Chapter 1

Introduction

1.1 Why derivative-free optimization

It is well known that extensive useful information is contained in the derivatives of any function one wishes to optimize. After all, the "standard" mathematical characterization of a local minimum, given by the first-order necessary conditions, requires, for continuously differentiable functions, that the first-order derivatives are zero. However, for a variety of reasons there have always been many instances where (at least some) derivatives are unavailable or unreliable. Nevertheless, under such circumstances it may still be desirable to carry out optimization. Consequently, a class of nonlinear optimization techniques called derivative-free optimization methods has always been needed. In fact, we consider optimization without derivatives one of the most important, open, and challenging areas in computational science and engineering, and one with enormous practical potential. The reason that it is challenging is that, from the point of view of optimization, one gives up so much information by not having derivatives. The source of its current, practical importance is the ever growing need to solve optimization problems defined by functions for which derivatives are unavailable or available at a prohibitive cost. Increasing complexity in mathematical modeling, higher sophistication of scientific computing, and an abundance of legacy codes are some of the reasons why derivative-free optimization is currently an area of great demand.

In earlier days of nonlinear optimization perhaps one of the most common reasons for using derivative-free methods was the lack of sophistication or perseverance of the user. The users knew they wanted to improve on their current "solution," but they wanted to use something simple that they could understand, and so they used (and, unfortunately, sometimes continue to use) nonderivative methods, like the method by Nelder and Mead [177], even when more appropriate algorithms were available. In defense of the practitioner, we should remember that until relatively recently computing derivatives was the single most common source of user error in applying optimization software (see, for example, [104, Chapter 8, page 297]). As the scale and difficulty of the applications increased, more sophisticated derivative-based optimization methods became more essential. With the growth and development of derivative-based nonlinear optimization methods it became evident that large-scale problems can be solved efficiently, but only if there is accurate derivative infor-