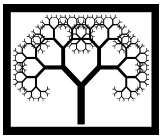
Paper 63



Proceedings of the Eleventh International Conference
on Computational Structures Technology,
B.H.V. Topping, (Editor),
Civil-Comp Press, Stirlingshire, Scotland

Searching for Multiple Minima of Bound Constrained Optimization Problems using Derivative Free Optimization Techniques

U.M. García Palomares
Departamento de Enxeñaría Telemática
Escola de Enxeñaría de Telecomunicación, Universidade de Vigo, Spain

Abstract

This paper proposes the use of derivative free optimization techniques in an algorithm that locates either a *good* single minimizer or several minimizers with close function values of unconstrained and bound constrained minimization problems. The algorithm assumes that the objective function is Lipschitzean with a constant provided by the user. This is an artifice that identifies sub domains where the objective function cannot reach a *good* minimum value, and search on that sub domain is immediately abandoned. The algorithm has been tested on problems with a small number of variables with encouraging results. We believe this is an innovative approach, but more numerical tests should be carried out before reaching conclusive statements.

Keywords: derivative free, bound constraint, multi processing, global minimum.

Notation

We use a rather standard notation with certain peculiarities: R^n is the *n-th* dimensional Euclidean spaces; vectors in R^n are denoted by small Latin letters with components $x^k, k = 1, \cdots, n$; $e = (1, \ldots, 1)$ is a vectors of ones; e_k is a vector of null values, except that $e_k^k = 1$; scalars are denoted by small Greek letters; R^+ is the set of non-negative scalars; a σ-function $\sigma(\cdot) : R^+ \rightarrow R^+$ is any non decreasing function with $\lim_{\tau \rightarrow 0} \sigma(\tau)/\tau = 0$. The index i is used as an iteration number; P_i is a set of points at the i-th iteration and the subindex ij refers to the j-th element of P_i.

1 Introduction

This paper deals with algorithms for solving the bound constrained optimization problem:

$$\text{(BCOP)} \qquad \underset{x\in F}{\text{minimize}}\ f(x), \quad F=\{x\in R^n : s\leq x\leq t\}, \tag{1}$$

where, for the sake of formality, we assume that the upper and lower bounds have different value, that is, $s^k < t^k, k = 1,\ldots,n$; otherwise x^k is not further considered as a variable of the problem. Bounds can take infinite values and unconstrained optimization is a particular case of the BCOP. The algorithm solves (1) by solving a sequence of the same optimization model defined in sub domains. At the i-th iteration we are given a finite set of *prospective* solutions $P_i = \{x_{i1},\ldots,x_{ip(i)}\}$. For each $x_{ij}\in P_i$ we define new bounds (s_{ij}, t_{ij}) and solve

$$\text{BCOP}_{ij} \qquad \underset{x\in F_{ij}}{\text{minimize}}\ f(x), \quad F_{ij}=\{x\in F : s_{ij}\leq x\leq t_{ij}\}. \tag{2}$$

For the convergence theory special attention must be given to the *active* set

$$\partial F_{ij}(\delta) = \big\{x\in F_{ij} : (\exists k)\big(s_{ij}^k+\delta > x^k \text{ OR } t_{ij}^k-\delta < x^k\big)\big\}, \tag{3}$$

where $\delta > 0$ is a given parameter, that may also change from iteration to iteration.

Efficient Derivative Free Optimization (DFO) techniques exist for solving (1) when derivative information on $f(\cdot)$ are neither existent nor readily computable. Literature on DFO is rapidly expanding, but we only cite those recent works [2, 5, 7, 9] that are linked to our line of research. This paper uses DFO techniques for solving the sequence of problems BCOP$_{ij}$ given above (2), but our aim departs from the cited works. We do not try to find a *local* minimum with the least number of function evaluations possible; we instead want to find a *good* set of local minima of a function that might have multiple local minimizers. We show how to formulate the subproblems (2) to achieve convergence to different minima.

We recall that, under suitable conditions, it is easy to identify in a finite time, whether $x\in F$ is a local minimum. On the contrary, a global minimum has no mathematical characterization. To declare that x is a *good* minimizer we should amply search the feasible set, which means that $f(\cdot)$ must be evaluated a significant number of times. The idea of our algorithm is to identify subdomains F_{ij} with interior solutions. These are as well, solutions to the original problem (1). From a practical point of view the algorithm declares that x_* is a *good* minimizer when it is a local minimum and when $f(x_*)$ is not too much larger than the best minimum found. The final asset on the relevance of this minimizer is left to the practitioner.

