]. \tag{12}$$

Step 3.2. Find $\alpha_k \in (0, 1]$ as large as possible such that (10) and (11) hold for $\alpha = \alpha_k$ and, in addition,

$$\begin{aligned} &\phi(u_k + d_k(\alpha_k), y_k + \alpha_k p_k, N_{k+1}(\alpha_k), \theta_{k+1}) \\ &\le \phi(z_k, x_k, N_k, \theta_{k+1}) + \frac{1-r}{2}\Bigg(h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k)\Bigg). \end{aligned} \tag{13}$$

Step 4. Set $x_{k+1} = y_k + \alpha_k p_k$, $z_{k+1} = u_k + d_k(\alpha_k)$, $N_{k+1} = N_{k+1}(\alpha_k)$, $k \leftarrow k+1$ and go to Step 1.

Before proving the theoretical properties of Algorithm 2.1 we wish to give some hints on the reasons why we believe that the IR aproach represents a suitable way of handling the schedule sequence $\{N_k\}$. A key point is formula (13) where we find the new iterate $(z_{k+1}, x_{k+1}, N_{k+1})$ in such a way that the merit function combining feasibility and optimality decreases. Recall that feasibility, in this case, corresponds to accuracy in the evaluation of the objective function $f(x)$. A lower objective function with better accuracy is all we need but both objectives may not be achieved simultaneously, so that a merit function, which gives different weights to the goals, must be employed. We will see that the penalty parameter weighs the accuracy and the optimality objectives in a fair way, since it is non-increasing and bounded away from zero. The way in which the penalty parameter is modified is endogenous to the algorithm and depends on quantities internally computed, not relying on arbitrarily chosen parameters. Note that at Step 1 of the algorithm one increases accuracy (feasibility) and, at Step 2, one computes the penalty parameter as the maximal value for which the intermediate iterate with improved accuracy (y_k) is better than the current iterate in terms of the merit function. This would be obviously true in the case of a null penalty parameter but, fortunately, we will be able to prove that the property will remain to be true for a sequence $\{\theta_k\}$ well separated from zero. After obtaining a more accurate evaluation at Step 1 and deciding the penalty parameter at Step 2, the IR approach decreases the approximate objective function trying first less expensive function evaluations. In fact, note that the requirement $N_{k+1}(\alpha) \leq \tilde{N}_{k+1}$ must hold for α small enough (smaller than τ) but at Step 3.2 we try first $\alpha_k = 1$ so we aim to accelerate the whole process getting enough decrease of the merit function without increasing evaluation work. (We may take $N_{k+1}(\alpha) << \tilde{N}_{k+1}$ when $\alpha \in (\tau, 1]$.) As a whole, the algorithm increases the precision at Step 1 and tries to save computer evaluation time at Step 3.2 when α is not very small. This process represents a cautious and conservative way of increasing precision with occasional savings of computer time by means of less expensive evaluations. Of course, one needs to end up with the maximal allowable precision in order to guarantee minimization of the best approximation to f. The description of Algorithm 2.1, in principle, does not guarantee that the precision N is achieved. However, we will prove that, with proper assumptions, in spite of the cheaper trial function evaluations, the algorithm eventually arrives to the maximal precision. Theoretical results will lead us to that conclusion.

The inexact Restoration method assumes that the feasibility step can always be performed and thus the following assumption is necessary.

Assumption A1 For all $k = 0, 1, 2, \ldots$, it is possible to compute sequences $\{N_k\}$ and $\{(u_k, y_k)\}$ such that (6) and (7) are satisfied.

Let us discuss the plausibility of Assumption A1. To show that satisfying (6) and (7) is possible note that (6) could be satisfied trivially taking $\tilde{N}_{k+1} = N$, $y_k = x_k$, and $u_k = f_N(x_k)$, since, in that case, we would have that $h(u_k, y_k \tilde{N}_{k+1}) = 0$ and $\|(u_k, y_k) - (z_k, x_k)\|_\infty = |z_k - f_N(x_k)|$. So, by the equivalence of norms, there exists $c > 0$ such that $\|(u_k, y_k) - (z_k, x_k)\| \leq c|z_k - f_N(x_k)|$. Therefore, (7) will hold if

$$|z_k - f_N(x_k)| \leq (\beta/c)[|z_k - f_{N_k}(x_k)| + g(N_k)].$$

This inequality is satisfied if

$$|f_{N_k}(x_k) - f_N(x_k)| \leq (\beta/c - 1)[|z_k - f_{N_k}(x_k)| + g(N_k)],$$

and, thus, with $z_k = f_{N_k}(x_k)$, the following is a sufficient condition for the fulfillment of (7):

$$|f_{N_k}(x_k) - f_N(x_k)| \leq (\beta/c - 1)g(N_k),$$

The last inequality essentially says that the error in the N_k-approximation of the function f should be bounded by a (possibly big) multiple of $g(N_k)$. This property is generally satisfied if the function g is a reasonable measure of accuracy, as expected in applications

The penalty parameter is updated at Step 2. The next Lemma states that the sequence of penalty parameters $\{\theta_k\}$ is non-increasing and bounded away from zero.

