SIAM J. OPTIM.
Vol. 7, No. 4, pp. 913–926, November 1997

002

A NEW CLASS OF INCREMENTAL GRADIENT METHODS FOR LEAST SQUARES PROBLEMS*

DIMITRI P. BERTSEKAS†

Abstract. The least mean squares (LMS) method for linear least squares problems differs from the steepest descent method in that it processes data blocks one-by-one, with intermediate adjustment of the parameter vector under optimization. This mode of operation often leads to faster convergence when far from the eventual limit and to slower (sublinear) convergence when close to the optimal solution. We embed both LMS and steepest descent, as well as other intermediate methods, within a one-parameter class of algorithms, and we propose a hybrid class of methods that combine the faster early convergence rate of LMS with the faster ultimate linear convergence rate of steepest descent. These methods are well suited for neural network training problems with large data sets. Furthermore, these methods allow the effective use of scaling based, for example, on diagonal or other approximations of the Hessian matrix.

Key words. gradient methods, nonlinear programming, least squares, neural networks

AMS subject classifications. 49N10, 65K05, 65K10, 65F20

PII. S1052623495287022

1. Introduction.

We consider least squares problems of the form

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

where $\Re^n$ denotes the n-dimensional Euclidean space and $f_i : \Re^n \to \Re$ are continuously differentiable scalar functions on $\Re^n$. A special case of particular interest to us is the least squares problem

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\sum_{i=1}^{m} \|g_i(x)\|^2 \\ &\text{subject to} \quad x \in \Re^n, \end{aligned}$$

where $g_i : \Re^n \to \Re^{r_i}$, $i = 1, \ldots, m$, are continuously differentiable functions. Here we write $\|z\|$ for the usual Euclidean norm of a vector z; that is, $\|z\| = \sqrt{z'z}$, where prime denotes transposition. We also write ∇f and ∇f_i for the gradients of the functions f and f_i, respectively. Least squares problems often arise in contexts where the functions g_i correspond to data that we are trying to fit with a model parameterized by x. Motivated by this context, we refer to each component f_i as a *data block*, and we refer to the entire collection $(f_1, \ldots, f_m)$ as the *data set*.

In problems where there are many data blocks, and particularly in neural network training problems, gradient-like incremental methods are frequently used. In such methods, one does not wait to process the entire data set before updating x; instead, one cycles through the data blocks in sequence and updates the estimate of x after

*Received by the editors June 2, 1995; accepted for publication (in revised form) October 7, 1996. This research was supported by NSF grant 9300494-DMI.
http://www.siam.org/journals/siopt/7-4/28702.html
†Deptartment of Electrical Engineering and Computer Science, M.I.T., Cambridge, MA 02139 (dimitrib@mit.edu).

each data block is processed. Such methods include the Widrow–Hoff LMS algorithm [WiH60], [WiS85], for the case of a linear least squares problem, and its extension to nonlinear least squares problems. A cycle through the data set of this method starts with a vector x^k and generates x^{k+1} according to

$$x^{k+1} = \psi_m,$$

where ψ_m is obtained at the last step of the recursion

$$\psi_0 = x^k, \qquad \psi_i = \psi_{i-1} - \alpha^k \nabla f_i(\psi_{i-1}), \quad i = 1, \ldots, m, \tag{2}$$

and α^k is a positive stepsize. Thus the method has the form

$$x^{k+1} = x^k - \alpha^k \sum_{i=1}^{m} \nabla f_i(\psi_{i-1}). \tag{3}$$

We refer to this method, which is just the nonlinear version of the LMS algorithm, as the *incremental gradient method*.

The above method should be contrasted with the steepest descent method, where the data blocks f_i and their gradients are evaluated at the same vector x^k, that is,

$$\psi_0 = x^k, \qquad \psi_i = \psi_{i-1} - \alpha^k \nabla f_i(x^k), \quad i = 1, \ldots, m, \tag{4}$$

so that the iteration consisting of a cycle over the entire data set starting from x^k has the form

$$x^{k+1} = x^k - \alpha^k \sum_{i=1}^{m} \nabla f_i(x^k) = x^k - \alpha^k \nabla f(x^k). \tag{5}$$

Incremental methods are supported by stochastic convergence analyses [PoT73], [Lju77], [KuC78], [TBA86], [Pol87], [BeT89], [Whi89], [Gai94], [BeT96] as well as deterministic convergence analyses [Luo91], [Gri94], [LuT94], [MaS94], [Man93], [Ber95a], [BeT96]. It has been experimentally observed that the incremental gradient method (2)–(3) often converges much faster than the steepest descent method (5) when far from the eventual limit. However, near convergence, the incremental gradient method typically converges slowly because it requires a diminishing stepsize $\alpha^k = O(1/k)$ for convergence. If α^k is instead taken to be a small constant, an oscillation within each data cycle arises, as shown by [Luo91]. By contrast, for convergence of the steepest descent method, it is sufficient that the stepsize α^k is a small constant (this requires that ∇f be Lipschitz continuous; see, e.g., [Pol87]). The asymptotic convergence rate of steepest descent with a constant stepsize is typically linear and much faster than that of the incremental gradient method.

