Generating Box-Constrained Optimization Problems

FRANCISCO FACCHINEI
Università di Roma "La Sapienza"
and
JOAQUIM JÚDICE and JOÃO SOARES
Universidade de Coimbra

We present a method for generating box-constrained nonlinear programming test problems. The technique allows the user to control some properties of the generated test problems that are known to influence the behavior of algorithms for their solution. A corresponding set of Fortran 77 routines is described in a companion algorithm (774).

Categories and Subject Descriptors: G.1.6 [**Numerical Analysis**]: Optimization; G.4 [**Mathematics of Computing**]: Mathematical Software—*certification and testing*; *verification*

General Terms: Algorithms, Performance, Verification

Additional Key Words and Phrases: Nonlinear programming test problems, optimization, test problems generation

1. INTRODUCTION

We consider box-constrained optimization problems of the form

$$\min_{x \in K} f(x), \tag{1}$$

where

$$K = \{x \in \mathbb{R}^n : l_i \leq x_i \leq u_i, i = 1, \ldots, n\}. \tag{2}$$

We allow the bounds to be finite or infinite and assume that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable in an open set containing K. In particular, if all the bounds are infinite then Problem (1) reduces to an unconstrained optimization problem.

Authors' addresses: F. Facchinei, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Rome, Italy; J. Júdice and J. Soares, Departamento de Matemática, Universidade de Coimbra, Coimbra, 3000, Portugal.

Box-constrained problems are probably the simplest kind of constrained nonlinear programming problems, and they often arise in practice. Actually, most "unconstrained" problems encountered in applications are only meaningful if the variables belong to some prefixed range of values and should therefore be considered as box-constrained problems.

It is well accepted that the best way to assess the practical efficiency of an algorithm to solve Problem (1) and its robustness is to test it at least on a large set of test problems with different characteristics. Unfortunately, such a set of test problems does not yet exist, especially if one considers large-scale problems. For example, even in the huge collection CUTE [Bongartz et al. 1995], there are only eight nonlinear (and nonquadratic) box-constrained problems with more than 500 variables. Furthermore, it is often difficult to know the characteristics of available test problems. On the contrary, there exists a great number of test problems for unconstrained optimization (see, for example, Averick et al. [1992], Bongartz et al. [1995], Brown and Bartholomew-Biggs [1989], Hock and Schittkowski [1981], Moré [1981], and Nash and Nocedal [1991]) whose characteristics are fairly well understood. Thus, it seems natural to use these unconstrained test problems to generate, in a simple way, box-constrained test problems. Furthermore, one would like to be able to control as many factors as possible among those which are known to affect the performance of box-constrained optimization codes (e.g., number of constraints, shape of the box, number of active constraints at the solution, degeneracy).

Existing generating techniques [Bartels and Mahdavi-Amiri 1986; Lootsma 1985; Rosen and Suzuki 1965; Schittkowski 1980; 1992] aim at generating *general* constrained optimization problems. Again, all the relevant characteristics can be user controlled. However, it is not clear if they are well suited for large-scale problems, and they have never been used to test box-constrained optimization algorithms, possibly because of the lack of available software.

In this work we present a simple method that can be used to generate box-constrained test problems with known characteristics by suitably modifying a given unconstrained problem. A corresponding set of Fortran 77 routines is described in a companion algorithm (774).

2. THE GENERATION TECHNIQUE

A vector $\bar{x} \in K$ is said to be a stationary point for Problem (1) if it satisfies

$$\left\{ \begin{array}{ll} l_i = \bar{x}_i & \Rightarrow \nabla f_i(\bar{x}) \geq 0 \\ l_i < \bar{x}_i < u_i & \Rightarrow \nabla f_i(\bar{x}) = 0 \\ \bar{x}_i = u_i & \Rightarrow \nabla f_i(\bar{x}) \leq 0, \end{array} \right\} \tag{3}$$

where $\nabla f(x)$ is the gradient vector of f at x. Every local, or global, minimum point of Problem (1) is a stationary point. Our aim in this section is to describe a simple method to build a box-constrained problem, with a

known local minimum point and characteristics, starting from a given unconstrained problem

$$\min_{x \in \mathbb{R}^n} g(x), \tag{4}$$

where g is a twice continuously differentiable function.

