4 Incremental Gradient, Subgradient, and Proximal Methods for Convex Optimization: A Survey

Dimitri P. Bertsekas dimitri@mit.edu
Dept. of Electr. Engineering and Comp. Science, M.I.T.
Cambridge, MA, 02139

We survey incremental methods for minimizing a sum $\sum_{i=1}^{m} f_i(x)$ consisting of a large number of convex component functions f_i. Our methods consist of iterations applied to single components, and have proved very effective in practice. We introduce a unified algorithmic framework for a variety of such methods, some involving gradient and subgradient iterations, which are known, and some involving combinations of subgradient and proximal methods, which are new and offer greater flexibility in exploiting the special structure of f_i. We provide an analysis of the convergence and rate of convergence properties of these methods, including the advantages offered by randomization in the selection of components. We also survey applications in inference/machine learning, signal processing, and large-scale and distributed optimization.

4.1 Introduction

We consider optimization problems with a cost function consisting of a large number of component functions, such as

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{m} f_i(x) \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{4.1}$$

where $f_i : \Re^n \mapsto \Re$, $i = 1, \ldots, m$ are real-valued functions, and X is a closed convex set.[1] We focus on the case where the number of components m is very large, and there is an incentive to use incremental methods that operate on a single component f_i at each iteration, rather than on the entire cost function. If each incremental iteration tends to make reasonable progress in some "average" sense, then, depending on the value of m, an incremental method may significantly outperform (by orders of magnitude) its nonincremental counterpart, as extensive experience has shown.

In this chapter, we survey the algorithmic properties of incremental methods in a unified framework, based on the author's recent work on incremental proximal methods (Bertsekas, 2010). In this section, we first provide an overview of representative applications, and then we discuss three types of incremental methods: gradient, subgradient, and proximal. We unify these methods into a combined method, which we use as a vehicle for analysis in Sections 4.2, 4.3, and 4.4. Finally, we discuss in greater detail some illustrative applications in Section 4.5. Some of the proofs of propositions have been omitted and can be found in the report (Bertsekas, 2010).

4.1.1 Some Examples of Additive Cost Problems

Additive cost problems of the form (4.1) arise in a variety of contexts. Let us provide a few examples where the incremental approach may have an advantage over alternatives.

Example 4.1 (Least Squares and Inference). An important context where cost functions of the form $\sum_{i=1}^{m} f_i(x)$ arise is inference/machine learning, where each term $f_i(x)$ corresponds to error between some data and

1. Throughout the chapter, we will operate within the n-dimensional space $\Re^n$ with the standard Euclidean norm, denoted $\|\cdot\|$. All vectors are considered column vectors and a prime denotes transposition, so $x'x = \|x\|^2$. We will be using standard terminology of convex optimization throughout, as given, for example, in textbooks such as Rockafellar (1970), or the author's recent book (Bertsekas, 2009).

the output of a parametric model, with x being the vector of parameters. An example is *linear least-squares* problems, where f_i has quadratic structure, except for a regularization function, which may be differentiable/quadratic, as in the classical regression problem

$$\sum_{i=1}^{m}(a_i'x - b_i)^2 + \gamma\|x - \overline{x}\|^2, \quad \text{s.t.} \quad x \in \Re^n,$$

where $\overline{x}$ is given, or nondifferentiable, as in the ℓ_1-regularization problem

$$\sum_{i=1}^{m}(a_i'x - b_i)^2 + \gamma\sum_{j=1}^{n}|x_j|, \quad \text{s.t.} \quad (x_1, \ldots, x_n) \in \Re^n,$$

which will be discussed further in Section 4.5.

A more general class of additive cost problems is *nonlinear least squares*. Here

$$f_i(x) = \big(h_i(x)\big)^2,$$

where $h_i(x)$ represents the difference between the ith measurement (out of m) from a physical system and the output of a parametric model whose parameter vector is x. Problems of nonlinear curve fitting and regression, as well as problems of training neural networks, fall in this category, and they are typically nonconvex.

Another possibility is to use a nonquadratic function to penalize the error between some data and the output of the parametric model. For example, in place of the squared error $(a_i'x - b_i)^2$, we may use

$$f_i(x) = \ell(a_i'x - b_i),$$

where ℓ is a convex function. This is a common approach in robust estimation and some support vector machine formulations.

Still another example is *maximum likelihood estimation*, where f_i is a log-likelihood function of the form

$$f_i(x) = -\log P_Y(y_i; x),$$

where $y_1, \ldots, y_m$ represents values of independent samples of a random vector whose distribution $P_Y(\cdot; x)$ depends on an unknown parameter vector $x \in \Re^n$ that one wishes to estimate. Related contexts include "incomplete" data cases, where the expectation-maximization (EM) approach is used.

Example 4.2 (Dual Optimization in Separable Problems). Consider

the problem

$$\begin{aligned} &\text{maximize} \quad \sum_{i=1}^{m} c_i(y_i) \\ &\text{subject to} \quad \sum_{i=1}^{m} g_i(y_i) \geq 0, \quad y_i \in Y_i, \quad i = 1, \ldots, m, \end{aligned}$$

where $c_i : \Re \mapsto \Re$ and $g_i : \Re \mapsto \Re^n$ are functions of the single scalar coordinate y_i, and Y_i are given sets of scalars. Then, by assigning a dual vector/multiplier $x \in \Re^n$ to the n-dimensional constraint function, we obtain the dual problem

$$\text{minimize} \quad \sum_{i=1}^{n} f_i(x), \quad \text{subject to } x \geq 0,$$

where

$$f_i(x) = \sup_{y_i \in Y_i} \left\{ c_i(y_i) + x' g_i(y_i) \right\},$$

which has the additive form (4.1). Note that Y_i is not assumed to be convex, so integer programming and other discrete optimization problems are included. However, the dual cost function components f_i are always convex, and their values and subgradients can often be computed either analytically or with a one-dimensional maximization.

Example 4.3 (Minimization of an Expected Value: Stochastic Programming). Consider the minimization of an expected value

$$\begin{aligned} &\text{minimize} \quad E\big\{F(x, w)\big\} \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{4.2}$$

where w is a random variable taking a finite but very large number of values w_i, $i = 1, \ldots, m$, with corresponding probabilities π_i. Then the cost function consists of the sum of the m functions $\pi_i F(x, w_i)$.

An example is *stochastic programming*, a classical model of two-stage optimization under uncertainty. A vector $x \in X$ is selected, a random event occurs that has m possible outcomes $w_1, \ldots, w_m$, and then another vector y is selected from some set Y with knowledge of the outcome that occurred. Then, for optimization purposes, we need to specify a different vector $y_i \in Y$ for each outcome w_i. The problem is to minimize the expected cost

$$F(x) + \sum_{i=1}^{m} \pi_i G_i(y_i),$$

where $G_i(y_i)$ is the cost associated with the occurrence of w_i, and π_i is the corresponding probability. This is a problem with an additive cost function. Furthermore, if there are separable (e.g., linear) constraints coupling the vectors x and y_i, the problem has a separable form.

Additive cost function problems also arise from problem (4.2) in a different way: when the expected value $E\{F(x,w)\}$ is approximated by an m-sample average

$$f(x) = \frac{1}{m}\sum_{i=1}^{m} F(x,w_i),$$

where w_i are independent samples of the random variable w. The minimum of the sample average $f(x)$ is then taken as an approximation of the minimum of $E\{F(x,w)\}$.

Example 4.4 (Problems with Many Constraints). Problems of the form

$$\begin{aligned} &\text{minimize} \quad f(x) \\ &\text{subject to} \quad g_j(x) \le 0, \quad j = 1,\dots,m,\ x \in X, \end{aligned} \tag{4.3}$$

where the number r of constraints is very large, often arise in practice, either directly or via reformulation from other problems. They can be handled in a variety of ways. One possibility is to adopt a penalty function approach, and replace problem (4.3) with

$$\begin{aligned} &\text{minimize} \quad f(x) + c\sum_{j=1}^{r} P\big(g_j(x)\big) \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{4.4}$$

where $P(\cdot)$ is a scalar penalty function satisfying $P(t) = 0$ if $t \le 0$, and $P(t) > 0$ if $t > 0$, and c is a positive penalty parameter. For example, one may use the quadratic penalty $P(t) = \big(\max\{0,t\}\big)^2$. An interesting alternative is to use $P(t) = \max\{0,t\}$, in which case it can be shown that the optimal solutions of problems (4.3) and (4.4) coincide when c is sufficiently large (see, for example, Bertsekas et al. (2003, Section 7.3) for the case in which f is convex). The cost function of the penalized problem (4.4) is of the additive form (4.1).

The idea of replacing constraints with penalties can be extended to the case where the constraint $x \in X$ in problem (4.3) has the form $x \in \cap_{j=1}^{m} X_j$. Then, under relatively mild conditions, problem (4.3) is equivalent to the

unconstrained minimization of

$$f(x) + c\sum_{j=1}^{r} P\big(g_j(x)\big) + \gamma \sum_{j=1}^{m} \text{dist}(x; X_j),$$

where $\text{dist}(x; X_j) = \inf_{y \in X_j} \|y - x\|$ and γ is a sufficiently large penalty parameter. We discuss this possibility in Section 4.5.

Example 4.5 (**Distributed Incremental Optimization in Sensor Networks).** Consider a network of m sensors where data are collected and used to solve some inference problem involving a parameter vector x. If $f_i(x)$ represents an error penalty for the data collected by the ith sensor, the inference problem is of the form (4.1). While it is possible to collect all the data at a fusion center where the problem will be solved in centralized manner, it may be preferable to adopt a distributed approach in order to save data communication overhead and/or take advantage of parallelism in computation. In such an approach the current iterate x_k is passed from one sensor to another, with each sensor i performing an incremental iteration involving just its local component function f_i, and the entire cost function need not be known at any one location. We refer to Blatt et al. (2007), and Rabbat and Nowak (2004, 2005) for further discussion.

Example 4.6 (**Weber Problem in Location Theory).** We want to find a point x in the plane whose sum of weighted distances from a given set of points $y_1, \ldots, y_m$ is minimized. Mathematically, the problem is

$$\sum_{i=1}^{m} w_i \|x - y_i\|, \quad \text{s.t.} \quad x \in \Re^n,$$

where $w_1, \ldots, w_m$ are given positive scalars. This problem descends from the famous Fermat-Torricelli-Viviani problem (see (Boltyanski et al., 1999) for an account of the history; Fermat's formulation was for the case of a triangle, where $m = 3$). It is a basic problem in location theory, and has received a lot of attention. The algorithmic approaches of this chapter would be of potential interest when the number of points m is large. We refer to Beck and Teboulle (2010) for a discussion that is relevant to our context.

4.1.2 Incremental Gradient Methods: Differentiable Problems

In the case where the components f_i are differentiable (not necessarily convex), we may use incremental gradient methods, which have the form

$$x_{k+1} = P_X\big(x_k - \alpha_k \nabla f_{i_k}(x_k)\big), \tag{4.5}$$

where α_k is a positive stepsize, $P_X(\cdot)$ denotes projection on X, and i_k is the index of the cost component that is iterated on. Such methods have a long history, particularly for the unconstrained case ($X = \Re^n$), starting with the Widrow-Hoff least-mean-squares (LMS) method (Widrow and Hoff, 1960) for positive semidefinite quadratic component functions (see e.g., (Luo, 1993), (Bertsekas and Tsitsiklis, 1996, Section 3.2.5), (Bertsekas, 1999, Section 1.5.2)). They have also been used extensively for the training of neural networks, a case of nonquadratic/nonconvex cost components, under the generic name "backpropagation methods." There are several variants of these methods, which differ in the stepsize selection scheme, and iin the order in which components are taken up for iteration (it could be deterministic or randomized). They are supported by convergence analyses under various conditions; see Luo (1993), Grippo (1994), Grippo (2000), Luo and Tseng (1994), Mangasarian and Solodov (1994), Bertsekas (1997), Solodov (1998), and Tseng (1998).

When comparing the incremental gradient method with its classical non-incremental gradient counterpart (where $m = 1$ and all components are lumped into a single function $f(x) = \sum_{i=1}^m f_i(x)$), it is important to realize that there are two complementary performance issues to consider.

1. *Progress when far from convergence.* Here the incremental method can be much faster. For an extreme case let $X = \Re^n$ (no constraints), and take m very large and all components f_i identical to each other. Then an incremental iteration requires m times less computation than a classical gradient iteration, but gives exactly the same result. While this is an extreme example, it reflects the essential mechanism by which incremental methods can be far superior: when the components f_i are not too dissimilar, far from the minimum a single component gradient will point to, "more or less," the right direction (see also the discussion of Bertsekas (1997) and Bertsekas (1999, Example 1.5.5 and Exercise 1.5.5).)

2. *Progress when close to convergence.* Here the incremental method is generally inferior. As we will discuss shortly, it converges at a sublinear rate because it requires a diminishing stepsize α_k, compared with the typically linear rate achieved with the classical gradient method when a small, constant stepsize is used ($\alpha_k \equiv \alpha$). One may use a constant stepsize with the incremental method - and indeed this may be the preferred mode of implementation - but then the method typically oscillates in the neighborhood of a solution, with the size of the oscillation roughly proportional to α, as examples and theoretical analysis show.

To understand the convergence mechanism of incremental gradient methods, let us consider the case $X = \Re^n$, and assume that the component

functions f_i are selected for iteration according to a cyclic order (i.e., for every ℓ, $i_{\ell m} = 1, i_{\ell m+1} = 2, \ldots, i_{\ell m+m-1} = m$), and let us assume that α_k is constant within a cycle (i.e., $\alpha_{\ell m} = \alpha_{\ell m+1} = \cdots = \alpha_{\ell m+m-1}$). Then, viewing the iteration (4.5) in terms of cycles, we have, for every k that marks the beginning of a cycle ($i_k = 1$),

$$x_{k+m} = x_k - \alpha_k \sum_{i=1}^{m} \nabla f_i(x_{k+i-1}) = x_k - \alpha_k\big(\nabla f(x_k) + e_k\big),$$

where f is the cost function/sum of components, $f(x) = \sum_{i=1}^m f_i(x)$, and e_k is given by

$$e_k = \sum_{i=1}^{m} \big(\nabla f_i(x_k) - \nabla f_i(x_{k+i-1})\big),$$

and may be viewed as an error in the calculation of the gradient $\nabla f(x_k)$. For Lipschitz continuous gradient functions ∇f_i, the error e_k is proportional to α_k, and this shows two fundamental properties of incremental gradient methods, which hold generally for the other incremental methods of this chapter as well.

1. A constant stepsize ($\alpha_k \equiv \alpha$) typically cannot guarantee convergence, since then the size of the gradient error $\|e_k\|$ is typically bounded away from 0. Instead, a peculiar form of convergence takes place for constant but sufficiently small α, whereby the iterates within cycles converge to corresponding points of a limit cycle. This is true even in the most favorable case of a linear least squares problem (see Luo (1993), or the textbook analysis of Bertsekas (1999, Section 1.5.1)).

2. A diminishing stepsize (such as $\alpha_k = O(1/k)$) leads to a diminishing error e_k, so (under the appropriate Lipschitz condition) it can result in convergence to a stationary point of f.

A corollary of these properties is that the price for achieving convergence is the slow (sublinear) asymptotic rate of convergence associated with a diminishing stepsize, which compares unfavorably with the often linear rate of convergence associated with a constant stepsize and the nonincremental gradient method. However, in practical terms this argument does not tell the entire story, since in the early iterations, the incremental gradient method often achieves a much faster convergence rate than its nonincremental counterpart. In practice, the incremental method is usually operated with a stepsize that either is constant or is gradually reduced up to a positive value small enough that the resulting asymptotic oscillation is of no essential concern. An alternative is to use a constant stepsize throughout, but to

reduce over time the degree of incrementalism, so that ultimately the method becomes nonincremental and achieves a linear convergence rate (see Bertsekas (1997) and Solodov (1998)).

Aside from extensions to nondifferentiable cost problems, for $X = \Re^n$ there is an important variant of the incremental gradient method that involves extrapolation along the direction of the difference of the preceding two iterates:

$$x_{k+1} = x_k - \alpha_k \nabla f_{i_k}(x_k) + \beta(x_k - x_{k-1}), \tag{4.6}$$

where β is a scalar in $[0, 1)$ and $x_{-1} = x_0$ (see e.g., Mangasarian and Solodov (1994), Tseng (1998), Bertsekas (1996, Section 3.2)). This is sometimes called the *incremental gradient method with momentum*. The nonincremental version of this method is the *heavy ball* method of Poljak (1964), which can be shown to have a faster convergence rate than the corresponding gradient method (see Polyak (1987, Section 3.2.1)). A nonincremental method of this type, but with variable and suitably chosen value of β, has been proposed by Nesterov (1983), and has received a lot of attention recently because it has optimal iteration complexity properties under certain conditions (see Nesterov (2004, 2005), Lu et al. (2008), Tseng (2008), and Beck and Teboulle (2009, 2010)). However, no incremental analogs of this method with favorable complexity properties are currently known.

Another variant of the incremental gradient method for the case $X = \Re^n$ has been proposed by Blatt et al. (2007), which (after the first m iterates are computed) has the form

$$x_{k+1} = x_k - \alpha \sum_{\ell=0}^{m-1} \nabla f_{i_{k-\ell}}(x_{k-\ell}). \tag{4.7}$$

(For $k < m$, the summation should go up to $\ell = \min\{k, m-1\}$, and α should be replaced by a corresponding larger value, such as $\alpha_k = m\alpha/(k+1)$.) This method also computes the gradient incrementally, one component per iteration, but in place of the single component gradient $\nabla f_{i_k}(x_k)$ in (4.5), it uses an approximation to the total cost gradient $\nabla f(x_k)$, which is an aggregate of the component gradients computed in the past m iterations. A cyclic order of component function selection ($i_k = k$ modulo m plus 1) is assumed in (Blatt et al., 2007), and a convergence analysis is given, including a linear convergence rate result for a sufficiently small constant stepsize α and quadratic component functions f_i. It is not clear how iterations (4.5) and (4.7) compare in terms of rate of convergence, although the latter seems likely to make faster progress when close to convergence. Note that iteration (4.7) bears similarity to the incremental gradient iteration with momentum

(4.6) where $\beta \approx 1$. In particular, when $\alpha_k \equiv \alpha$, the sequence generated by (4.6) satisfies

$$x_{k+1} = x_k - \alpha \sum_{\ell=0}^{k} \beta^\ell \nabla f_{i_{k-\ell}}(x_{k-\ell}),$$

which resembles (4.7). There are no known analogs of iterations (4.6) and (4.7) for nondifferentiable cost problems.

Among alternative incremental methods for differentiable cost problems, we also mention versions of the Gauss-Newton method for nonlinear least-squares problems, based on the extended Kalman filter ((Davidon, 1976), (Bertsekas, 1996), and (Moriyama et al., 2003)). They are mathematically equivalent to the ordinary Gauss-Newton method for linear least squares, which they solve exactly after a single pass through the component functions f_i, but they often perform much faster in the nonlinear case, particularly when m is large.

Let us finally note that incremental gradient methods are related to stochastic gradient methods, which aim to minimize an expected value $E\{F(x,w)\}$ (cf. Example 1.3) by using the iteration

$$x_{k+1} = x_k - \alpha_k \nabla F(x_k, w_k),$$

where w_k is a sample of the random variable w. These methods also have a long history (see Polyak and Tsypkin (1973), Ljung (1977), Kushner and Clark (1978), Tsitsiklis et al. (1986), Polyak (1987), Bertsekas and Tsitsiklis (1989, 1996, 2000), Gaivoronski (1994), Pflug (1996), Kushner and Yin (1997), Bottou (2005), Meyn (2007), Borkar (2008), Nemirovski et al. (2009), Lee and Wright (2010)), and are strongly connected with stochastic approximation algorithms. The main difference between stochastic and deterministic formulations is that the former involve sequentially sampling cost components from an infinite population under some statistical assumptions, while in the latter the set of cost components is predetermined and finite. However, it is possible to view the incremental gradient method (4.5), with a randomized selection of the component function f_i (i.e., with i_k chosen to be any one of the indexes $1, \ldots, m$, with equal probability $1/m$), as a stochastic gradient method (see Bertsekas and Tsitsiklis (1996, Example 4.4) and (Bertsekas and Tsitsiklis, 2000, Section 5)).

The stochastic formulation of incremental methods just discussed highlights an important application context where the component functions f_i are not given a priori, but become known sequentially through some observation process. Then it often makes sense to use an incremental method to process the component functions as they become available, and to obtain

approximate solutions as early as possible. In fact, this may be essential in time-sensitive and possibly time-varying environments, where solutions are needed "online." In such cases, one may hope that an adequate estimate of the optimal solution will be obtained before all the functions f_i are processed for the first time.

4.1.3 Incremental Subgradient Methods - Nondifferentiable Problems

Incremental subgradient methods apply to the case where the component functions f_i are convex and nondifferentiable at some points. They are similar to their gradient counterparts (4.5) except that an arbitrary subgradient $\tilde{\nabla} f_{i_k}(x_k)$ of the cost component f_{i_k} is used in place of the gradient:[2]

$$x_{k+1} = P_X\big(x_k - \alpha_k \tilde{\nabla} f_{i_k}(x_k)\big). \tag{4.8}$$

Such methods were first proposed in the general form (4.8) in the Soviet Union by Kibardin (1980), following the earlier paper by Litvakov (1966) (which considered convex/nondifferentiable extensions of linear least-squares problems) and related subsequent proposals.[3] These works remained unnoticed until about 2005 in the Western literature, where incremental methods were often reinvented in different contexts and with different lines of analysis. See Ben-Tal et al. (2001), Nedić and Bertsekas (2000, 2001, 2010), Nedić et al. (2001), Kiwiel (2004), Rabbat and Nowak (2004, 2005), Gaudioso et al. (2006), Shalev-Shwartz et al. (2007), Neto and De Pierro (2009), Johansson et al. (2009), Predd et al. (2009), Ram et al. (2009a,b), and Duchi et al. (2010).

Incremental subgradient methods have convergence characteristics that are similar in many ways to their gradient counterparts, the most important similarity being the necessity for a diminishing stepsize α_k for convergence. The lines of analysis, however, tend to be different, since incremental gradient methods rely for convergence on arguments based on decrease of the cost function value, while incremental subgradient methods rely on argu-

---

2. In this chapter, we use $\tilde{\nabla} f(x)$ to denote a subgradient of a convex real-valued function f at a vector x. The choice of $\tilde{\nabla} f(x)$ from within the subdifferential $\partial f(x)$ at x will be clear from the context.

3. Generally, in those times, algorithmic ideas relating to simple gradient methods with and without deterministic and stochastic errors were popular in the Soviet scientific community, partly due to an emphasis on stochastic iterative algorithms, such as pseudogradient and stochastic approximation; the works of Ermoliev, Polyak, and Tsypkin, to name a few of the principal contributors, are representative (Ermoliev, 1969; Polyak and Tsypkin, 1973; Ermoliev, 1976; Polyak, 1978, 1987). By contrast, the emphasis in the Western literature at the time was on more complex Newton-like and conjugate direction methods.

ments based on decrease of the iterates' distance from the optimal solution set. The line of analysis of this chapter is of the latter type, and is similar to earlier works of the author and his collaborators (see Nedić and Bertsekas (2000), Nedić and Bertsekas (2001), Nedić et al. (2001), and the textbook presentations in Bertsekas (1999) and Bertsekas et al. (2003)).