The behavior described above is most vividly illustrated in the case of a linear least squares problem where the vector x is one dimensional, as shown in the following example.

Example 1. Consider the least squares problem

$$\begin{aligned} &\text{minimize} \quad f(x) = \frac{1}{2} \sum_{i=1}^{m} (a_i x - b_i)^2 \\ &\text{subject to} \quad x \in \Re, \end{aligned} \tag{6}$$

where a_i and b_i are given scalars with $a_i \neq 0$ for all i. The minimum of each of the data blocks

$$f_i(x) = \frac{1}{2}(a_i x - b_i)^2 \tag{7}$$

is

$$x_i^* = \frac{b_i}{a_i},$$

while the minimum of the least squares cost function f is

$$x^* = \frac{\sum_{i=1}^m a_i b_i}{\sum_{i=1}^m a_i^2}.$$

It can be seen that x^* lies within the range of the data block minima

$$R = \left[\min_i x_i^*, \max_i x_i^*\right] \tag{8}$$

and that for all x *outside* the range R the gradient

$$\nabla f_i(x) = a_i(a_i x - b_i)$$

has the same sign as $\nabla f(x)$. As a result, the incremental gradient method given by

$$\psi_i = \psi_{i-1} - \alpha^k \nabla f_i(\psi_{i-1}) \tag{9}$$

(cf. (2)) approaches x^* at each step provided the stepsize α^k is small enough. In fact it is sufficient that

$$\alpha^k \leq \min_i \frac{1}{a_i^2}. \tag{10}$$

However, for x *inside* the region R, the ith step of a cycle of the incremental gradient method, given by (9), need not make progress because it aims to approach x_i^* but not necessarily x^*. It will approach x^* (for small enough stepsize α^k) only if the current point ψ_{i-1} does not lie in the interval connecting x_i^* and x^*. This induces an oscillatory behavior within the region R, and as a result the incremental gradient method will typically not converge to x^* unless $\alpha^k \to 0$. By contrast, it can be shown that the steepest descent method, which takes the form

$$x^{k+1} = x^k - \alpha^k \sum_{i=1}^m a_i(a_i x^k - b_i),$$

converges to x^* for any constant stepsize satisfying

$$\alpha^k \leq \frac{2}{\sum_{i=1}^m a_i^2}. \tag{11}$$

However, unless the stepsize choice is particularly favorable, for x outside the region R, a full iteration of steepest descent need not make more progress toward the solution than a single step of the incremental gradient method. In other words, *far from the solution (outside R), a single pass through the entire data set by the incremental*

gradient method is roughly as effective as m passes through the data set by the steepest descent method.

The analysis of the preceding example relies on x being one dimensional, but in many multidimensional problems the same qualitative behavior can be observed. In particular, a pass through the ith data block f_i by the incremental gradient method can make progress toward the solution in the region where the data block gradient $\nabla f_i(\psi_{i-1})$ makes an angle less than 90 degrees with the cost function gradient $\nabla f(\psi_{i-1})$. If the data blocks f_i are not "too dissimilar," this is likely to happen in a region that is not too close to the optimal solution set. For example, consider the case of a linear least squares problem

$$f_i(x) = \frac{1}{2}\|A_i x - b_i\|^2, \tag{12}$$

where the vectors b_i and the matrices A_i are given. Then, it can be shown that sufficiently far from the optimal solution, the direction $\nabla f_i(x)$ used at the ith step of a data cycle of the incremental gradient method will be a descent direction for the entire cost function f if the matrix $A_i' A_i \sum_{j=1}^m A_j' A_j$ is positive definite in the sense that

$$x' A_i' A_i \left(\sum_{j=1}^m A_j' A_j \right) x > 0 \qquad \forall\, x \neq 0. \tag{13}$$

This will be true if the matrices A_i are sufficiently close to each other with respect to some matrix norm. One may also similarly argue on a heuristic basis that the incremental gradient method will be substantially more effective than the steepest descent method far from the solution if the above relation holds for a substantial majority of the indices i.

It is also worth mentioning that a similar argument can be made in favor of incremental versions of the Gauss–Newton method for least squares problems. These methods are closely related to the extended Kalman filter algorithm that is used extensively in control and estimation contexts; see, e.g., [Ber95b], [Bel94], [Dav76], [WaT90]. However, like the incremental gradient method, incremental Gauss–Newton methods also suffer from slow ultimate convergence because for convergence they require a diminishing stepsize [Ber95b]. Furthermore, for difficult least squares problems, such as many neural network training problems, it is unclear whether Gauss–Newton methods hold any advantage over gradient methods.