Lemma 2.1. *Assume that A1 is satisfied. Then Steps 1 and 2 are well defined. Moreover, the sequence $\{\theta_k\}$ is positive and non-increasing, the inequality*

$$\phi(u_k, y_k, \tilde{N}_{k+1}, \theta_{k+1}) - \phi(z_k, x_k, N_k, \theta_{k+1}) \leq \frac{1-r}{2}\Big(h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k)\Big)$$

holds and there exists $\theta^* > 0$ such that

$$\lim_{k \to \infty} \theta_k = \theta^*.$$

Proof. The proof is similar to the one of Lemma 4.1 in [6]. If (8) is satisfied then $\theta_{k+1} = \theta_k$. Otherwise, since (8) does not hold, we have that

$$\theta_k(u_k - z_k) + (1-\theta_k)[h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k)] > \frac{1-r}{2}[h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k)].$$

Moreover, by (6),

$$h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k) < 0.$$

Therefore,

$$\theta_k > \frac{(1+r)\left(h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1}\right)}{2\left[u_k - z_k + h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1})\right]} := \theta_{k+1},$$

and thus θ_k is a non-increasing sequence. Let us now prove that θ_{k+1} given by (9) is bounded away from zero. Let $c_{norm} > 0$ be such that $|a| \leq c_{norm}\|(a,b)\|$ for all $a, b \in \mathbb{R}$. Then, by (6) and (7),

$$\begin{aligned} \frac{1}{\theta_{k+1}} &= \frac{2\left[u_k - z_k + h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1})\right]}{(1+r)\left(h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1})\right)} \\ &\leq \frac{2}{1+r}\left[\frac{|u_k - z_k|}{h(z_k, x_k, N_k) - h(u_k, y_k, \tilde{N}_{k+1})} + 1\right] \\ &\leq \frac{2}{1+r}\left[\frac{c_{norm}\beta}{1-r} + 1\right]. \end{aligned}$$

This completes the proof. □

The fact that the penalty parameter is bounded away from zero is crucial for algorithmic efficiency. In that case the merit function is guaranteed to give a positive weight to optimality with respect to feasibility, making it possible, in practice, to find feasible and optimal points and not merely feasible ones.

The optimality step consists of two parts. Step 3.1 is always well defined as the choice $p_k = 0$ and $N_{k+1}(\alpha) = \tilde{N}_{k+1}$ ensures that (11) holds for an arbitrary τ. Let us now show that the second part of Step 3 is also well defined.

Lemma 2.2. *Assume that A1 holds, (z_k, x_k, N_k) and $(y_k, u_k, \tilde{N}_{k+1})$ are generated through steps 1 - 3.1. Then there exists $\alpha^* > 0$ such that for all $k = 0, 1, 2, \dots$, (13) is fulfilled with $\alpha_k \geq \alpha^*$.*

Proof. By (12), (10)-(11), and Lemma 2.1,

$$\begin{aligned}
& \phi(u_k + d_k(\alpha), y_k + \alpha_k p_k, N_{k+1}(\alpha), \theta_{k+1}) - \phi(z_k, x_k, N_k, \theta_{k+1}) \\
= \ & \phi(u_k + d_k(\alpha), y_k + \alpha_k p_k, N_{k+1}(\alpha), \theta_{k+1}) - \phi(u_k, y_k, \tilde{N}_{k+1}, \theta_{k+1}) \\
+ \ & \phi(u_k, y_k, \tilde{N}_{k+1}, \theta_{k+1}) - \phi(z_k, x_k, N_k, \theta_{k+1}) \\
\leq \ & -\theta_{k+1}\gamma\alpha\|p_k\|^2 + (1-\theta_{k+1})\bar{\gamma}\alpha^2\|p_k\|^2 + \frac{1-r}{2}\Big(h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k)\Big)
\end{aligned}$$

for all $\alpha \in (0, \tau]$. Thus, in order to get (13), we need to establish the conditions for α_k under which

$$-\theta_{k+1}\gamma\alpha_k + (1-\theta_{k+1})\bar{\gamma}\alpha_k^2 \leq 0, \tag{14}$$

which is equivalent to

$$\alpha_k \leq \frac{\theta_{k+1}\gamma}{\bar{\gamma}(1-\theta_{k+1})}.$$

Now, as $\theta_{k+1} \geq \theta^* > 0$, the inequality (13) is fulfilled for

$$\alpha_k \leq \bar{\alpha} := \min\{\tau, \frac{\theta^*\gamma}{\bar{\gamma}(1-\theta^*)}\}.$$

So, the statement is true for

$$\alpha^* = \frac{1}{2}\bar{\alpha}.$$

□

The following theorem states that the schedule sequence eventually becomes stationary and, thus, we solve the problem with full precision at the final stages of the iterative procedure. The proof is conceptually similar to the proof of Theorem 4.1 in [6].

Theorem 2.1. *Assume that A1 is satisfied, f_N is Lipschitz continuous, the functions f_M are continuous for $M \leq N$, and the sequences $\{z_k\} \subset \mathbb{R}$, $\{x_k\} \subset \mathbb{R}^n$ generated by Algorithm 2.1 are bounded. Then, there exists $k_0 \in \mathbb{N}$ such that $N_k = \tilde{N}_{k+1} = N$ for $k \geq k_0$. Furthermore,* $\lim_{k\to\infty}\|p_k\| = 0$.

Proof. The boundedness of $\{z_k\}$, $\{x_k\}$, and N_k, and the continuity of f_M imply that $h(z_k, x_k, N_k)$ is bounded.

Let us define the sequence $\rho_k = (1-\theta_k)/\theta_k$. As $\theta_k \geq \theta^*$ and θ_k is nonincreasing, we have that ρ_k is nondecreasing and $\rho_k \leq 1/\theta^* - 1$. Therefore

$$\sum_{k=0}^{\infty}(\rho_{k+1} - \rho_k) = \lim_{k\to\infty}\rho_{k+1} - \rho_0 < \infty. \tag{15}$$

As $h(z_k, x_k, N_k)$ is bounded, (15) yields

$$\sum_{k=0}^{\infty}(\rho_{k+1}-\rho_k)h(z_k, x_k, N_k) < \infty.$$

By condition (13) we have

$$\phi(z_{k+1}, x_{k+1}, N_{k+1}, \theta_{k+1}) \leq \phi(z_k, x_k, N_k, \theta_{k+1}) + \frac{1-r}{2}(h(u_k, y_k, \tilde{N}_{k+1}) - h(z_k, x_k, N_k))$$

and, taking (6) into account,

$$\phi(z_{k+1}, x_{k+1}, N_{k+1}, \theta_{k+1}) \leq \phi(z_k, x_k, N_k, \theta_{k+1}) - \frac{(1-r)^2}{2}h(z_k, x_k, N_k).$$

The last inequality can be stated as

$$z_{k+1} + \rho_{k+1}h(z_{k+1}, x_{k+1}, N_{k+1}) \leq z_k + \rho_{k+1}h(z_k, x_k, N_k) - \frac{(1-r)^2}{2\theta_{k+1}}h(z_k, x_k, N_k).$$

Since $\theta_{k+1} < 1$, adding and subtracting $\rho_k h(z_k, x_k, N_k)$, we obtain:

$$\begin{aligned} & z_{k+1} + \rho_{k+1}h(z_{k+1}, x_{k+1}, N_{k+1}) \\ \leq\ & z_k + \rho_k h(z_k, x_k, N_k) + (\rho_{k+1}-\rho_k)h(z_k, x_k, N_k) - \frac{(1-r)^2}{2}h(z_k, x_k, N_k). \end{aligned} \tag{16}$$

Therefore,

$$\begin{aligned} & \sum_{j=0}^{k}\frac{(1-r)^2}{2}h(z_j, x_j, N_j) \\ \leq\ & z_0 + \rho_0 h(z_0, x_0, N_0) - z_{k+1} + \rho_{k+1}h(z_{k+1}, x_{k+1}, N_{k+1}) + \sum_{j=0}^{k}(\rho_{j+1}-\rho_j)h(z_j, x_j, N_j). \end{aligned}$$

Now, by (15) and the boundedness of z_k, ρ_{k+1}, and $h(z_{k+1}, x_{k+1}, N_{k+1})$, we have that

$$\lim_{k\to\infty}\sum_{j=0}^{k}\frac{(1-r)^2}{2}h(z_j, x_j, N_j) < \infty$$

and

$$\lim_{k\to\infty}h(z_k, x_k, N_k) = 0.$$

Thus, by (7)

$$\lim_{k\to\infty} u_k - z_k = 0 \text{ and } \lim_{k\to\infty} y_k - x_k = 0,$$

and, by (6),

$$\lim_{k\to\infty} h(u_k, y_k, \tilde{N}_{k+1}) = 0.$$

The definition of h implies

$$\lim_{k\to\infty} g(N_k) = \lim_{k\to\infty} g(\tilde{N}_{k+1}) = 0.$$

Then, eventually the schedule sequences N_k and $\tilde{N}_{k+1}$ become stationary i.e.

$$N_k = \tilde{N}_{k+1} = N, \; k \geq k_0.$$

To prove the rest of the statement let us denote by L the Lipschitz constant of f_N. Then,

$$f_N(x_{k+1}) - f_N(x_k) \leq f_N(x_{k+1}) - f_N(y_k) + f_N(y_k) - f_N(x_k),$$

and for $k \geq k_0$, by (10) and (7), we have:

$$\begin{aligned} f_N(x_{k+1}) - f_N(x_k) &\leq -\gamma\alpha_k\|p_k\|^2 + L\|y_k - x_k\| \\ &\leq -\gamma\alpha_k\|p_k\|^2 + L\beta h(z_k, x_k, N_k). \end{aligned}$$

Therefore

$$f_N(x_{k+1}) \leq f_N(x_{k_0}) - \gamma \sum_{j=k_0}^{k} \alpha_j\|p_j\|^2 + L\beta \sum_{j=k_0}^{k} h(z_j, x_j, N_j).$$

Since $\sum_{j=k_0}^{k} h(z_j, x_j, N_j) < \infty$ we obtain $\gamma \sum_{j=k_0}^{k} \alpha_j\|p_j\|^2 < \infty$ for an arbitrary $k \geq k_0$.Thus

$$\lim_{k\to\infty} \alpha_k\|p_k\|^2 = 0.$$

But (13) is satisfied for $\alpha_k \geq \alpha^*$ and thus

$$\lim_{k\to\infty} \|p_k\| = 0.$$

□

Remark

Theorem 2.1 confirms that, eventually, the algorithm addresses the most accurate approximation of $f(x)$. The fact that, in addition, the algorithm minimizes this approximation depends on the choice of p_k. A gradient-related choice such that $\|p_k\| \geq c\|\nabla f_N(x^k)\|$ for some $c > 0$ and k large enough obviously garantees that $\|\nabla f_N(x^k)\|$ tends to zero, and, thus, that every limit point of $\{x^k\}$ is stationary.