Note two important ramifications of the lack of differentiability of the component functions f_i:

1. Convexity of f_i becomes essential, since the notion of subgradient is connected with convexity (subgradient-like algorithms for nondifferentiable / nonconvex problems have been suggested in the literature, but tend to be complicated and have not found much application thus far).

2. There is more reason to favor the incremental over the nonincremental methods, since (contrary to the differentiable case) nonincremental subgradient methods also require a diminishing stepsize for convergence, and typically achieve a sublinear rate of convergence. Thus the one theoretical advantage of the nonincremental gradient method discussed earlier is not shared by its subgradient counterpart.

Finally, just as in the differentiable case, there is a substantial literature for stochastic versions of subgradient methods. In fact, as we will discuss in this chapter, there is a potentially significant advantage in turning the method into a stochastic one by randomizing the order of selection of the components f_i for iteration.

4.1.4 Incremental Proximal Methods

We now consider an extension of the incremental approach to proximal algorithms. The simplest one for problem (4.1) is of the form

$$x_{k+1} = \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}, \tag{4.9}$$

which relates to the proximal minimization algorithm ((Martinet, 1970), (Rockafellar, 1976)) in the same way that the incremental subgradient method (4.8) relates to the classical nonincremental subgradient method.[4] Here $\{\alpha_k\}$ is a positive scalar sequence, and we will assume that each $f_i : \Re^n \mapsto \Re$ is a convex function and X is a nonempty closed convex set.

4. In this chapter, we restrict our attention to proximal methods with the quadratic regularization term $\|x-x_k\|^2$. Our approach is applicable in principle when a nonquadratic term is used instead, in order to match the structure of the given problem. The discussion of such alternative algorithms is beyond our scope.

The motivation for this type of method, which was considered only recently in Bertsekas (2010), is that with a favorable structure of the components, the proximal iteration (4.8) may be obtained in closed form or be relatively simple, in which case it may be preferable to a gradient or subgradient iteration. In this connection, we note that, generally, proximal iterations are considered more stable than gradient iterations; for example, in the nonincremental case, they converge essentially for any choice of α_k, while this is not so for gradient methods.

While some cost function components may be well suited for a proximal iteration, others may not be because the minimization (4.9) is inconvenient, so it makes sense to consider combinations of gradient/subgradient and proximal iterations. In fact, in the past this has motivated nonincremental combinations of gradient and proximal methods for minimizing the sum of two functions (or more generally, finding a zero of the sum of two nonlinear operators). These methods have a long history, dating to the splitting algorithms of Lions and Mercier (1979) and Passty (1979), and have become popular more recently (see Beck and Teboulle (2009, 2010), and the references they cite for specialized algorithms, such as shrinkage/thresholding, cf. Section 5.1).

With similar motivation in mind, we adopt in this paper a unified algorithmic framework that includes incremental gradient, subgradient, and proximal methods and their combinations, and highlights their common structure and behavior. We focus on problems of the form

$$\begin{aligned} &\text{minimize} \quad F(x) \stackrel{\text{def}}{=} \sum_{i=1}^{m} F_i(x) \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{4.10}$$

where for all i,

$$F_i(x) = f_i(x) + h_i(x), \tag{4.11}$$

$f_i : \Re^n \mapsto \Re^n$ and $h_i : \Re^n \mapsto \Re$ are real-valued convex functions, and X is a nonempty closed convex set.

In Section 4.2, we consider several incremental algorithms that iterate on the components f_i with a proximal iteration, and on the components h_i with a subgradient iteration. By choosing all the f_i or all the h_i to be identically zero, we obtain the subgradient and proximal iterations (4.8) and (4.9), respectively, as special cases. However, our methods offer greater flexibility, and may exploit the special structure of problems where the functions f_i are suitable for a proximal iteration, while the components h_i are not suitable, and thus may be preferably treated with a subgradient iteration.

In Section 4.3, we discuss the convergence and rate of convergence properties of methods that use a cyclic rule for component selection, and in Section 4.4, we discuss a randomized component selection rule. In summary, the convergence behavior of our incremental methods is similar to the one outlined earlier for the incremental subgradient method (4.8). This includes convergence within a certain error bound for a constant stepsize, exact convergence to an optimal solution for an appropriately diminishing stepsize, and improved convergence rate/iteration complexity when randomization is used to select the cost component for iteration. In Section 4.5, we illustrate our methods for some example applications.

4.2 Incremental Subgradient-Proximal Methods

In this section, we consider problems (4.10) and (4.11), and introduce several incremental algorithms that involve a combination of a proximal and a subgradient iteration. One of our algorithms has the form

$$z_k = \arg\min_{x\in X}\left\{f_{i_k}(x) + \frac{1}{2\alpha_k}\|x - x_k\|^2\right\}, \tag{4.12}$$

$$x_{k+1} = P_X\big(z_k - \alpha_k\tilde{\nabla}h_{i_k}(z_k)\big), \tag{4.13}$$

where $\tilde{\nabla}h_{i_k}(z_k)$ is an arbitrary subgradient of h_{i_k} at z_k. The iteration is well defined because the minimum in (4.12) is uniquely attained since f_i is continuous and $\|x - x_k\|^2$ is real-valued, strictly convex, and coercive, while the subdifferential $\partial h_i(z_k)$ is nonempty since h_i is real-valued. Also, by choosing all the f_i or all the h_i to be identically zero, we obtain the subgradient and proximal iterations (4.8) and (4.9), respectively, as special cases.

The iterations (4.12) and (4.13) maintain both sequences $\{z_k\}$ and $\{x_k\}$ within the constraint set X, but it may be convenient to relax this constraint for either the proximal or the subgradient iteration, thereby requiring a potentially simpler computation. This leads to the algorithm

$$z_k = \arg\min_{x\in\Re^n}\left\{f_{i_k}(x) + \frac{1}{2\alpha_k}\|x - x_k\|^2\right\}, \tag{4.14}$$

$$x_{k+1} = P_X\big(z_k - \alpha_k\tilde{\nabla}h_{i_k}(z_k)\big), \tag{4.15}$$

where the restriction $x \in X$ has been omitted from the proximal iteration,

and to the algorithm

$$z_k = x_k - \alpha_k \tilde{\nabla} h_{i_k}(x_k), \tag{4.16}$$

$$x_{k+1} = \arg\min_{x \in X} \left\{ f_{i_k}(x) + \frac{1}{2\alpha_k} \|x - z_k\|^2 \right\}, \tag{4.17}$$

where the projection onto X has been omitted from the subgradient iteration. It is also possible to use different stepsize sequences in the proximal and subgradient iterations, but for notational simplicity we will not discuss this type of algorithm.

All of the incremental proximal algorithms given above are new to our knowledge, having first been proposed by Bertsekas (2010). The closest connection to the existing proximal methods is the "proximal gradient" method, which has been analyzed and discussed recently in the context of several machine-learning applications by Beck and Teboulle (2009, 2010). (It can also be interpreted in terms of splitting algorithms (Lions and Mercier, 1979), (Passty, 1979).) This method is nonincremental, applies to differentiable h_i and, contrary to subgradient and incremental methods, it does not require a diminishing stepsize for convergence to the optimum. In fact, the line of convergence analysis of Beck and Teboulle (2009, 2010) relies on the differentiability of h_i and the nonincremental character of the proximal gradient method, and thus is different from ours.

Part (a) of the following proposition is a key fact about incremental proximal iterations. It shows that they are closely related to incremental subgradient iterations, the only difference being that the subgradient is evaluated at the end point of the iteration rather than at the starting point. Part (b) of the proposition provides an inequality that is well known in the theory of proximal methods, and will be useful for our convergence analysis. In the following method, we denote by $\mathrm{ri}(S)$ the relative interior of a convex set S, and by $\mathrm{dom}(f)$ the effective domain $\{x \mid f(x) < \infty\}$ of a function $f : \Re^n \mapsto (-\infty, \infty]$.

Proposition 4.1. *Let X be a nonempty closed convex set, and let $f : \Re^n \mapsto (-\infty, \infty]$ be a closed proper convex function such that $\mathrm{ri}(X) \cap \mathrm{ri}(\mathrm{dom}(f)) \neq \emptyset$. For any $x_k \in \Re^n$ and $\alpha_k > 0$, consider the proximal iteration*

$$x_{k+1} = \arg\min_{x \in X} \left\{ f(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}. \tag{4.18}$$

(a) The iteration can be written as

$$x_{k+1} = P_X\big(x_k - \alpha_k \tilde{\nabla} f(x_{k+1})\big), \qquad i = 1, \ldots, m, \tag{4.19}$$

where $\tilde{\nabla} f(x_{k+1})$ is some subgradient of f at x_{k+1}.

(b) For all $y \in X$, we have

$$\begin{aligned}\|x_{k+1}-y\|^2 &\le \|x_k-y\|^2 - 2\alpha_k(f(x_{k+1})-f(y)) - \|x_k-x_{k+1}\|^2\\ &\le \|x_k-y\|^2 - 2\alpha_k(f(x_{k+1})-f(y)).\end{aligned} \tag{4.20}$$

Proof. (a) We use the formula for the subdifferential of the sum of the three functions f, $(1/2\alpha_k)\|x-x_k\|^2$, and the indicator function of X (cf. (Bertsekas, 2009, Proposition 5.4.6)), together with the condition that 0 should belong to this subdifferential at the optimum x_{k+1}. We obtain that (4.18) holds if and only if

$$\frac{1}{\alpha_k}(x_k - x_{k+1}) \in \partial f(x_{k+1}) + N_X(x_{k+1}), \tag{4.21}$$

where $N_X(x_{k+1})$ is the normal cone of X at x_{k+1} (which is the set of vectors y such that $y'(x-x_{k+1}) \le 0$ for all $x \in X$, and also the subdifferential of the indicator function of X at x_{k+1}; see (Bertsekas, 2009, p. 185)). This is true if and only if

$$x_k - x_{k+1} - \alpha_k \tilde{\nabla} f(x_{k+1}) \in N_X(x_{k+1})$$

for some $\tilde{\nabla} f(x_{k+1}) \in \partial f(x_{k+1})$, which in turn is true if and only if (4.19) holds (cf. Bertsekas (2009, Proposition 5.4.6)).

(b) We have

$$\begin{aligned}\|x_k-y\|^2 &= \|x_k - x_{k+1} + x_{k+1} - y\|^2\\ &= \|x_k-x_{k+1}\|^2 - 2(x_k-x_{k+1})'(y-x_{k+1}) + \|x_{k+1}-y\|^2.\end{aligned} \tag{4.22}$$

Also since from (4.21), $\frac{1}{\alpha_k}(x_k - x_{k+1})$ is a subgradient at x_{k+1} of the sum of f and the indicator function of X, we have (also using the assumption $y \in X$) that

$$f(x_{k+1}) + \frac{1}{\alpha_k}(x_k-x_{k+1})'(y-x_{k+1}) \le f(y).$$

Combining this relation with (4.22), the result follows. □

Based on the preceding proposition, we see that all the preceding iterations can be written in an incremental subgradient format:

(a) Iteration (4.12)-(4.13) can be written as

$$z_k = P_X\big(x_k - \alpha_k \tilde{\nabla} f_{i_k}(z_k)\big), \qquad x_{k+1} = P_X\big(z_k - \alpha_k \tilde{\nabla} h_{i_k}(z_k)\big). \tag{4.23}$$

(b) Iteration (4.14)-(4.15) can be written as

$$z_k = x_k - \alpha_k \tilde{\nabla} f_{i_k}(z_k), \qquad x_{k+1} = P_X\big(z_k - \alpha_k \tilde{\nabla} h_{i_k}(z_k)\big). \tag{4.24}$$

(c) Iteration (4.16)-(4.17) can be written as

$$z_k = x_k - \alpha_k \tilde{\nabla} h_{i_k}(x_k), \qquad x_{k+1} = P_X\big(z_k - \alpha_k \tilde{\nabla} f_{i_k}(x_{k+1})\big). \tag{4.25}$$

In all the preceding updates, the subgradient $\tilde{\nabla} h_{i_k}$ can be *any* vector in the subdifferential of h_{i_k}, while the subgradient $\tilde{\nabla} f_{i_k}$ must be a *specific* vector in the subdifferential of f_{i_k}, specified according to Proposition 4.1(a). Also, iteration (4.24) can be written as

$$x_{k+1} = P_X\big(x_k - \alpha_k \tilde{\nabla} F_{i_k}(z_k)\big),$$

and resembles the incremental subgradient method for minimizing over X the cost $F(x) = \sum_{i=1}^m F_i(x)$ (cf. (4.10)), the only difference being that the subgradient of F_{i_k} is taken at z_k rather than x_k.

An important issue which affects the methods' effectiveness is the order in which the components $\{f_i, h_i\}$ are chosen for iteration. We consider two possibilities:

1. A *cyclic order*, whereby $\{f_i, h_i\}$ are taken up in the fixed deterministic order $1, \ldots, m$, so that i_k is equal to (k modulo m) plus 1. A contiguous block of iterations involving $\{f_1, h_1\}, \ldots, \{f_m, h_m\}$ in this order and exactly once is called a *cycle*. We assume that the stepsize α_k is constant within a cycle (for all k with $i_k = 1$ we have $\alpha_k = \alpha_{k+1} \ldots = \alpha_{k+m-1}$).

2. A *randomized order*, whereby at each iteration a component pair $\{f_i, h_i\}$ is chosen randomly by sampling over all component pairs with a uniform distribution, independently of the past history of the algorithm.

It is essential to include all components in a cycle in the cyclic case, and to sample according to the uniform distribution in the randomized case, for otherwise some components will be sampled more often than others, leading to a bias in the convergence process.

For the remainder of the chapter, we denote the optimal value of problem (4.10) by F^* :

$$F^* = \inf_{x \in X} F(x),$$

and the set of optimal solutions (which could be empty) by X^*:

$$X^* = \big\{x^* \mid x^* \in X,\, F(x^*) = F^*\big\}.$$

Also, for a nonempty closed set X, we denote by $\mathrm{dist}(\cdot; X)$ the distance function, defined as follows:

$$\mathrm{dist}(x; X) = \min_{z \in X} \|x - z\|, \qquad x \in \Re^n.$$

4.3 Convergence for Methods with Cyclic Order

In this section, we discuss convergence under the cyclic order. We consider a randomized order in the next section. We focus on the sequence $\{x_k\}$ rather than $\{z_k\}$, which need not lie within X in the case of iterations (4.24) and (4.25) when $X \neq \Re^n$. In summary, the idea is to show that the effect of taking subgradients of f_i or h_i at points near x_k (e.g., at z_k rather than at x_k) is inconsequential, and diminishes as the stepsize α_k becomes smaller, as long as some subgradients relevant to the algorithms are uniformly bounded in norm by some constant. This is similar to the convergence mechanism of incremental gradient methods described in Section 4.2. We use the following assumptions throughout the present section.

Assumption 4.1 (For iterations (4.23) and (4.24))**.** *There is a constant $c \in \Re$ such that for all k*

$$\max\left\{\|\tilde{\nabla} f_{i_k}(z_k)\|, \|\tilde{\nabla} h_{i_k}(z_k)\|\right\} \leq c. \tag{4.26}$$

Furthermore, for all k that mark the beginning of a cycle (i.e., all $k > 0$ with $i_k = 1$), we have for all $j = 1, \ldots, m$:

$$\max\left\{f_j(x_k) - f_j(z_{k+j-1}),\, h_j(x_k) - h_j(z_{k+j-1})\right\} \leq c\,\|x_k - z_{k+j-1}\|. \tag{4.27}$$

Assumption 4.2 (For iteration (4.25))**.** *There is a constant $c \in \Re$ such that for all k*

$$\max\left\{\|\tilde{\nabla} f_{i_k}(x_{k+1})\|, \|\tilde{\nabla} h_{i_k}(x_k)\|\right\} \leq c. \tag{4.28}$$

Furthermore, for all k that mark the beginning of a cycle (i.e., all $k > 0$ with $i_k = 1$), we have for all $j = 1, \ldots, m$:

$$\max\left\{f_j(x_k) - f_j(x_{k+j-1}),\, h_j(x_k) - h_j(x_{k+j-1})\right\} \leq c\,\|x_k - x_{k+j-1}\|, \tag{4.29}$$

$$f_j(x_{k+j-1}) - f_j(x_{k+j}) \leq c\,\|x_{k+j-1} - x_{k+j}\|. \tag{4.30}$$

The condition (4.27) is satisfied if for each i and k, there is a subgradient of f_i at x_k and a subgradient of h_i at x_k, whose norms are bounded by c. Conditions that imply the preceding assumptions are:

(a) For algorithm (4.23): f_i and h_i are Lipschitz continuous over the set X.

(b) For algorithms (4.24) and (4.25): f_i and h_i are Lipschitz continuous over the entire space $\Re^n$.

(c) For algorithms (4.23), (4.24), and (4.25): f_i and h_i are polyhedra (this

is a special case of (a) and (b)).

(d) The sequences $\{x_k\}$ and $\{z_k\}$ are bounded, since then, f_i and h_i, being real-valued and convex, are Lipschitz continuous over any bounded set that contains $\{x_k\}$ and $\{z_k\}$ (see, e.g., Bertsekas (2009, Proposition 5.4.2))].