In this paper we introduce a class of gradient-like methods parameterized by a single nonnegative constant μ. For the two extreme values $\mu = 0$ and $\mu = \infty$, we obtain as special cases the incremental gradient and steepest descent methods, respectively. Positive values of μ yield hybrid methods with varying degrees of incrementalism in processing the data blocks. We also propose a time-varying hybrid method, where μ is gradually increased from $\mu = 0$ toward $\mu = \infty$. This method aims to combine the typically faster initial convergence rate of incremental gradient with the faster ultimate convergence rate of steepest descent. It starts out as the incremental gradient method (2)–(3), but gradually (based on algorithmic progress) it becomes less and less incremental, and asymptotically it approaches the steepest descent method (5). In contrast to the incremental gradient method, it uses a constant stepsize without resulting in an asymptotic oscillation. We prove convergence and a linear rate of convergence for this method in the case where the data blocks are positive semidefinite

quadratic functions. Similar results can be shown for the case of nonquadratic data blocks and a parallel asynchronous computing environment.

In addition to a linear convergence rate, the use of a constant stepsize offers another important practical advantage: it allows a more effective use of scaling based, for example, on approximations of the Hessian matrix. Our experience shows that our method performs better than both the incremental gradient and the steepest descent method, particularly when scaling is used.

2. The new incremental gradient method. We embed the incremental gradient method (2)–(3) and the steepest descent method (5) within a one-parameter family of methods for the least squares problem. Let us fix a scalar $\mu \geq 0$. Consider the method which given x^k generates x^{k+1} according to

$$x^{k+1} = \psi_m, \tag{14}$$

where ψ_m is generated at the last step of the algorithm

$$\psi_i = x^k - \alpha^k h_i, \qquad i = 1, \ldots, m, \tag{15}$$

and the vectors h_i are defined as follows:

$$h_i = \sum_{j=1}^{i} w_{ij}(\mu) \nabla f_j(\psi_{j-1}), \qquad i = 1, \ldots, m, \tag{16}$$

where

$$\psi_0 = x^k, \tag{17}$$

and

$$w_{ij}(\mu) = \frac{1 + \mu + \cdots + \mu^{i-j}}{1 + \mu + \cdots + \mu^{m-j}}, \qquad i = 1, \ldots, m,\ 1 \leq j \leq i. \tag{18}$$

It can be verified using induction that the vectors h_i can be generated recursively using the formulas

$$h_i = \mu h_{i-1} + \sum_{j=1}^{i} \xi_j(\mu) \nabla f_j(\psi_{j-1}), \qquad i = 1, \ldots, m, \tag{19}$$

where $h_0 = 0$ and

$$\xi_i(\mu) = \frac{1}{1 + \mu + \cdots + \mu^{m-i}}, \qquad i = 1, \ldots, m. \tag{20}$$

Thus the computation of h_i using (19) requires (essentially) no more storage or overhead per iteration than either the steepest descent method (5) or the incremental gradient method (2)–(3).

Note that since

$$w_{mj}(\mu) = 1, \qquad j = 1, \ldots, m,$$

it follows using (15)–(16) that the vector ψ_m obtained at the end of a pass through all the data blocks is

$$\psi_m = x^{k+1} = x^k - \alpha^k h_m = x^k - \alpha^k \sum_{j=1}^{m} \nabla f_j(\psi_{j-1}). \tag{21}$$

In the special case where $\mu = 0$, we have $w_{ij}(\mu) = 1$ for all i and j, and by comparing (15), (18), (2), and (3) we see that the method coincides with the incremental gradient method (2)–(3). In the case where $\mu \to \infty$, we have from (15), (18), and (19) $w_{ij}(\mu) \to 0$, $h_i \to 0$, and $\psi_i \to x^k$ for $i = 0, 1, \ldots, m-1$, so by comparing (21) and (5) we see that the method approaches the steepest descent method (5). Generally, it can be seen that as μ increases the method becomes "less incremental."

We first prove a convergence result for the method (13)–(17) for the case where μ is fixed and each data block f_i is positive semidefinite quadratic. This covers the case of a linear least squares problem. In particular, we show that if the stepsize α^k is a sufficiently small constant, the algorithm asymptotically oscillates around the optimal solution. However, the "size" of the oscillation diminishes as either $\alpha \to 0$ and μ is constant or as α is constant and $\mu \to \infty$. If the stepsize is diminishing of the form $\alpha^k = O(1/k)$, the method converges to the minimum for all values of μ.

PROPOSITION 2.1. *Suppose that the functions f_i have the form*

$$f_i(x) = \frac{1}{2} x' Q_i x - c_i' x, \qquad i = 1, \ldots, m,$$

where Q_i are given positive semidefinite symmetric matrices and c_i are given vectors. Consider the algorithm (cf. (13)–(17))

$$x^{k+1} = \psi_m, \tag{22}$$

where

$$\psi_0 = x^k, \qquad \psi_i = x^k - \alpha^k h_i, \quad i = 1, \ldots, m, \tag{23}$$

$$h_0 = 0, \qquad h_i = \mu h_{i-1} + \sum_{j=1}^{i} \xi_j(\mu)(Q_j \psi_{j-1} - c_j), \quad i = 1, \ldots, m. \tag{24}$$

Assume that $\sum_{j=1}^m Q_j$ is a positive definite matrix, and let x^* be the optimal solution of (1). Then the following hold:

(a) *For each $\mu \geq 0$, there exists $\overline{\alpha}(\mu) > 0$ such that if α^k is equal to some constant $\alpha \in (0, \overline{\alpha}(\mu)]$ for all k, $\{x^k\}$ converges to some vector $x(\alpha, \mu)$, and we have $\lim_{\alpha \to 0^+} x(\alpha, \mu) = x^*$. Furthermore, there exists $\overline{\alpha} > 0$ such that $\overline{\alpha} \leq \overline{\alpha}(\mu)$ for all $\mu \geq 0$, and for all $\alpha \in (0, \overline{\alpha}]$ we have $\lim_{\mu \to \infty} x(\alpha, \mu) = x^*$.*

(b) *For each $\mu \geq 0$, if $\alpha^k > 0$ for all k and*

$$\alpha^k \to 0, \qquad \sum_{k=0}^{\infty} \alpha^k = \infty, \tag{25}$$

then $\{x^k\}$ converges to x^*.

Proof. (a) We first note that from (21) we have

$$x^{k+1} = x^k - \alpha \sum_{j=1}^{m} (Q_j \psi_{j-1} - c_j),$$

so by using the definition $\psi_{j-1} = x^k - \alpha h_{j-1}$ we obtain

$$x^{k+1} = x^k - \alpha \sum_{j=1}^{m} (Q_j x^k - c_j) + \alpha^2 \sum_{j=1}^{m} Q_j h_{j-1}. \tag{26}$$

We next observe that from (18) and the definition $\psi_{j-1} = x^k - \alpha h_{j-1}$ we have for all i

$$h_i = \sum_{j=1}^{i} w_{ij}(\mu)(Q_j\psi_{j-1} - c_j) \tag{27}$$
$$= \sum_{j=1}^{i} w_{ij}(\mu)Q_j x^k - \alpha\sum_{j=1}^{i} w_{ij}(\mu)Q_j h_{j-1} - \sum_{j=1}^{i} w_{ij}(\mu)c_j.$$

From this relation it can be seen inductively that for all i, h_i can be written as

$$h_i = \sum_{j=1}^{i} w_{ij}(\mu)Q_j x^k - \sum_{j=1}^{i} w_{ij}(\mu)c_j + \alpha R_i(\alpha,\mu)x^k + \alpha r_i(\alpha,\mu), \tag{28}$$

where $R_i(\alpha,\mu)$ and $r_i(\alpha,\mu)$ are some matrices and vectors, respectively, depending on α and μ. Furthermore, using (27) and the fact that $w_{ij}(\mu) \in (0,1]$ for all i, j, and $\mu \geq 0$, we have that for any bounded interval T of stepsizes α there exist positive uniform bounds $\overline{R}$ and $\overline{r}$ for $\|R_i(\alpha,\mu)\|$ and $\|r_i(\alpha,\mu)\|$; that is,

$$\|R_i(\alpha,\mu)\| \leq \overline{R}, \qquad \|r_i(\alpha,\mu)\| \leq \overline{r} \qquad \forall\ i,\ \mu \geq 0,\ \alpha \in T. \tag{29}$$

From (26), (28), and (29) we obtain

$$x^{k+1} = A(\alpha,\mu)x^k + b(\alpha,\mu), \tag{30}$$

where

$$A(\alpha,\mu) = I - \alpha\sum_{j=1}^{m} Q_j + \alpha^2 S(\alpha,\mu), \tag{31}$$

$$b(\alpha,\mu) = \alpha\sum_{j=1}^{m} c_j + \alpha^2 s(\alpha,\mu), \tag{32}$$

I is the identity matrix, and the matrix $S(\alpha,\mu)$ and the vector $s(\alpha,\mu)$ are uniformly bounded over $\mu \geq 0$ and any bounded interval T of stepsizes α; that is, for some scalars $\overline{S}$ and $\overline{s}$,

$$\|S(\alpha,\mu)\| \leq \overline{S}, \qquad \|s(\alpha,\mu)\| \leq \overline{s} \qquad \forall\ \mu \geq 0,\ \alpha \in T. \tag{33}$$

Let us choose the interval T to contain small enough stepsizes so that for all $\mu \geq 0$ and $\alpha \in T$, the eigenvalues of $A(\alpha,\mu)$ are all strictly within the unit circle; this is possible since $\sum_{j=1}^{m} Q_j$ is assumed positive definite and (31) and (33) hold. Define

$$x(\alpha,\mu) = \big(I - A(\alpha,\mu)\big)^{-1} b(\alpha,\mu). \tag{34}$$

Then $b(\alpha,\mu) = \big(I - A(\alpha,\mu)\big)x(\alpha,\mu)$, and by substituting this expression in (30) it can be seen that

$$x^{k+1} - x(\alpha,\mu) = A(\alpha,\mu)\big(x^k - x(\alpha,\mu)\big),$$

from which

$$x^{k+1} - x(\alpha,\mu) = A(\alpha,\mu)^k\big(x^0 - x(\alpha,\mu)\big) \qquad \forall\ k.$$

Since all the eigenvalues of $A(\alpha,\mu)$ are strictly within the unit circle, we have $A(\alpha,\mu)^k \to 0$, so $x^k \to x(\alpha,\mu)$.

To prove that $\lim_{\alpha\to 0} x(\alpha,\mu) = x^*$, we first calculate x^*. We set the gradient of f to 0 to obtain

$$\sum_{j=1}^{m}(Q_j x^* - c_j) = 0,$$

so that

$$x^* = \left(\sum_{j=1}^{m} Q_j\right)^{-1} \sum_{i=1}^{m} c_j. \tag{35}$$

Then we use (34) to write $x(\alpha,\mu) = \big(I/\alpha - A(\alpha,\mu)/\alpha\big)^{-1}\big(b(\alpha,\mu)/\alpha\big)$, and we see from (31) and (32) that

$$\lim_{\alpha\to 0} x(\alpha,\mu) = \left(\sum_{j=1}^{m} Q_j\right)^{-1} \sum_{i=1}^{m} c_j = x^*.$$

To prove that $\lim_{\mu\to\infty} x(\alpha,\mu) = x^*$, we note that since $\lim_{\mu\to\infty} w_{ij}(\mu) = 0$ for $i = 1,\ldots,m-1$, it follows from (16) that h_{j-1} tends to 0 as $\mu \to \infty$ for $j = 1,\ldots,m-1$. Using this fact in conjunction with (26) and (30)–(32) it follows that

$$\lim_{\mu\to\infty} S(\alpha,\mu) = 0, \qquad \lim_{\mu\to\infty} s(\alpha,\mu) = 0.$$

From (31), (32), and (34) we then obtain

$$\lim_{\mu\to\infty} x(\alpha,\mu) = \left(\alpha\sum_{j=1}^{m} Q_j\right)^{-1} \left(\alpha\sum_{j=1}^{m} c_j\right) = x^*.$$

(b) We need the following well-known lemma (for a proof, see [Luo91], [Ber95a], [BeT96]).

LEMMA 2.1. *Suppose that $\{e^k\}$ and $\{\gamma^k\}$ are nonnegative sequences and c is a positive constant such that*

$$e^{k+1} \le (1-\gamma^k)e^k + c(\gamma^k)^2, \quad \gamma^k \le 1, \qquad k = 0, 1\ldots,$$

and

$$\gamma^k \to 0, \qquad \sum_{k=0}^{\infty}\gamma^k = \infty.$$

Then $e^k \to 0$.

Returning to the proof of Proposition 2.1, from (21) and (30)–(32) we have

$$x^{k+1} = x^k - \alpha^k \sum_{j=1}^{m} (Q_j x^k - c_j) + (\alpha^k)^2 S(\alpha^k, \mu)(x^k - x^*) + (\alpha^k)^2 e^k, \tag{36}$$

where

$$e^k = S(\alpha^k, \mu)x^* + s(\alpha^k, \mu). \tag{37}$$

Using also the expression (35) for x^*, we can write (36) as

$$x^{k+1} - x^* = \left(I - \alpha^k \sum_{j=1}^{m} Q_j + (\alpha^k)^2 S(\alpha^k, \mu) \right) (x^k - x^*) + (\alpha^k)^2 e^k. \tag{38}$$

For large enough k, the eigenvalues of $\alpha^k \sum_{j=1}^m Q_j$ are bounded from above by 1, and hence the matrix $I - \alpha^k \sum_{j=1}^m Q_j$ is positive definite. Without loss of generality, we assume that this is so for all k. Then we have

$$\left\| \left(I - \alpha^k \sum_{j=1}^{m} Q_j \right) (x^k - x^*) \right\| \leq (1 - \alpha^k A)\|x^k - x^*\|, \tag{39}$$

where A is the smallest eigenvalue of $\sum_{j=1}^m Q_j$. Let also B and δ be positive scalars such that for all k we have

$$\left\| S(\alpha^k, \mu)(x^k - x^*) \right\| \leq B\|x^k - x^*\|, \qquad \|e^k\| \leq \delta. \tag{40}$$

Combining (38)–(40), we have

$$\begin{aligned} \|x^{k+1} - x^*\| &\leq \left\| \left(I - \alpha^k \sum_{j=1}^{m} Q_j \right) (x^k - x^*) \right\| + (\alpha^k)^2 \left\| S(\alpha^k, \mu)(x^k - x^*) \right\| + (\alpha^k)^2 \|e^k\| \\ &\leq (1 - \alpha^k A + (\alpha^k)^2 B)\|x^k - x^*\| + (\alpha^k)^2 \delta. \end{aligned} \tag{41}$$

Let $\overline{k}$ be such that $\alpha^k B \leq A/2$ for all $k \geq \overline{k}$. Then from (41) we obtain

$$\|x^{k+1} - x^*\| \leq (1 - \alpha^k A/2)\|x^k - x^*\| + (\alpha^k)^2 \delta \qquad \forall\ k \geq \overline{k},$$

and Lemma 2.1 can be used to show that $\|x^k - x^*\| \to 0$. □

The following proposition shows that if μ is increased toward ∞ at a sufficiently fast rate, the sequence $\{x^k\}$ generated by the method with a constant stepsize converges at a linear rate.

PROPOSITION 2.2. *Suppose that in the kth iteration of the method* (14)–(18)*, a k-dependent value of μ, say $\mu(k)$, and a constant stepsize $\alpha^k = \alpha$ are used. Under the assumptions of Proposition* 2.1*, if for some $q > 1$ and all k greater than some index $\overline{k}$, we have $\mu(k) \geq q^k$, then there exists $\overline{\alpha} > 0$ such that for all $\alpha \in (0, \overline{\alpha}]$ and k we have $\|x^k - x^*\| \leq p(\alpha)\beta(\alpha)^k$, where $p(\alpha) > 0$ and $\beta(\alpha) \in (0, 1)$ are some scalars depending on α.*

Proof. We first note that the proof of Proposition 2.1(a) can be modified to show that there exists $\overline{\alpha} > 0$ such that for all $\alpha \in (0, \overline{\alpha}]$ we have $x^k \to x^*$. We also note

that if for some $q > 1$, we have $\mu(k) \geq q^k$ for k after some index $\overline{k}$, then for all $i < m$ and $j \leq i$ we have

$$w_{ij}\big(\mu(k)\big) = O(\gamma^k), \tag{42}$$

where γ is some scalar with $\gamma \in (0,1)$.

We next observe that similar to the derivation of (38) we have

$$x^{k+1} - x^* = \left(I - \alpha \sum_{j=1}^{m} Q_j + \alpha^2 S\big(\alpha, \mu(k)\big)\right)(x^k - x^*) + \alpha^2 e^k, \tag{43}$$

where

$$e^k = S\big(\alpha, \mu(k)\big)x^* + s\big(\alpha, \mu(k)\big). \tag{44}$$

From (27), we see that h_i can be written as a finite number of terms of bounded norm, which are multiplied by some term $w_{ij}(\mu(k))$. Thus, in view of (42), for $i < m$ we have $\|h_i\| = O(\gamma^k)$, which by comparing (27) and (28) implies that for all i

$$\|R_i\big(\alpha, \mu(k)\big)\| = O(\gamma^k), \qquad \|r_i\big(\alpha, \mu(k)\big)\| = O(\gamma^k).$$

It follows that

$$\|S\big(\alpha, \mu(k)\big)\| = O(\gamma^k), \qquad \|s\big(\alpha, \mu(k)\big)\| = O(\gamma^k). \tag{45}$$

From (44) we then obtain

$$\|e^k\| = O(\gamma^k). \tag{46}$$

From (43), (45), and (46), we obtain

$$\|x^{k+1} - x^*\| \leq \big(|1 - \alpha\delta| + O(\gamma^k)\big)\|x^k - x^*\| + \alpha^2 O(\gamma^k),$$

where δ is the minimum eigenvalue of $\sum_{j=1}^{m} Q_j$. This relation implies the desired rate of convergence result. □

There are a number of fairly straightforward extensions of the methods and the results just presented.

(1) When the data blocks are nonquadratic, stationarity of the limit points of sequences $\{x^k\}$ generated by the method (13)–(17) can be shown under certain assumptions (including Lipschitz continuity of the data block gradients) for the case of a fixed μ and the stepsize $\alpha^k = \gamma/(k+\delta)$, where γ and δ are positive scalars. Contrary to the case of quadratic data blocks, γ may have to be chosen sufficiently small to guarantee boundedness of $\{x^k\}$. The convergence proof is similar to the one of the preceding proposition, but it is technically more involved. In the case where the stepsize is constant, $\mu \to \infty$, and the data blocks are nonquadratic, it is also possible to show a result analogous to Proposition 2.2, but again the proof is technically complex and will not be given.

(2) Convergence results for parallel asynchronous versions of our method can be given, in the spirit of those in [TBA86], [BeT89, Chap. 7], and [MaS94]. These results follow well-established methods of analysis that rely on the stepsize being sufficiently small.

(3) Variations of our method involving a quadratic momentum term are possible. The use of such terms dates to the heavy ball method of Poljak (see [Pol64], [Pol87], [Ber95a]) in connection with the steepest descent method and has become popular in the context of the incremental gradient method, particularly for neural network training problems (see [MaS94] for an analysis).

(4) Diagonal scaling of the iterations generating ψ_i is possible by replacing the equation $\psi_i = x^k - \alpha^k h_i$ (cf. (15)) with the equation

$$\psi_i = x^k - \alpha^k D h_i, \qquad i = 1, \ldots, m,$$

where D is a positive-definite symmetric matrix. A common approach is to use a diagonal matrix D whose diagonal elements are the inverses of the corresponding diagonal elements of the Hessian of the cost function

$$\sum_{j=1}^{m} \nabla^2 f_j(\psi_{j-1}).$$

An important advantage of this type of diagonal scaling is that it simplifies the choice of a constant stepsize; a value of stepsize equal to 1 or a little smaller typically works well. Diagonal scaling is often beneficial for steepest descent-like methods that use a constant stepsize but is not as helpful for the incremental gradient method because the latter uses a variable (diminishing) stepsize. For this reason diagonal scaling should be typically more effective for the constant stepsize methods proposed here than for the incremental gradient method. This was confirmed in our computational experiments; see also the discussion of the next section. For this reason, we believe that for problems where diagonal scaling is important for good performance our constant stepsize methods have a significant advantage over the LMS and the incremental gradient methods.

3. Implementation and experimentation. Let us consider algorithms where μ is iteration dependent and is increased with k toward ∞. While Proposition 2.2 suggests that a linear convergence rate can be obtained by keeping α constant, we have found in our experimentation that it may be important to change α simultaneously with μ when μ is still relatively small. In particular, as the problem of Example 1 suggests, when μ is near 0 and the method is similar to the incremental gradient method, the stepsize should be larger, while when μ is large, the stepsize should be of comparable magnitude to the corresponding stepsize of steepest descent.

The formula for $\xi_i(\mu)$ suggests that for $\mu \leq 1$ the incremental character of the method is strong, so we have experimented with a μ-dependent stepsize formula of the form

$$\alpha(\mu) = \begin{cases} \gamma & \text{if } \mu > 1, \\ \big(1 + \phi(\mu)\big)\gamma & \text{if } \mu \in [0, 1]. \end{cases} \tag{47}$$

Here γ is the stepsize that works well with the steepest descent method and should be determined to some extent by trial and error (if diagonal scaling is used, then a choice of γ close to 1 often works well). The function $\phi(\mu)$ is a monotonically decreasing function with

$$\phi(0) = \zeta, \qquad \phi(1) = 0, \tag{48}$$

where ζ is a scalar in the range $[0, m-1]$. Examples are

$$\phi(\mu) = \zeta(1-\mu), \qquad \phi(\mu) = \zeta(1-\mu^2), \qquad \phi(\mu) = \zeta(1-\sqrt{\mu}). \tag{49}$$

In some of the variations of the method that we experimented with, the scalar ζ was decreased by a certain factor each time μ was increased. Generally, with μ-dependent stepsize selection of the form (49) and diagonal scaling, we have found the constant stepsize methods proposed here far more effective than the incremental gradient method that uses the same diagonal scaling and a diminishing stepsize.

Regarding the rule for increasing μ, we have experimented with schemes that start with $\mu = 0$ and update μ according to a formula of the form

$$\mu := \beta\mu + \delta,$$

where β and δ are fixed positive scalars with $\beta > 1$. The update of μ takes place at the start of a data cycle following the computation of x^{k+1} if either

$$\|x^{k+1} - x^k\| \leq \epsilon, \tag{50}$$

where ϵ is a fixed tolerance, or $\hat{n}$ data cycles have been performed since the last update of μ, where $\hat{n}$ is an integer chosen by trial and error. This criterion tries to update μ when the method appears to be making little further progress at the current level of μ but also updates μ after a maximum specified number $\hat{n}$ of data cycles have been performed with the current μ.

We noted one difficulty with the method. When the number of data blocks m is large, the calculation of $\xi_i(\mu)$ using (20) involves high powers of μ. This tends to introduce substantial numerical error when μ is substantially larger than 1. To get around this difficulty, we modified the method by lumping together an increasing number of data blocks (the minimum number of terms in a data block was incremented by 1) each time μ was increased to a value above 1. This device effectively reduces the number of data blocks m and keeps the power μ^m bounded. In our computational experiments, it has eliminated the difficulty with numerical errors without substantially affecting the performance of the method.

Finally, let us try to compare the diagonally scaled version of our method with the diagonally scaled incremental gradient method given by

$$x^{k+1} = x^k - \alpha^k D \sum_{j=1}^{m} \nabla f_j(\psi_{j-1}), \tag{51}$$

where ψ_i is generated by

$$\psi_i = x^k - \alpha^k D \sum_{j=1}^{i} \nabla f_j(\psi_{j-1}). \tag{52}$$

We assume that D is a diagonal approximation of the inverse Hessian of f. It is difficult to draw definitive conclusions regarding the two methods because their performance depends a lot on various tuning parameters. In particular, it is very difficult to compare the methods using computational results with only a few test problems, and this will not be attempted. On the other hand, it is helpful to consider some extreme problem cases.

