Improving the Decomposition of Partially Separable Functions in the Context of Large-Scale Optimization: a First Approach*

Andrew R. Conn†
IBM T.J. Watson Research Center, Yorktown Heights, USA

Nick Gould
CERFACS, Toulouse, France, EC

Philippe L. Toint
Department of Mathematics, FUNDP, Namur, Belgium, EC

Abstract

This paper examines the question of modifying the decomposition of a partially separable function in order to improve computational efficiency of large-scale minimization algorithms using a conjugate-gradient inner iteration. The context and motivation are given and the application of a simple strategy discussed on examples extracted from the CUTE test problem collection.

Keywords: exploitation of structure, algorithmic efficiency, partially separable functions.

*This research was supported in part by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Air Force Office of Scientific Research under Contract No F49620-91-C-0079. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

†The authors wish to acknowledge additional funding provided by a NATO travel grant.

W. W. Hager et al. (eds.), Large Scale Optimization: State of the Art, 82–94.
© 1994 *Kluwer Academic Publishers.*

1 Introduction

Large-scale numerical optimization, like many other fields involving large problems, heavily relies on two fundamental but distinct endeavours: the use of structure and the search for maximum algorithmic efficiency. Indeed, many of the methods proposed in this area require that the user specifies the problem's structure in some prescribed way, and are designed to exploit this given structure to the largest extent possible. However, it is frequently assumed that the problem's structure is *given*, and that a good algorithm has to exploit it. In this paper, we consider the complementary point of view: we examine the question of *modifying* the problem's structure, in the hope that this modified structure can lead to improved algorithmic performance. Although not new in other areas of computational mathematics (see, for instance the work by Chan and McCormick [3] on how to make sparse matrices sparser), this idea does not seem to have been much studied in the context of large-scale optimization.

It is the purpose of this paper to consider this question in the context of partially separable functions. Introduced by Griewank and Toint [8], this particular structure and its generalization to group partial separability have shown to be very useful in the design of algorithms for large-scale optimization problems, both constrained and unconstrained. For instance, the LANCELOT package (see Conn *et al.* [7]) is based on this structural concept. In this context, we will consider that the partially separable structure of a function is given, and will then try to improve it with a very specific goal in mind: we aim at reducing the amount of computational time spent in the calculation of a step of a truncated-Newton algorithm using the conjugate gradient technique. This particular choice is motivated by the frequent use of this technique in large-scale optimization methods, and, more precisely, by the potential benefits that could be achieved within the LANCELOT package itself.

The paper is organized as follows. Section 2 formally introduces two related problems in modifying a partially separable structure: element merging and expansion. Section 3 describes a simple algorithmic approach to partially separable structure improvement, while Section 4 presents some results obtained by applying the algorithm of Section 3 to test examples extracted from the CUTE test problem collection of Bongartz *et al.* [2]. A more general discussion of the subject is presented in Section 5.

2 The Merging and Expansion Problems in Partially Separable Structures

In order to motivate our approach in a simple framework, we consider the unconstrained optimization problem of minimizing

$$f(x) = \sum_{i=1}^{m} f_i(x), \tag{2.1}$$