European Journal of Operational Research 000 (2017) 1–12

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Continuous Optimization

Efficient solution of quadratically constrained quadratic subproblems within the mesh adaptive direct search algorithm☆

Nadir Amaioua[a], Charles Audet[a], Andrew R. Conn[b], Sébastien Le Digabel[a,*]

[a] GERAD and Département de mathématiques et génie industriel, École Polytechnique de Montréal, C.P. 6079, Succ. Centre-ville, Montréal, Québec H3C 3A7, Canada

[b] Mathematical Sciences, IBM T J Watson Research Center, 1101 Kitchawan Rd. Yorktown Heights NY 10598, USA

ARTICLE INFO

Article history:
Received 9 December 2016
Accepted 25 October 2017
Available online xxx

Keywords:
Nonlinear programming
Derivative-free optimization
Quadratic programming
Trust-region subproblem
Mesh adaptive direct search

ABSTRACT

The mesh adaptive direct search algorithm (MADS) is an iterative method for constrained blackbox optimization problems. One of the optional MADS features is a versatile search step in which quadratic models are built leading to a series of quadratically constrained quadratic subproblems. This work explores different algorithms that exploit the structure of the quadratic models: the first one applies an l_1-exact penalty function, the second uses an augmented Lagrangian and the third one combines the former two, resulting in a new algorithm. It is notable that this latter approach is uniquely suitable for quadratically constrained quadratic problems. These methods are implemented within the NOMAD software package and their impact are assessed through computational experiments on 65 analytical test problems and 4 simulation-based engineering applications.

1. Introduction

We consider the following constrained optimization problem:

$$\begin{array}{lll} \min\limits_{x\in\mathcal{X}} & f(x) & \\ \text{subject to} & c_j(x) \le 0, & j \in [\![1,m]\!] \end{array} \tag{1}$$

where m is a positive integer, $\mathcal{X}$ is a subset of $\mathbb{R}^n$, f and $(c_j)_{j\in[\![1,m]\!]}$ are real-valued functions, possibly evaluated by a computer simulation, seen as a blackbox with the following characteristics: function evaluations take a long time to compute, the simulation may fail for some input values, the derivatives are not available and their approximations may be unreliable, and/or too expensive. The set $\mathcal{X}$ is often defined by bound on the variables.

Derivative-free optimization (DFO, (Conn, Scheinberg, & Vicente, 2009)) algorithms are designed to handle this type of problem. DFO methods do not use or try to approximate derivatives of the problems. Instead, they either rely on a direct search approach which uses a discretization of the solution space and generates directions to test trial points, or they use model-based methods by constructing polynomial or other approximations to mimic the functions over some specified trust-region or as a surrogate for a line-search method. Recent surveys on blackbox and derivative-free optimization appear in Audet (2014), Boukouvala, Misener, and Floudas (2016).

The present work focuses on the mesh adaptive direct search algorithm (MADS) (Audet & Dennis, Jr., 2006) with quadratic models (Conn & Le Digabel, 2013). MADS principally relies on a pair of steps, called the search and the poll, to explore the space of variables and a third step to update its parameters. Both the search and the poll are complementary: the search allows local and global exploration while the poll is local and ensures convergence. We consider a model-based approach in the search step that has no impact on the theoretical convergence analysis of MADS, but improves its practical performance. At each iteration k, the search builds local quadratic models near the current iterate x^k for the objective function f and for each of the m inequality constraints. This leads to a quadratically constrained quadratic subproblem over an l_∞ norm trust-region:

$$\begin{array}{lll} \min\limits_{x\in\mathcal{X}} & f^k(x) & \\ \text{subject to} & c_j^k \le 0, & j \in [\![1,m]\!] \\ & \left\| x - x^k \right\|_\infty \le \Delta^k & \end{array} \tag{2}$$

where the scalar $\Delta^k \ge 0$ defines the trust-region, f^k is the quadratic model of the objective function near the current iterate x^k and

☆ This work is supported by the NSERC CRD RDCPJ 490744-15 grant in collaboration with Hydro-Québec and Rio Tinto.

* Corresponding author.
E-mail addresses: nadir.amaioua@gerad.ca (N. Amaioua), sebastien.le.digabel@gerad.ca (S. Le Digabel).
URL: http://www.gerad.ca/Charles.Audet (C. Audet), http://researcher.watson.ibm.com/researcher/view.php?person=us-arconn (A.R. Conn), http://www.gerad.ca/Sebastien.Le.Digabel (S. Le Digabel)

https://doi.org/10.1016/j.ejor.2017.10.058
0377-2217/

the $(c_j^k)_{j\in[\![1,m]\!]}$ are the quadratic models of the m inequality constraints. We prefer an l_∞ trust region because it has the same orientation as a box determined by simple bounds.

In Conn and Le Digabel (2013), to solve Problem (1), a series of subproblems of the type (2) are created and each of them is solved with a new instance of MADS. The paper concludes by stating that the quadratic structure of the subproblems is not exploited and that a blackbox optimization solver is certainly not the appropriate choice to solve a quadratically constrained quadratic problem. The main purpose of this paper is to test dedicated algorithms to solve Problem (2). We choose two widely known methods from the literature: the l_1-exact penalty function and the augmented Lagrangian methods. A new methodological approach combining the strengths of both methods is introduced and called the l_1-augmented Lagrangian: it uses an l_1 penalty term instead of the squared one used in the standard augmented Lagrangian.