3 Application to Electronic Structure Calculations

Given fixed nuclei coordinates, an electronic structure calculation consists of finding the wave functions from which the spatial electronic distribution of the system can be derived. These wave functions are the solutions of the Schrödinger equation [13, 21].

The practical solution of the Schrödinger equation is computationally very demanding. Therefore, simplifications are made leading to more tractable mathematical problems. The best-known approach leads to the "Hartree-Fock" formulation, where one needs to minimize an energy function $\bar{E}(C) \equiv E(CC^T)$ in which, due to Pauli's Exclusion Principle, the columns of the $K \times nocc$ matrix C must be orthonormal:

$$\text{Minimize } E(CC^T) \text{ subject to } C^T C = I_{nocc \times nocc}, C \in \mathbb{R}^{K \times nocc}. \tag{17}$$

In (17), C is said to be the "coefficients matrix", $nocc$ is the number of occupied orbitals (pairs of electrons) and the basis has K elements. Thus, defining the "density matrix" P by $P = CC^T$, problem (17) can be written as:

$$\text{Minimize } E(P) \text{ subject to } P = P^T, P^2 = P, Trace(P) = nocc, P \in \mathbb{R}^{K \times K}. \tag{18}$$

The most popular approach for solving (18) is the Fixed Point Self-Consistent Field (SCF) Method. This is an iterative algorithm that, at each iteration, given the current P_c, minimizes the linear approximation of $E(P)$:

$$\begin{gathered}\text{Minimize } Trace(\nabla E(P_c)P) \\ \text{subject to } P = P^T, P^2 = P, Trace(P) = nocc, P \in \mathbb{R}^{K \times K}.\end{gathered} \tag{19}$$

The solution P_{new} of (19) is the projection matrix on the subspace generated by the eigenvectors associated with the $nocc$ smallest eigenvalues of $\nabla E(P_c)$. Unfortunately, the spectral decomposition of $\nabla E(P_c)$ cannot be computed if $nocc$ and K are very large, even in the case that $\nabla E(P_c)$ is sparse. However, P_{new} has, approximately, the same sparsity structure as $\nabla E(P_c)$ if the gap between the eigenvalues $nocc$ and $nocc + 1$ of $\nabla E(P_c)$ is big enough. This is the case of many relevant molecular systems. This property allows one to address problem (19) by means of the following conceptual algorithm [5]:

Algorithm 3.1

Let $x_0 \in \mathbb{R}$ be an educated guess for the Fermi level $(\lambda_{nocc} + \lambda_{nocc+1})/2$, where $\lambda_1 \leq \lambda_{nocc} \leq \lambda_{nocc+1} \leq \dots \lambda_K$ are the eigenvalues of $A \equiv \nabla E(P_c)$. Initialize $k \leftarrow 0$.

Step 1 Using Gershgorin bounds [10], compute $\gamma > 0$ such that all the eigenvalues of $\gamma(-A - x_k I)$ are between $-1/2$ and $1/2$. Compute $B_{start} = \gamma(-A - x_k I) + \frac{1}{2} I_{K \times K}$. Then, all the eigenvalues of B_{start} are between 0 and 1.

Step 2 Starting with the initial approximation B_{start}, and employing the projected gradient method of [5], obtain B_k as the solution of the optimization problem

$$\text{Minimize } \|B^2 - B\|_F^2 \text{ subject to } B = B^T, \tag{20}$$

with the additional constraint that the sparsity pattern of B coincides with the sparsity pattern of $\nabla E(P_c)$.

Step 3 If $Trace(B_k) = nocc$, stop the execution of Algorithm 3.1, define $P_{new} = B_k$, and stop. Otherwise, obtain a new guess x_{k+1}, set $k \leftarrow k+1$ and go to Step 1. (In [5] problem (20) is solved with high precision and x_{k+1} is computed by means of a root-finding bisection procedure.)

The properties of Algorithm 3.1 have been analyzed in [5], where it was shown that, suitably implemented, it is useful for practical large-scale electronic calculations. Writing $f(x_k) = (Trace(B_k) - nocc)^2$, we note that the objective of Algorithm 3.1 is to minimize f and the evaluation of this function is expensive, because it is based on the application of an iterative (projected gradient) method to solve (20). In practice, each time we want to solve (20) we prescribe a maximum number of iterations N_k for the projected gradient method. In [5] a constant (with respect to k) and rather large N_k (generally $N_k = 1000$) was used at each call of the projected gradient method. If $x_k \in (\lambda_{nocc}, \lambda_{nocc+1})$ the projected gradient method finds, under mild assumptions, a matrix B_k such that $Trace(B_k) \approx nocc$. However, if an iterate x_k does not belong to $(\lambda_{nocc}, \lambda_{nocc+1})$, a lot of projected gradient iterations may be spent before convergence to a solution of (20) occurs, perhaps completing the maximum N_k. This motivates the employment of algorithms that, like the one presented in this paper, avoid to compute many iterations to obtain the solution of (20) when x_k is not close to the solution. Although many heuristic procedures could be suggested, the Inexact Restoration approach provides a natural way to choose the schedule sequence N_k, as explained in Section 2.

Summing up, problem (19) consists of minimizing $f(x)$, defined by

$$f(x) = (Trace[B(x)] - nocc)^2,$$

where the approximate value of $f(x)$, as well as the approximate density matrix $B(x)$, with level of accuracy $N_k \in \{1, \ldots, N\}$ is obtained by means of the following algorithm:

Algorithm 3.2

Assume that $\varepsilon > 0$ and $N_k \in \{1, \ldots, N\}$.

Step 1 Compute $B_{start} = c(-A - xI) + \frac{1}{2} I_{K \times K}$ in such a way that all the eigenvalues of B_{start} are between 0 and 1.

Step 2 Consider Problem (20) (with the sparsity pattern constraint) and obtain an approximate solution $B(x)$ as the result of applying the Projected Gradient method with convergence stopping criterion ε on the ∞-norm of the projected gradient and a maximum of N_k projected gradient iterations.

Step 3 Define $f_{N_k}(x) = (Trace[B(x)] - nocc)^2$.

Here we propose to compute the approximate solution of (19) (with $f_{N_k}(x)$ given by Algorithm 3.2) by means of Algorithm 2.1 with the following specifications:

1. We used $N = 1000$.

2. The accuracy measure $g(M)$ is given by

$$g(M) = \frac{N - M}{M}.$$

3. As in [5], x_0 is computed in the following way:

$$x_0 = \frac{[K - (nocc + 0.5)]a + (nocc + 0.5)b}{K}$$

   where a and b are lower and upper bounds for the eigenvalues of $\nabla E(P_c)$ computed using the Gershgorin Theorem [10].

4. We choose $N_0 = 10$ and $z_0 = f_{N_0}(x_0)$.

5. We choose $r = 0.5$, $\beta = 10^3$, $\gamma = 10^{-4}$, $\bar{\gamma} = 100$, $\theta_0 = 0.9$, and $\tau = 10^{-2}$.

6. At Step 1 we choose $\tilde{N}_{k+1} = 2N_k$, $y_k = x_k$, and $u_k = f_{\tilde{N}_{k+1}}(y_k)$.

7. We choose $z_0 = f_{N_0}(x_0)$.

8. Using the fact that $Trace[B(x)]$ is non-decreasing as a function of x and that we wish to find x such that $Trace[B(x)] = nocc$, we keep approximate upper and lower bounds of the solution (recall that the function is computed only approximately) and choose a first trial for p_k based on safeguarded regula-falsi and bisection. If this direction satisfies (10) and (11) for $\alpha = 1$ we adopt this choice for p_k and set $N_{k+1}(1) = \tilde{N}_{k+1}/2$ [1]. Otherwise we choose $p_k = -\nabla f_{\tilde{N}_{k+1}}(x_k)$ and $N_{k+1}(1) = \tilde{N}_{k+1}$.

9. The value of α_k that satisfies (13) is obtained by backtracking (with factor 0.5) using $\alpha = 1$ as first trial. If $\alpha < 1$ we define $N_{k+1}(\alpha) = \tilde{N}_{k+1}$.

10. The algorithm was stopped declaring success when the matrix $B(x)$ computed at some call of Algorithm 3.2 satisfies

$$\|B(x)^2 - B(x)\| \le 10^{-8} \text{ and } |Trace[B(x)] - nocc| \le 0.4 \qquad (21)$$

    or when the iteration index k exceeds 1000.

The properties of Algorithm 2.1 were discussed in Section 2 assuming that Assumption 1 is satisfied. Let us show that this assumption is fulfilled

[1]The alternative choice $N_{k+1}(1) = \tilde{N}_{k+1}/2.5$ was also tested with similar computational results.

with the specifications given above. As the projected gradient method is convergent for problem (20) we can assume that there exists $C > 0$ such that

$$|f_M(x)| \leq C, \ M \in \{1, 2, \ldots, n\}. \tag{22}$$

The function $g(M)$ is decreasing and the choice $\tilde{N}_{k+1} = 2N_k$ implies

$$g(\tilde{N}_{k+1}) \leq rg(N_k),$$

for $\tilde{N}_{k+1} < N$ while for $\tilde{N}_{k+1} = N$ we have $g(\tilde{N}_{k+1}) = 0$. So (6) is fulfilled with the special choice $u_k = f_{\tilde{N}_{k+1}}(x_k)$, $z_k = f_{N_k}(x_k)$ and $y_k = x_k$. Let us now show that (7) is also valid. If $N_k = N$ then $\tilde{N}_{k+1} = N$ and

$$\|(u_k, y_k) - (z_k, x_k)\| = |u_k - z_k| = |f_N(x_k) - f_N(x_k)| = 0$$

and (7) holds. For $N_k \leq N - 1$ we have

$$\|(u_k, y_k) - (z_k, x_k)\| = |u_k - z_k| = |f_{\tilde{N}_{k+1}}(x_k) - f_{N_k}(x_k)| \leq 2C,$$

due to (22). Then (7) holds for

$$\beta \geq \frac{2C}{N-1}$$

as $g(N_k) \leq (N-1)^{-1}$ for $N_k \leq N - 1$.

