A CLASS OF CONVERGENT PRIMAL–DUAL SUBGRADIENT ALGORITHMS FOR DECOMPOSABLE CONVEX PROGRAMS

S. SEN
Department of Systems and Industrial Engineering, University of Arizona, Tucson, AZ 85721, USA

Hanif D. SHERALI
Department of Industrial Engineering and Operations Research, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, USA

Received 24 April 1985
Revised manuscript received 1 August 1985

In this paper we develop a primal–dual subgradient algorithm for preferably decomposable, generally nondifferentiable, convex programming problems, under usual regularity conditions. The algorithm employs a Lagrangian dual function along with a suitable penalty function which satisfies a specified set of properties, in order to generate a sequence of primal and dual iterates for which some subsequence converges to a pair of primal–dual optimal solutions. Several classical types of penalty functions are shown to satisfy these specified properties. A geometric convergence rate is established for the algorithm under some additional assumptions. This approach has three principal advantages. Firstly, both primal and dual solutions are available which prove to be useful in several contexts. Secondly, the choice of step sizes, which plays an important role in subgradient optimization, is guided more determinably in this method via primal and dual information. Thirdly, typical subgradient algorithms suffer from the lack of an appropriate stopping criterion, and so the quality of the solution obtained after a finite number of steps is usually unknown. In contrast, by using the primal–dual gap, the proposed algorithm possesses a natural stopping criterion.

Key words: Nondifferentiable optimization, subgradient optimization, penalty functions, Lagrangian duals.

1. Introduction

Results on relaxation methods and subgradient optimization have been reported in the literature for at least three decades, starting with Agmon (1954) and Motzkin and Schoenberg (1954). Early results on the convergence rates of such algorithms were given by Poljak (1969). Recent research, inspired by Khacijan's method (Khacijan (1979)) has shifted the focus on to ellipsoid methods and convergence rates have been established by Goffin (1981). Nonetheless, the subgradient optimization type of algorithms which were first popularized by Poljak (1967, 1969) and by Held, Wolfe and Crowder (1974) are still widely used today. However, several implementation aspects of algorithms of this type have been a source of frustration. In the first place, algorithms of this type are usually not dual adequate for the problems they solve, in the terminology of Geoffrion (1972). For example, if one
were using subgradient optimization on a Lagrangian relaxation of some discrete optimization problem in order to compute a quick lower bound in the context of a branch and bound algorithm, then the subgradient algorithm would typically provide one with only a (hopefully) near optimal set of dual multipliers. However, in addition to the lower bound thus obtained, it would be desirable to obtain a near optimal primal solution as well for fathoming purposes or for making branching decisions. Secondly, the choice of a step size in procedures of this type is a very crucial decision from the viewpoint of algorithmic efficiency or even adequacy. The step size condition stated by Poljak (1967) leaves a great degree of flexibility in this choice, and the research of Poljak (1969), Held, Wolfe and Crowder (1974), and Bazaraa and Sherali (1981) has focused on providing more definite guidelines in this direction. However, these step size choices are heuristics, and different variations have different effects on different problems. Thirdly, and perhaps most importantly, no effective stopping criterion is available for such algorithms. For the most part, one terminates the algorithm after a prescribed number of iterations or when the heuristic step size becomes small enough. Sufficient stopping conditions such as when the norm of a subgradient falls below a tolerance level are seldom realized, even when an unconstrained optimum is also a constrained optimum. A recent study by Sherali and Myers (1984) indicates that even a carefully designed subgradient algorithm can come to a crawling halt with a significant gap from optimality, for several classes of not-so-well structured problems. The discerning factor is that when the algorithm does begin to crawl, there is no available indication of its gap from optimality.

This paper presents a primal–dual subgradient algorithm for convex programming problems under usual regularity conditions which ensure the absence of a duality gap. Although differentiability is not a requirement, it is preferred that the problem is decomposable in that it possesses an easily optimized Lagrangian function. The algorithm is motivated to circumvent the above mentioned drawbacks of typical subgradient algorithms. Here, a pair of primal–dual iterates are generated which are demonstrated to converge to an optimal pair of primal–dual solutions. These iterates are produced via a penalty function satisfying certain stated properties and a Lagrangian dual function. Consequently, not only does one have the facility of obtaining both primal and dual (near) optimal solutions, but one also has a convenient, natural stopping criterion based on the gap between the penalty and Lagrangian functions. Furthermore, in the event of premature termination, this gap provides an indication of the quality of solutions at hand. Moreover, as will be seen in the sequel, the choice of a step size is also suitably guided by the primal penalty and the dual Lagrangian functions. As with Poljak's (1969) scheme, a geometric convergence rate is established under some additional assumptions. The algorithm reserves a degree of flexibility, principally by permitting a wide range of well established penalty functions to be used in its context.

The following section states the problem, the assumptions and the algorithm, and provides a convergence proof along with a rate of convergence analysis. Thereafter, Section 3 provides specific forms of the penalty functions that may be used in the