Let $\bar{x}$ be a local minimum of this unconstrained problem. The box-constrained problem we will generate has the same solution $\bar{x}$. We start by choosing an arbitrary partition of the set of indices $\{1, \ldots, n\}$ into three subsets L, F, and U. The generated problem will be such that $\bar{x}_i = l_i$ for $i \in L$, $l_i < \bar{x} < u_i$, for $i \in F$, and $\bar{x}_i = u_i$ for $i \in U$, so that F, L, and U are the sets of indices of the variables that are free, at a lower bound and at an upper bound at $\bar{x}$, respectively.

To achieve this, we choose the vectors l and u to satisfy the following relationships

$$\begin{aligned} l_L = \bar{x}_L < u_L \\ l_F < \bar{x}_F < u_F \\ l_U < \bar{x}_U = u_U, \end{aligned} \tag{5}$$

where the subscript is meant to identify the corresponding components. We assume that if some component of u_L or u_F is equal to $+\infty$, the corresponding variable has no upper bound, while if some component of l_U or l_F is equal to $-\infty$, the corresponding variable has no lower bound.

Now consider the objective function

$$f(x) = g(x) + \sum_{i \in L} h_i(x_i) - \sum_{i \in U} h_i(x_i), \tag{6}$$

where $h_i : \mathbb{R} \to \mathbb{R}$, $i \in L \cup U$, are twice continuously differentiable nondecreasing functions. The gradient of f is given by

$$\nabla f_i(x) = \begin{cases} \nabla g_i(x) + h'_i(x_i) & \text{if } i \in L \\ \nabla g_i(x) & \text{if } i \in F, i = 1, \ldots n \\ \nabla g_i(x) - h'_i(x_i) & \text{if } i \in U \end{cases} \tag{7}$$

where h'_i indicates the first derivative of h_i. Since the functions h_i, $i \in L \cup U$, are nondecreasing, it follows from (5) and (6) that $\bar{x}$ is a local minimum of the box-constrained optimization problem

$$\min_{l \le x \le u} f(x). \tag{8}$$

If $\bar{x}$ is just a stationary point of (4), since $h'_i(\bar{x}) \ge 0$ for $i \in L \cup U$, then $\bar{x}$ is a stationary point of problem (8) as well.

Thus, we have generated a box-constrained problem with the same solution as the underlying unconstrained problem. By suitably choosing the sets L, F, and U we can control the number of active bounds at $\bar{x}$ and decide exactly which variables are at their lower bounds, which are at their upper bounds, and which are free at $\bar{x}$. The choice of the vectors l and u determines the total number of constraints of the problem and the "shape" of the feasible region. The Lagrange multipliers associated with the constraints $l_L \leq x_L$ and $x_U \leq u_U$ are $h'_i(\bar{x}_i)$ for $i \in L \cup U$, while the remaining multipliers are 0. Hence, by suitably choosing h_i (see below), we also have complete control over the degeneracy of problem (8) at $\bar{x}$. Furthermore, it is important to note that the Hessian of f will differ from that of g in at most the diagonal elements, so that any sparsity pattern of the Hessian of g is maintained in the Hessian of f.

Note that all these features are known to influence the behavior of algorithms for the solution of Problem (1).

Examples of possible choices for the functions h_i are

(1) $\beta_i(x_i - \bar{x}_i)$,

(2) $\alpha_i(x_i - \bar{x}_i)^3 + \beta_i(x_i - \bar{x}_i)$,

(3) $\alpha_i(x_i - \bar{x}_i)^{7/3} + \beta_i(x_i - \bar{x}_i)$,

where α_i, β_i are nonnegative constants. We note that $h'_i(\bar{x}_i) = \beta_i$ with all these three choices, so that the values of the multipliers are easily assigned. Furthermore, the Hessian of f at $\bar{x}$ is the same as that of g. This latter property is significant because the Hessian of f at $\bar{x}$ will determine, to a large extent, the "second-order" characteristics of the problem, which therefore are inherited from those of g. To be more precise, what really matters is the projection of the Hessian of f onto the subspace of free variables (those indexed by F), which is also user established.