4 Experiments

The objective of this section is to show the plausibility of Algorithm 2.1 for solving problem (19), by means of the minimization of $f(x)$, whose approximate computation is described in Algorithm 3.2. Algorithm 2.1 was implemented with the specifications given at the end of Section 3.

The codes were written in Fortran, employing double precision and the following computer environment: Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz with 4GB of RAM memory.

For all the problems we report the number of occupied states *nocc*, the number of the elements of the basis K, the "pseudo-gap" between the eigenvalues *nocc* and *nocc* + 1 of $\nabla E(P_c)$, the accuracy on the idempotency requirement $\|B^2 - B\|$, the number of iterations performed by Algorithm 2.1, and the CPU time. We ran Algorithm 2.1 as described above against a version of this algorithm in which $N_k = N = 1000$ for all k, which corresponds essentially to the algorithm introduced in [5]. The numbers corresponding to the two algorithms are separated by a vertical bar "|". For example, by writing " 3 | 5" under the caption "Iterations" we mean that Algorithm 2.1 (with schedule sequence N_k computed by the IR strategy) employed 3 iterations to solve the problem, whereas the algorithm with constant $N_k \equiv N$ employed 5 iterations.

In the set of problems identified as "Family A" the matrix $\nabla E(P_c)$ is diagonal and its elements have been randomly generated between 0 and 1. This is a trivial problem whose objective is to compare the algorithms in the case that the solution has *exactly* the sparsity pattern of $\nabla E(P_c)$ and, so, no error is introduced when we project the gradient of $\|P^2 - P\|_F^2$ onto the subspace of matrices with the sparse structure of $\nabla E(P_c)$. The analytic solution of problem (19) in this case is the diagonal matrix with 1's in the places in which $\nabla E(P_c)$ has its smallest (diagonal) elements, and 0's otherwise.

The pseudo-gap is defined in the following way: In the case of Family A it is the gap between eigenvalues *nocc* and *nocc* + 1. In the other families it is a number that we add to the first N diagonal entries of the (random) matrix under consideration. In these families we do not know the value of the true gap but we roughly know that the true gap is larger, the larger is the pseudo-gap.

In the set of problems identified as "Family B" the matrix $\nabla E(P_c)$ is tridiagonal, its elements are randomly generated between 0 and 1 and then modified by the pseudo-gap. In the case of this family and Family C we do not know the sparsity pattern of the projection matrix on the subspace generated by the eigenvectors associated with the *nocc* smallest eigenvalues. However, we know that when the pseudo-gap is big enough the sparsity pattern of the projection matrix is similar to the sparsity pattern of $\nabla E(P_c)$ [5]. .

In "Family C" the matrix $\nabla E(P_c)$ has band structure with "diag" diagonals. Inside the diagonal band 80 % of the elements are null. For computing matrix products we consider that the matrix elements inside the band are non-zeros. The dimensions K and *nocc* considered in Family C correspond to a molecular structure with 6,000 molecules of water under semiempirical handling of electrons [5].

The numerical results are given in Table 1. The trace of the final matrix $B(x)$ has not been reported in this table because it coincided with the value of *nocc* (as desired) up to a very high precision (better than 10^{-8}) in all the cases.

In Family A we observe that even for extremely small gaps the numerical solution is very accurate for very large values of K and *nocc* and the computer time employed by Algorithm 2.1 is consistently smaller than the one employed by the competitor. In Family B the same tendency is maintained, except in Problem 7, in which the algorithm with constant schedule sequence was faster than the IR algorithm. In this case the initial x was a solution, the winner algorithm recognized it after allowing a maximum of 1000 iterations, but the IR algorithm did not because it allowed ony 10 iterations in the first restoration phase.

In Families B and C both algorithms failed for values of the pseudo-gap smaller than $0.95 p_g$, where p_g is the pseudo-gap reported in Table 1. This is not unexpected because in the cases of small pseudo-gap the solution of the true Density matrix does not exhibit the sparsity pattern of the data matrix $\nabla E(P_c)$. As a consequence, both algorithms tend to converge to local mimimizers of $\|P^2 - P\|_F^2$ with the sparsity pattern of the data, but this local minimizer is not an idempotent matrix as desired. Problems with these characteristics should be solved by different methods than the ones analyzed here and, many times, are unsolvable with the present technology.