It is worth mentioning that the algorithm uses strategies to find a global solution to each BCOP$_ij$. We include the non monotone behaviour proposed in [6, 5], linear extrapolation techniques [8, 3, 5], and evaluate the function on random points. We could also try simulated annealing or any other techniques, but we have not implemented them yet. Anyway, a global minimum cannot be ensured for the reasons just given in the above paragraph, and there is a tradeoff between the use of resources and finding a good minimum.

At the i-th iteration, the algorithm works with a finite *population* $P_i = \{x_{i1}, \ldots, x_{ip(i)}\}$ and for a fixed $\Delta > \delta$ problem (2) is explicitly formulated as

$$\underset{x \in F_{ij}}{\text{minimize}}\ f(x), \quad F_{ij} = \{x \in F : x_{ij} - \Delta e \leq x \leq x_{ij} + \Delta e\}, \tag{4}$$

that is, the search is confined to a neighborhood of x_{ij}. Other models may be appropriate for more general problems. The solution of 4 is truncated with an accuracy that is more demanding as the iteration advances. But the *truncated* solution is always a Discrete Quasi Minimal Point (DQMP) of (4); and the convergence theory developed in [5] can be applied. We claim that $\exists\{Q_i \subseteq P_i\}$ converging to a set $Q_* = \{x_{*1}, \ldots, x_{*q*}\}$ of $q*$ local minima. DQMPs were introduced in [1] for unconstrained minimization and adapted in [5] to the BCOP. It is shown in those papers that local minima are limit points of sequences of DQMPs. It is appropriate to recall that

Definition D1. *Given a set of directions $D = \{d_k \in R^n : k = 1, \cdots, q\}$, a stepsize $\tau > 0$, an upperbound $\varphi \geq f(x)$, and a σ-function $\sigma(\tau) : R^+ \to R^+$, let $D^+ = \{y \in R^n : y = \sum_{k=1}^{q} \alpha_k d_k, d_k \in D, \alpha_k \geq 0, k = 1, \cdots, q\}$.*

A point $x \in F_{ij}$ is denoted a DQMP if

$$\begin{array}{l} D^+ = R^n, \text{ and} \\ \big[d \in D\big] \Rightarrow \langle \begin{array}{l} (x + \tau d) \notin F_{ij}, \text{ or} \\ f(x + \tau d) > \varphi - \sigma(\tau). \end{array} \end{array} \tag{5}$$

It is obvious to observe that local minima, including the global one, satisfy (5) for sufficiently small τ.

Each DQMP ($x_{ij} \in P_i$) generates a new DQMP ($x_{i+1,j} \in F_{ij}$) using DFO techniques or any gradient like method. We assume that $f(\cdot)$ is Lipschitzean and an estimate of the Lipschitz constant is known, that is,

$$|f(x) - f(y)| \leq \kappa ||x - y||, \quad \forall (x \in F, y \in F),$$

where κ is estimated by the user. If κ is unknown, we can assume that it is very large. The use of a Lipschitz estimate κ makes possible to eliminate x_{ij} from P_i; specifically, x_{ij} is discarded when

$$f_i^m = \min_{1 \leq j \leq p} \big(f(x_{ij})\big) < f(x_{ij}) - \kappa\Delta, \tag{6}$$

because $f(x_{ij}) - \kappa\Delta \leq f(x)$ for all $x \in F_{ij} : f(x) < f(x_{ij})$. In other words, the algorithm has detected that no point in F_{ij} has a function value below f_i^m and no search on this neighborhood is needed. We also claim that the algorithm works even if the set F is discrete and can be described as a grid of a regular size; nonetheless to avoid distraction from the main ideas, we only glimpse onto this, and refer the reader to [4].

The scheme enclosed in Figure 1 is a rough description of the algorithm. As usual, we dropped the iteration index i and include a parameter $\epsilon > 0$ to stop the algorithm

in finite time. It is worth noticing that this algorithm embodies many variants and can be easily generalized, at least conceptually, for solving more general optimization problems. We have decided to focus on BCOP, mainly to detect appropriate values for the parameters involved and to find out the pros and cons to our approach. It has been argued that the algorithm performance for solving BCOP$_{ij}$ is not affected for any constant $\mu \in [0.3\ 0.7]$. We have decided to generate at each iteration a random μ in that interval. Other parameters $(|P_0|, \Delta, \bar{\tau}, \beta)$ may have a significant influence. Intuitively, the larger their values, the slower the process, and the larger the number of function evaluations. We now ask the reader to turn the attention to Figure 1.

At each outer iteration (WHILE loop) the algorithm solves $|P|$ BCOP models, with an accuracy that is more demanding as the algorithm proceeds (Step 3). In other words, these subproblems generate approximate local solutions, namely, DQMPs. These points however, are discarded when (6) holds (Step 2.1). Step 2.3 is necessary because a point located in ∂F_j is not necessarily a DQMP of the main problem (1).