The following proposition provides a key estimate that reveals the convergence mechanism of our methods.

Proposition 4.2. *Let $\{x_k\}$ be the sequence generated by any one of the algorithms (4.23)-(4.25), with a cyclic order of component selection. Then for all $y \in X$ and all k that mark the beginning of a cycle (i.e., all k with $i_k = 1$), we have*

$$\|x_{k+m} - y\|^2 \leq \|x_k - y\|^2 - 2\alpha_k\big(F(x_k) - F(y)\big) + \alpha_k^2 \beta m^2 c^2, \tag{4.31}$$

where $\beta = \frac{1}{m} + 4$ in the case of (4.23) and (4.24), and $\beta = \frac{5}{m} + 4$ in the case of (4.25).

Proof. We first prove the result for algorithms (4.23) and (4.24), and then indicate the modifications necessary for algorithm (4.25). Using Proposition 4.1(b), we have for all $y \in X$ and k,

$$\|z_k - y\|^2 \leq \|x_k - y\|^2 - 2\alpha_k\big(f_{i_k}(z_k) - f_{i_k}(y)\big). \tag{4.32}$$

Also, using the nonexpansion property of the projection (i.e., $\big\|P_X(u) - P_X(v)\big\| \leq \|u - v\|$ for all $u, v \in \Re^n$), the definition of subgradient, and (4.26), we obtain for all $y \in X$ and k:

$$\begin{aligned} \|x_{k+1} - y\|^2 &= \big\|P_X\big(z_k - \alpha_k \tilde{\nabla} h_{i_k}(z_k)\big) - y\big\|^2 \\ &\leq \|z_k - \alpha_k \tilde{\nabla} h_{i_k}(z_k) - y\|^2 \\ &\leq \|z_k - y\|^2 - 2\alpha_k \tilde{\nabla} h_{i_k}(z_k)'(z_k - y) + \alpha_k^2 \big\|\tilde{\nabla} h_{i_k}(z_k)\big\|^2 \\ &\leq \|z_k - y\|^2 - 2\alpha_k\big(h_{i_k}(z_k) - h_{i_k}(y)\big) + \alpha_k^2 c^2. \end{aligned} \tag{4.33}$$

Combining (4.32) and (4.33), and using the definition $F_j = f_j + h_j$, we have

$$\begin{aligned} \|x_{k+1} - y\|^2 &\leq \|x_k - y\|^2 - 2\alpha_k\big(f_{i_k}(z_k) + h_{i_k}(z_k) - f_{i_k}(y) - h_{i_k}(y)\big) + \alpha_k^2 c^2 \\ &= \|x_k - y\|^2 - 2\alpha_k\big(F_{i_k}(z_k) - F_{i_k}(y)\big) + \alpha_k^2 c^2. \end{aligned} \tag{4.34}$$

Now let k mark the beginning of a cycle (i.e., $i_k = 1$). Then, at iteration $k + j - 1$, $j = 1, \ldots, m$, the selected components are $\{f_j, h_j\}$, in view of the assumed cyclic order. We may thus replicate the preceding inequality with

k replaced by $k+1,\ldots,k+m-1$, and add to obtain

$$\|x_{k+m}-y\|^2 \le \|x_k-y\|^2 - 2\alpha_k \sum_{j=1}^{m} \big(F_j(z_{k+j-1}) - F_j(y)\big) + m\alpha_k^2 c^2$$

or, equivalently,

$$\|x_{k+m}-y\|^2 \le \|x_k-y\|^2 - 2\alpha_k\big(F(x_k)-F(y)\big) + m\alpha_k^2 c^2 + 2\alpha_k \sum_{j=1}^{m} \big(F_j(x_k) - F_j(z_{k+j-1})\big). \quad (4.35)$$

The remainder of the proof deals with appropriately bounding the last term above.

From (4.27), we have for $j=1,\ldots,m$ that

$$F_j(x_k) - F_j(z_{k+j-1}) \le 2c\,\|x_k - z_{k+j-1}\|. \quad (4.36)$$

We also have

$$\|x_k - z_{k+j-1}\| \le \|x_k - x_{k+1}\| + \cdots + \|x_{k+j-2} - x_{k+j-1}\| + \|x_{k+j-1} - z_{k+j-1}\|, \quad (4.37)$$

and by the definition of algorithms (4.23) and (4.24), the nonexpansion property of the projection, and (4.26), each of the terms in the right-hand side above is bounded by $2\alpha_k c$, except for the last, which is bounded by $\alpha_k c$. Thus (4.37) yields $\|x_k - z_{k+j-1}\| \le \alpha_k(2j-1)c$ which, together with (4.36), shows that

$$F_j(x_k) - F_j(z_{k+j-1}) \le 2\alpha_k c^2(2j-1). \quad (4.38)$$

Combining (4.35) and (4.38), we have

$$\|x_{k+m}-y\|^2 \le \|x_k-y\|^2 - 2\alpha_k\big(F(x_k)-F(y)\big) + m\alpha_k^2 c^2 + 4\alpha_k^2 c^2 \sum_{j=1}^{m}(2j-1),$$

and finally

$$\|x_{k+m}-y\|^2 \le \|x_k-y\|^2 - 2\alpha_k\big(F(x_k)-F(y)\big) + m\alpha_k^2 c^2 + 4\alpha_k^2 c^2 m^2,$$

which is of the form (4.31) with $\beta = \frac{1}{m} + 4$.

For algorithm (4.25), a similar argument goes through using Assumption 4.2. In place of (4.32), using the nonexpansion property of the projection, the definition of subgradient, and (4.28), we obtain, for all $y \in X$ and $k \ge 0$,

$$\|z_k - y\|^2 \le \|x_k - y\|^2 - 2\alpha_k\big(h_{i_k}(x_k) - h_{i_k}(y)\big) + \alpha_k^2 c^2. \quad (4.39)$$