Choice (1) implies that the Hessian of f is actually the same as that of g at every point. This choice seems particularly suitable when the function g is quadratic, since in this case the generated box-constrained problem will remain quadratic. Choice (3) is interesting because it will make f no more than twice continuously differentiable, while choice (2) can be regarded as a standard, average choice.

In general, we cannot guarantee that the problems generated by this technique have no stationary points other than $\bar{x}$. This can be assured if f is strictly convex—for example, if g is strictly convex—and the functions h_i are convex for $i \in L$ and concave for $i \in U$ on K, which is the case for all three h_i considered above.

3. CONCLUSIONS

We have presented a new, simple technique for generating box-constrained minimization problems. The generated problem is built from a given

unconstrained problem with a known solution point. The box-constrained problem obtained has the same solution point as that of the underlying unconstrained problem. Furthermore, the sparsity pattern of the unconstrained problem is also preserved. The number and position of the constraints and of the active constraints, the Lagrange multipliers, and the shape of the feasible region can be easily controlled.

We hope that the simplicity of our approach and the availability of corresponding software [Facchinei et al. 1997] will encourage researchers to employ the generation technique described here to test codes for box-constrained optimization.

REFERENCES

AVERICK, B., CARTER, R., MORÉ, J., AND XUE, G.-L. 1992. The MINPACK-2 test problem collection. Rep. MCS-p153-0692, Argonne National Laboratory, Argonne, IL.

BARTELS, R. H AND MAHDAVI AMIRI, N. 1986. On generating test problems for nonlinear programming algorithms. *SIAM J. Sci. Stat. Comput. 7*, 3 (July), 769–798.

BONGARTZ, I., CONN, A. R., GOULD, N., AND TOINT, PH. L. 1995. CUTE: Constrained and unconstrained testing environment. *ACM Trans. Math. Softw. 21*, 1 (Mar.), 123–160.

BROWN, A. A. AND BARTHOLOMEW-BIGGS, M. C. 1988. Some effective methods for unconstrained optimization based on the solution of systems of ordinary differential equations. *J. Optim. Theory Appl. 62*, 2 (Aug. 1989), 211–224.

FACCHINEI, F., JÚDICE, J., AND SOARES, J. 1997. Algorithm 774: Fortran subroutines for generating box-constrained optimization problems. *ACM Trans. Math. Softw. 23*, 3 (Sept.). This issue.

HOCK, W. AND SCHITTKOWSKI, K. 1981. *Test Examples for Nonlinear Programming Codes.* Springer Lecture Notes in Economics and Mathematical Systems, vol. 187. Springer-Verlag New York, Inc., New York, NY.

LOOTSMA, F. 1985. Comparative performance evaluation, experimental design and generation of test problems in nonlinear optimization. In *Proceedings of the NATO Advanced Study Institute* (Bad Windsheim, July 23-Aug. 2, 1984), K. Schittkowski, Ed. Springer-Verlag New York, Inc., New York, NY, 249–260.

GARBOW, B. S., HILLSTROM, K. E., AND MORÉ, J. J. 1981. Testing unconstrained optimization software. *ACM Trans. Math. Softw. 7*, 1, 17–41.

NASH, S. AND NOCEDAL, J. 1991. A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization. *SIAM J. Optim. 1*, 3 (Aug.), 358–372.

ROSEN, J. AND SUZUKI, S. 1965. Construction of nonlinear programming test problems. *Commun. ACM 8*, 2.

SCHITTKOWSKI, K. 1980. *Nonlinear Programming Codes: Information, Tests, Performance.* Springer Lecture Notes in Economics and Mathematical Systems, vol. 183. Springer-Verlag, Berlin, Germany.

SCHITTKOWSKI, K. 1992. *Randomly Generated Nonlinear Programming Test Problems.* Springer Lecture Notes in Control and Information Sciences, vol. 187. Springer-Verlag, Berlin, Germany.

Received: September 1995; revised: October 1996; accepted: February 1997