Parallelism can be invoked as suggested in [5], or it can be incorporated in Step 1. Albeit this is an important issue, we are only reporting single processor performance and the members of P are taken randomly in the FOR loop. For the sake of completeness Figure 1 also includes the basic scheme used to solve (4) and Figure 2 describes a function that constructs D satisfying $D^+ = R^n$.

The rest of this paper is organized as follows: Section 2 is short, yet important because convergence is shown as a proposition derived from previous works [6, 5]. Numerical tests on two small problems are reported in Section 3 and Section 4 ends the paper with conclusions and future search.

2 Convergence

We have presented the algorithm in such a way that we can easily claim convergence from the material given in [5]. Each particular sequence $\{x_{ij}\}_{i \in Z}, j = 1, \ldots, p_*$ has limit points that are local solutions to problem (1). We now state the convergence proposition and sketch its proof.

Proposition 1 **A1.** *The sequence $\{P_i\}$ remains in a compact.*

A2. *$f(\cdot)$ is bounded below in F, and strictly differentiable at limit points.*

There exists a subsequence $\{Q_i \subseteq P_i\}$ that converges to local minima of (1), provided that **A1** *and* **A2** *hold.*

Proof. We are assuming implicitly that at all iterations, the set of directions of search D satisfies that $D^+ = R^n$. Besides, the updating of φ satisfies **A4** in [5, Section 3]. Therefore, all assumptions required for convergence in [5] hold and P_i is a set of DQMPs for some subproblem with bounded variables. Note also that by the way the algorithm is constructed, the limit points $\{x_{*1}, \ldots, x_{*p(*)}\}$ are also DQMPs, and consequently local minima of (1). ∎

Input: $P, \Delta, \tau = \bar{\tau}, \bar{\epsilon} > \epsilon$

Loop: WHILE $(\tau > \epsilon)$
Step 1. Compute $f^m = \min_{x \in P} f(x)$
Step 2. FOR $j = 1, \ldots, p$
Step 2.1 IF $(f(x_j) > f^m + \kappa\Delta)$
Remove x_j from P
CONTINUE (next $x_j \in P$)
END IF
Step 2.2 Find a DQMP x' (stop solving (4) when $\tau < \bar{\epsilon}$)
Step 2.3 IF $(x' \in \partial F_j)$ Update $\tau = \bar{\tau}$
Step 2.4 Replace x by x' in P
END FOR
Step 3. Generate random $\mu \in [0.3\ 0.7]$, $\bar{\epsilon} = \mu\bar{\epsilon}$
END WHILE

Output: $P, f(P)$

Basic scheme for solving BCOP in Step 2.2

Input: $\varphi, x_j, x^* = x_j, \tau^* = \tau = \bar{\tau}, \bar{\epsilon}$

WHILE $(\tau > \bar{\epsilon})$
Construct $D : D^+ = R^n$
IF (5) holds
Generate random $\mu \in [0.3\ 0.7]$, $\tau = \mu\tau$
IF $\big((\tau < \bar{\epsilon})$ AND $(f(x^*) < f_j)\big)$
$x_j = x^*, \tau = \tau^*, \varphi = f(x^*)$
END IF
ELSE (5) does not hold
Pick $y \in F_j : f(y) \leq \varphi - \sigma(\tau)$
Generate a random $z \in F_j$
IF $(f(y) < f(z))$
$x_j = y$ ELSE $x_j = z$
END IF
IF $\big(f_j < f(x^*)\big)$
$x^* = x_j, \tau^* = \tau$
END IF
Generate a random $\beta \in [0\ 0.9]$
$\varphi = (1 - \beta)f_j + \beta\varphi$
END IF
END WHILE

Output: x_j, f_j

Figure 1: Algorithm framework

3 Numerical tests

Before we carry out the tests, it is advisable to analyze suitable parameters values; for instance, a large value of Δ could force $F_i = F$ and the algorithm reduces to

Formal arguments: $(\bar{s}, x, \bar{t}, \delta) : \bar{s} \leq x \leq \bar{t}, \delta > 0$
Let $A = \{k : x^k < \bar{s}^k + \delta,$ OR $x^k > \bar{t}^k - \delta\}$ be the index set of δ-Active constraints. Let $S = \{x \in R^n : x^k = 0, k \in A\}$ and let B be a *minimum* unitary basis for S.
Return $D = \{\pm B, \pm e_k, k \in A\}$

Figure 2: Function that builds $D^+ = R^n$

a multi starting approach. We simply apply several iterations of the DFO algoritm to each element in P_i and solve (4) up to the required accuracy. A large value for δ will have a tremendous impact on the algorithm performance: On one hand, only canonical vectors $e_1, \ldots, e_n$ will be used throughout the entire procedure. It is well know that these directions are not in general the best ones to choose from. On the other hand, the updating $\tau = \bar{\tau}$ will always be done after solving (4), which will increase the number of function evaluations. We decided to carry out some tests to try to determine appropriate values for these parameters.