In Family C, the advantage of considering the IR algorithm instead of the constant-schedule approach is quite evident. In this case the cost of evaluating the objective function is very high and, so, savings of computer time are dramatic when we employ the IR strategy. A peculiarity of these problems is that it is much more expensive to evaluate $f(x)$ when x is not the solution than when it is. The reason is that, when x is far from the solution a small gap appears that does not correspond to the true gap between eigenvalues *nocc* and *nocc*+1. As a consequence, convergence of the projected gradient method is painful in this case. For example, in Problem 12 the cost of a typical projected gradient iterations is 7.6 seconds. The algorithm with constant $N_k = N = 1000$ performs 1000 iterations (and takes 7600 seconds) in the restoration phase without achieving convergence. However, the first trial x in the optimization phase is found to be a solution of the problem after a small number of projected gradient iterations. **Algorithm 3.1** has a similar behavior except that the restoration phase involves only 20 iterations. It is clearly crucial to adopt an strategy that saves computer time at early stages of the minimization process.

It is interesting to analyze the cases in which the IR algorithm converges in less iterations than the constant-schedule approach (Problems 1 and 4).

In both cases, at some iteration, both algorithms arrive to the same value of y. Moreover, both algorithms compute the same trial shift $y + p$. In the case of the algorithm with constant schedule the objective function decreases enough at $y+p$ and, so, $y+p$ is accepted as the new iterate. In the case of the IR algorithm, the trace of $B(y + p)$ is computed with reduced accuracy and, by Algorithm 2.1, this accuracy affects the acceptance of the trial point. It turns out that $y + p$ is rejected and backtracking is necessary. On the other hand, the trace of $B(y+p)$ overestimates the target *nocc* whereas both the trace of $B(y)$ and the trace at the backtracked point underestimate the target. For example, in Problem 4, the target *nocc* is $2, 500, 000$, at the second iteration the trace of $B(y)$ was $2, 499, 909$, and the trace of $B(y + p)$ was $2, 500, 063$. However, the trace at the backtracked trial was $2, 499, 996$. Thus, the backtracked point was better than $y + p$ even from the point of view of functional value, and this causes the slight difference in the number of iterations that is observed in Table 1.

Family A: $\nabla E(P_c)$ Diagonal						
Problem	nocc	K	Pseudo-gap	$\|B^2 - B\|$	Iterations	CPU Time (seconds)
1	500,000	1,000,000	8.E-8	7.E-16 | 7.E-16	3 | 5	5.51 | 7.57
2	250,000	1,000,000	4.E-6	4.E-14 | 1.E-8	4 | 1	5.41 | 5.80
3	5,000,000	10,000,000	4.E-9	4.E-15 | 5.E-16	9 | 9	100.4 | 110.5
4	2,500,000	10,000,000	2.E-7	1.E-9 | 1.E-9	3 | 4	52.1 | 86.7
Family B: $\nabla E(P_c)$ Tridiagonal						
Problem	nocc	K	Pseudo-gap	$\|B^2 - B\|$	Iterations	CPU Time
5	500,000	1,000,000	3.25	9.E-7 | 9.E-7	4 | 4	22.0 | 172.2
6	250,000	1,000,000	3.25	5.E-7 | 9.E-7	7 | 7	38.0 | 154.0
7	5,000,000	10,000,000	3.35	9.E-7 | 9.E-11	1 | 0	51.9 | 6.89
8	2,500,000	10,000,000	3.35	6.E-8 | 6.E-8	4 | 4	121.1 | 1082.8
Family C: $\nabla E(P_c)$ Band Sparse						
Problem	nocc	K	Pseudo-gap	$\|B^2 - B\|$	Iterations	CPU Time
9. diags = 21	24,000	36,000	12.5	2.E-9 | 2.E-9	7 | 7	3.7 | 124.2
10, diags = 41	24,000	36,000	17.2	2.E-10 | 2.E-10	1 | 1	11.5 | 284.1
11, diags = 81	24,000	36,000	30.0	6.E-11 | 6.E-11	1 | 1	33.9 | 1057.6
12, diags = 161	24,000	36,000	60.0	2.E-9 | 2.E-9	1 | 1	231.6 | 7753.2

Table 1: Behavior of Algorithm 2.1 on Families A, B, and C.

5 Conclusions

We considered the problem of minimizing a function $f(x)$ whose exact evaluation is impossible or very expensive. The main idea is to reduce the original (unconstrained) problem to a constrained problem in which inexactness in the evaluation of $f(x)$ corresponds to infeasibility of the approximation. The Inexact Restoration approach is an attractive tool to deal with this situation since it allows us to control infeasibility without really evaluating the constraints. As a consequence of the application of Inexact Restoration we are able to define a schedule sequence that defines different degrees of accuracy in the evaluation of f. We have proved that the process is theoretically consistent employing suitable adaptations of inexact restoration arguments. In order to test the reliability of the ideas presented in this paper we applied the new method to a problem motivated by electronic structure calculations, in which the evaluation of the objective function comes from an iterative minimization method. The numerical results indicate that the idea of computing scheduling sequences by means of inexact restoration is promising. Accordingly, future research will include the application of this technology to problems in which lack of accuracy comes from different sources and the consideration of the case in which the original problem is constrained.

Acknowledgment We are grateful to the anonymous referees whose constructive remarks helped us to improve this paper.

References

[1] R. Andreani, S. L. Castro, J. L. Chela, A. Friedlander, and S. A. Santos, An inexact-restoration method for nonlinear bilevel programming problems, Computational Optimization and Applications 43 (2009), 307–328.