(1) Problems where diagonal scaling is effective because the Hessian matrix of f is nearly diagonal. For such problems, both methods can be very fast with proper tuning of the stepsize parameters. On the other hand the incremental gradient method after a few iterations slows down because of the diminishing stepsize. By contrast, our method maintains its rate of convergence, and, indeed, once μ reaches high values and when $\alpha^k \approx 1$, it may become even faster than in the early iterations where μ is small, because for large μ it effectively approximates Newton's method.

(2) Problems where diagonal scaling is ineffective because the Hessian matrix of f is not nearly diagonal and is ill conditioned. Then both methods will likely be slow regardless of how they are tuned. On the other hand the convergence rate of the incremental gradient method will continually deteriorate because of the diminishing stepsize, while our method will at least maintain a (slow) linear convergence rate.

(3) Problems that do not fall in the preceding categories but which have "homogeneous" data blocks, that is, problems where the Hessian matrices $\nabla^2 f_i$ of the data blocks are not too dissimilar. Then incrementalism is likely to be very beneficial (think of the extreme case where all the data blocks are identical). For such problems the incremental gradient method may have an edge in the early iterations because of its greater degree of incrementalism, although asymptotically our method maintains the advantage of the linear convergence rate.

(4) Problems that do not fall in the preceding categories, but which have "inhomogeneous" data blocks, where the Hessian matrices $\nabla^2 f_i$ of the data blocks are quite dissimilar. Then our method is likely to have an advantage over the incremental gradient method, because it gradually becomes nonincremental, while maintaining a nondiminishing stepsize and the attendant linear convergence rate.

The preceding arguments, while speculative, are consistent with the results of the author's experimentation. However, a far more comprehensive experimentation as well as experience with real-world problems is needed to support the preceding conclusions and to assess more reliably the merits of the method proposed.

REFERENCES

[BeT89] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice–Hall, Englewood Cliffs, NJ, 1989.

[BeT96] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA, 1996.

[Bel94] B. M. Bell, *The iterated Kalman smoother as a Gauss–Newton method*, SIAM J. Optim., 4 (1994), pp. 626–636.

[Ber95a] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, Belmont, MA, 1995.

[Ber95b] D. P. Bertsekas, *Incremental least squares methods and the extended Kalman filter*, SIAM J. Optim., 6 (1996), pp. 807–822.

[Dav76] W. C. Davidon, *New least squares algorithms*, J. Optim. Theory Appl., 18 (1976), pp. 187–197.

[Gai94] A. A. Gaivoronski, *Convergence analysis of parallel backpropagation algorithm for neural networks*, Optimization Methods and Software, 4 (1994), pp. 117–134.

[Gri94] L. Grippo, *A class of unconstrained minimization methods for neural network training*, Optimization Methods and Software, 4 (1994), pp. 135–150.

[KuC78] H. J. Kushner and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer-Verlag, New York, 1978.

[Lju77] L. Ljung, *Analysis of recursive stochastic algorithms*, IEEE Trans. Automat. Control,

22 (1977), pp. 551–575.

[LuT94] Z. Q. Luo and P. Tseng, *Analysis of an approximate gradient projection method with applications to the backpropagation algorithm*, Optimization Methods and Software, 4 (1994), pp. 85–101.

[Luo91] Z. Q. Luo, *On the convergence of the LMS algorithm with adaptive learning rate for linear feedforward networks*, Neural Computation, 3 (1991), pp. 226–245.

[MaS94] O. L. Mangasarian and M. V. Solodov, *Serial and parallel backpropagation convergence via nonmonotone perturbed minimization*, Optimization Methods and Software, 4 (1994), pp. 103–116.

[Man93] O. L. Mangasarian, *Mathematical programming in neural networks*, ORSA J. Comput., 5 (1993), pp. 349–360.

[PoT73] B. T. Poljak and Y. Z. Tsypkin, *Pseudogradient adaptation and training algorithms*, Automat. Remote Control, 12 (1973), pp. 83–94.

[Pol87] B. T. Poljak, *Introduction to Optimization*, Optimization Software Inc., New York, 1987.

[TBA86] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, *Distributed asynchronous deterministic and stochastic gradient optimization algorithms*, IEEE Trans. Automat. Control, AC-31, (1986), pp. 803–812.

[WaT90] K. Watanabe and S. G. Tzafestas, *Learning algorithms for neural networks with the Kalman filters*, J. Intelligent and Robotic Systems, 3 (1990), pp. 305–319.

[Whi89] H. White, *Some asymptotic results for learning in single hidden-layer feedforward network models*, J. Amer. Statist. Assoc., 84 (1989), pp. 1003–1013.

[WiH60] B. Widrow and M. E. Hoff, *Adaptive Switching Circuits*, Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, part 4, 1960, pp. 96–104.

[WiS85] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*, Prentice–Hall, Englewood Cliffs, NJ, 1985.