We started with the Branin-Hoo unconstrained optimization problem in 2 variables (x, y):

$$f(x, y) = \left((y - 1.275(x/\pi)^2 + 5x/\pi - 6\right)^2 + 10(1 - 1/8\pi)cos(x) + 10$$

This problem has 4 global minima. A population of 20 elements were randomly distributed in the box $-5 \leq x \leq 20; -5 \leq y \leq 20$. We set the following input to the problem: $\Delta = 2, \delta = 0.1, \bar{\tau} = 1, \epsilon = 10^{-4}, \bar{\epsilon} = 10^{-2}, \kappa = 20$, and $\varphi = 1.2 f_j$, for each $x_j \in P$.

For the solution of BCOP$_{ij}$ we used the code developed in MATLAB by [5]. The algorithm ended up with 16 elements converging to the global minima. We repeated the test with other reasonable parameter values and found similar results.

With the same set of parameter values we attempted the solution to the 5-dimensional Shekel's function

$$f(x) = -\sum_{j=1}^{30} \frac{1}{c^j + \sum_{k=1}^{5} (x^k - A_{jk})^2}.$$

The values for (c, A) can be seen in [6, Table 5]. They also report that this function has at least 12 local minima. The key factor here was the size of the initial population. The rate of success, measured as the number of times that the global minima is reported in 100 runs, is more than 40% with an initial population of at least 50 points.

Although these experiments cannot be considered as conclusive, the results are highly encouraging. It does seem that acceptable results can be obtained with a reasonable choice of parameter values.

4 Conclusions and future search

Global optimization is not necessarily a goal for practitioners, mainly because they are aware that several characteristics of their system could not be incorporated into the optimization model. This paper presents an algorithm that finds multiple local minima of a strictly differentiable function subjected to bounds on the variables. The algorithm works with a set of initial points -a population- and DFO techniques are used to generate DQMPs of models with bounded variables defined in subdomains. The algorithm uses the fact that a local minimizer is as well a DQMP.

The algorithm is tested on 2 small problems with a set of *magic* numbers for the different parameters involved. Although this is by no means conclusive, it seems that, except for the initial size of the population, the results not sensible to an ample range of the parameter values.

There are two obvious routes that this work can tackle: on the one hand, the solution of models with linear constraints using DFO techniques. On the other hand, the use of derivative information for generating DQMPs and for solving more general optimization problems.

Acknowledgment

This search was supported by grant CALM (TEC2010-21405-C02-01).

References

[1] Ian D. Coope and Christopher J. Price. Frame based methods for unconstrained optimization. *Optimization theory and Applications*, 107(2):261–274, 2000.

[2] Ana Luisa Custódio, H Rocha, and Luís Nunes Vicente. Incorporating minimum frobenius norm models in direct search. *Computational Optimization and Applications*, 46:265–278, 2010.

[3] M.A. Diniz-Ehrhardt, José Mario Martínez, and Marcos Raydán. A derivative-free nonmonotone line-search technique for unconstrained optimization. *Journal of Computational and Applied Mathematics*, 219:383–397, 2008.

[4] Ubaldo Manuel García Palomares, Enrique Costa Montenegro, Rafael Asorey Cacheda, and Francisco Javier González Castaño. Adapting derivative free optimization methods to engineering models with discrete variables. *Optimization and Engineering*, September 2011.

[5] Ubaldo Manuel García Palomares, Ildemaro García Urrea, and Pedro Rodríguez Hernández. On sequential and parallel non-monotone derivative free algorithms for box constrained optimization. *Optimization Methods & Software*, accepted.

[6] Ubaldo Manuel García Palomares, Francisco Javier González Castaño, and Juan Carlos Burguillo Rial. A combined global & local search (CGLS) approach to global optimization. *Journal of Global Optimization*, 34:409–426, 2006.

[7] Serge Gratton, Philippe L. Toint, and Anke Tröltzsch. An active-set trust-region method for derivative-free nonlinear bound-constrained optimization. *Optimization Methods and Software*, 26(4–5):873–894, 2011.

[8] Stefano Lucidi and Marco Sciandrone. On the global convergence of derivative-free methods for unconstrained optimization. *SIAM journal on optimization*, 13(1):97–116, 2002.

[9] Mike J. D. Powell. The bobyqa algorithm for bound constrained optimization without derivatives. Technical report, Department of Applied Mathematics and Theoretical Physics, Cambridge, August 2009.