[2] N. Banihashemi and C. Y. Kaya, Inexact Restoration for Euler Discretization of Box-Constrained Optimal Control Problems, Journal of Optimization Theory and Applications 156 (2013), pp. 726-760.

[3] F. Bastin, Trust-Region Algorithms for Nonlinear Stochastic Programming and Mixed Logit Models, PhD thesis, University of Namur, Belgium, (2004).

[4] F. Bastin, C. Cirillo, and Ph. L. Toint, An adaptive Monte Carlo algorithm for computing mixed logit estimators, Computational Management Science 3 (2006), 55-79.

[5] E. G. Birgin, J. M. Martínez, L. Martínez, and G. B. Rocha, Sparse projected-gradient method as a linear-scaling low-memory alternative to

diagonalization in self-consistent field electronic structure calculations, Journal of Chemical Theory and Computation 9 (2013), 1043-1051.

[6] L. F. Bueno, A. Friedlander, J. M. Martínez, and F. N. C. Sobral, Inexact Restoration method for Derivative-Free Optimization with smooth constraints, SIAM Journal on Optimization 23-2 (2013), 1189-1231

[7] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*, SIAM, Philadelphia, USA, 2000.

[8] A. Fischer and A. Friedlander, A new line search inexact restoration approach for nonlinear programming, Computational Optimization and Applications 46 (2010), 333–346.

[9] M. P. Friedlander, M. Schmidt, Hybrid deterministic-stochastic methods for data fitting, SIAM Journal on Scientific Computing 34 (2012), 1380-1405.

[10] G. H. Golub and Ch. Van Loan *Matrix Computations*, Johns Hopkins: Baltimore, MA, 1996.

[11] M. A. Gomes-Ruggiero, J. M. Martínez, and S. A. Santos, Spectral projected gradient method with inexact restoration for minimization with nonconvex constraints, SIAM Journal on Scientific Computing 31 (2009), 1628–1652.

[12] C. C. Gonzaga, E. Karas, and M. Vanti, A globally convergent filter method for Nonlinear Programming, SIAM Journal on Optimization 14 (2003), 646-669.

[13] T. Helgaker, P. Jorgensen, and J. Olsen, Molecular Electronic-Structure Theory; John Wiley & Sons: New York, 2000, 433-502.

[14] T. Homem-de-Mello, Variable-Sample Methods for Stochastic Optimization, ACM Transactions on Modeling and Computer Simulation 13 (2003), 108-133.

[15] E. W. Karas, C. C. Gonzaga, and A. A. Ribeiro, Local convergence of filter methods for equality constrained nonlinear programming, Optimization 59 (2010), 1153–1171.

[16] E. W. Karas, E. A. Pilotta, and A. A. Ribeiro, Numerical comparison of merit function with filter criterion in inexact restoration algorithms using Hard-Spheres Problems, Computational Optimization and Applications 44 (2009), 427–441.

[17] C. Y. Kaya, Inexact Restoration for Runge-Kutta discretization of Optimal Control problems, SIAM Journal on Numerical Analysis 48 (2010), 1492–1517.

[18] C. Y. Kaya and J. M. Martínez, Euler discretization and Inexact Restoration for Optimal Control, Journal of Optimization Theory and Applications 134 (2007), 191–206.

[19] N. Krejić and N. Krklec, Variable sample size methods for unconstrained optimization, Journal of Computational and Applied Mathematics 245 (2013), 213-231.

[20] N. Krejić and N. Krklec–Jerinkić, Nonmonotone line search methods with variable sample size, Numerical Algorithms (to appear).

[21] C. Le Bris, Computational chemistry from the perspective of numerical analysis. Acta Numerica 14 (2005), 363-444.

[22] J. M. Martínez, Inexact Restoration method with Lagrangian tangent decrease and new merit function for nonlinear programming, Journal of Optimization Theory and Applications 111 (2001), 39-58.

[23] J. M. Martínez and E. A. Pilotta, Inexact restoration algorithms for constrained optimization, Journal of Optimization Theory and Applications 104 (2000), 135-163.

[24] R. Pasupathy, On Choosing Parameters in Retrospective-Approximation Algorithms for Stochastic Root Finding and Simulation Optimization Operations Research 58 (2010), 889-901.

[25] E. Polak and J. O. Royset, Eficient sample sizes in stochastic nonlinear programing, Journal of Computational and Applied Mathematics 217 (2008), 301-310.

[26] J. O. Royset, Optimality functions in stochastic programming, Mathematical Programming 135 (2012), 293-321.

[27] A. Shapiro, D. Dentcheva, and A. Ruszczynski, Lectures on Stochastic Programming Modeling and Theory, SIAM, 2009.

[28] A. Shapiro and A. Ruszczynski, Stochastic Programming, Handbook in Operational Research and Management Science Vol. 10 Elsevier, 2003, 353-425.

[29] A. Shapiro and Y. Wardi, Convergence Analysis of Stochastic Algorithms, Mathematics of Operations Research 21 (1996) 615-628.

[30] J. C. Spall, Introduction to Stochastic Search and Optimization, Wiley-Interscience serises in discrete mathematics, New Jersey, 2003.

[31] Y. Wardi, Stochastic Algorithms with Armijo Stepsizes for Minimization of Functions, Journal of Optimization Theory and Applications 64 (1990), 399-417.