This paper is organized as follows. Section 2 describes the MADS algorithm and in particular, its mechanisms to build and use quadratic models. The section also summarizes several methods from the literature handling quadratic subproblems. Section 3 presents the l_1-exact penalty function algorithm and Section 4 gives a description of the augmented Lagrangian method. Section 5 introduces the new l_1-augmented Lagrangian method and describes implementation choices. Section 6 compares the results of the three algorithms on a set of 61 analytical and 4 simulation-based test problems and Section 7 concludes with recommendations.

2. Literature review

The MADS algorithm is an iterative method that uses a discretization of the solution space, called the mesh, to select and evaluate new trial points. Each iteration of MADS consists of two main steps: the search and poll, followed by a parameter update step.

The poll and update steps are critical to the convergence proof of MADS (Audet & Dennis, Jr., 2006), but the search is optional and more flexible: it can be omitted or defined by the user. A frequently used search strategy (Conn & Le Digabel, 2013) is to automatically construct quadratic models to try and find a promising trial point. If the search succeeds, i.e. the selected trial point improves upon the current iterate, then this trial point becomes the new iterate and the poll step is skipped. Moreover, the search direction can also be exploited elsewhere, for example to prioritize poll choices (see below). On the contrary, if the search fails, the poll step becomes mandatory. Other types of search step such as VNS (Audet, Béchard, & Le Digabel, 2008) and surrogate-based (Audet, Kokkolaras, Le Digabel, & Talgorn, 2017; Booker et al., 1999) are not the topic of the present paper. In this paper, we use quadratic models in the search step since our understanding, and in fact our computational experience, suggest that it is likely to be the best, or at least one of the best, approaches.

The poll is used to choose mesh points near the current iterate and to evaluate their objective and constraint values. On the one hand, if the poll fails to find a better solution, the update step will reduce the mesh size (the parameter that scales the space discretization) and the poll size (maximum distance allowed between a trial point and the current iterate) in order to concentrate near the current iterate. On the other hand, once a better solution is found, the poll step terminates and the update step increases the mesh size. The diagram in Fig. 1 represents a description of the MADS algorithm with a search step based on quadratic models.

The quadratic models constructed in the search step are also used in the poll step: the poll produces a list of trial mesh points, and instead of sending them directly to be evaluated, quadratic models of the objective function and of the constraints are used to sort the trial points, so that the most promising ones are evaluated first. This approach is used in conjunction with the opportunistic strategy: as soon as a better solution than the current iterate is found, the poll step stops without executing the simulation at the remaining trial points. Using quadratic models in both the search and poll steps greatly improves the MADS performance, as reported in Conn and Le Digabel (2013).

MADS relies on a cache structure, which stores all the evaluated points, to select an interpolation set in order to build the models. In the search step, the interpolation set is constructed by selecting all the points from the cache that are inside the ball centered around the current iterate with a radius equal to twice the poll size parameter. In the poll step, quadratic models are used to order trial points, and interpolation points from the cache are selected within a ball centered around the current iterate with a radius $2r$, where r is the smallest radius of the ball centered in the current iterate and containing all the trial points (Conn & Le Digabel, 2013).

At each iteration of MADS, a quadratic subproblem of the form (2) is constructed and solved within the search step. The resulting solution is projected on the mesh to provide a starting point that satisfies the convergence requirements of MADS. The subproblem is similar to those arising in trust-region methods called trust-region subproblems. Since the early eighties, many algorithms have been developed specifically to solve this kind of quadratic problems. The Moré and Sorenson algorithm (MS) (Moré & Sorensen, 1983) is one of the first algorithms used specifically for quadratic problems subject to an ellipsoidal constraint. It is based on solving the optimality conditions via the Newton algorithm with backtracking. The downside of this algorithm is that it uses Cholesky factorizations that become expensive for large matrices, which is not an issue in the present research since no large matrices are involved. Later, the generalized Lanczos trust-region algorithm (GLTR) (Gould, Lucidi, & Toint, 1999) was implemented by using the MS algorithm on Krylov subspaces. However, GLTR could not handle hard cases (see chapter 7 in Conn, Gould, & Toint (2000)) of the trust-region subproblems. The same principle is applied by the sequential subspace method (SSM) (Hager, 2001) that creates four dimensional subspaces instead of using the Krylov subspaces. Even if the SSM algorithm handles the hard case of the trust-region subproblem, it still solves quadratic problems over a sphere. The Gould-Robinson and Thorne algorithm (Gould, Robinson, & Thorne, 2010) improved the MS algorithm by using some high dimensional polynomial approximations that allow the Newton method to converge in fewer iterations. Another algorithm, that treats specifically quadratic problems over a convex quadratic constraint, is the Fortin-Rendl and Wolkowicz algorithm (Fortin & Wolkowicz, 2004; Rendl & Wolkowicz, 1997) which rewrites the problem into an eigenvalue subproblem that can be handled by the Newton method combined with Armijo–Goldstein conditions. This algorithm is suitable for large-scale matrix problems.

All the algorithms above are used for a quadratic objective over a unique constraint defining the trust-region. In our case, Problem (2) is additionally constrained by m quadratic constraints and, recently, one method was developed specifically for this kind of problems using an extension of the Fortin-Rendl and Wolkowicz algorithm (Pong & Wolkowicz, 2014). Solving Problem (2) can also be done by nonlinear optimization tools such as exact penalty functions and augmented Lagrangians. These latter two approaches are discussed further in the next two sections.

3. The l_1-exact penalty function (l_1EPF algorithm)

The l_1-exact penalty function starts by transforming Problem (2) into the following bound-constrained problem: