Successive Elememt Correction Algorithms for Sparse Unconstrained Optimization

Guangye Li

CRPC-TR91181
September, 1991

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

SUCCESSIVE ELEMENT CORRECTION ALGORITHMS FOR SPARSE UNCONSTRAINED OPTIMIZATION

GUANGYE LI *

Abstract. This paper presents a successive element correction algorithm and a secant modification of this algorithm. The new algorithms are designed to use the gradient evaluations as efficiently as possible in forming the approximate Hessian. The estimates of the q-convergence and r-convergence rates show that the new algorithms may have good local convergence properties. Some restricted numerical results and comparisons with some previously established algorithms suggest the new algorithms have some promise to be efficient in practice.

Key Words. unconstrained optimization, symmetry, sparsity, Hessian.

AMS(MOS) subject classification. 65K10, 65H10

1. Introduction.

Consider the unconstrained minimization problem

$$\min_{x \in R^n} f(x) \tag{1.1}$$

where $f : D \subset R^n \rightarrow R$ is twice differentiable, and the Hessian $H(x)$ is sparse. To solve problem (1.1), we consider the following iteration is considered:

$$x^{k+1} = x^k - (B^k)^{-1} \nabla f(x^k),\ k = 0, 1, ..., \tag{1.2}$$

where B^k, an approximation to $H(x^k)$, is a symmetric matrix with the same sparsity as the Hessian. To specify the sparsity of a given matrix B, we use M to denote the set of index pairs (i, j), where b_{ij} is a structural nonzero element of B, i.e.,

$$M = \{(i, j) : b_{ij} \neq 0\}.$$

Since B is symmetric, if $(i, j) \in M$, then $(j, i) \in M$. For convenience, we rewrite (1.2) as

$$\bar{x} = x - B^{-1} \nabla f(x), \tag{1.3}$$

where x is the current step, $\bar{x}$ is the new step, and B is an approximation to $H(x)$.

Obtaining a good cheap approximation to the Hessian is an important topic in many recent papers, and it is algo the purpose of this paper. Currently, there are several ways to get a sparse and symmetric approximation to the Hessian under the assumption that a subroutine for the evaluation of $\nabla f(x)$ is available (We assume that it is not convenient to access the component functions of $\nabla f(x)$ separately). In particular, finite-difference methods are often quite attractive since such methods retain good convergence properties, and the number of gradient evaluations needed to difference the

* Computer and Information Technology Institute, Rice University, P. O. Box 1892, Houston, Texas 77251-1892.

gradient is usually small relative to the dimension of the problem when the Hessian is sparse.

For solving sparse nonlinear systems of equations, Curtis, Powell and Reid [5] proposed an efficient finite-difference algorithm, called the CPR algorithm, which is based on a partition of the columns of the Jacobian. Subsequently, Coleman and Moré [1] improved on this method, using graph coloring technique, and then developed a software package for sparse finite differences [3]. Powell and Toint [11] extended the CPR idea to the symmetric case and proposed two practical methods to obtain an approximation to the Hessian: the direct and the indirect lower triangular method. The former is based on a symmetric partition of the columns of the Hessian. The latter is based on a partition of the columns of the lower triangular part of the Hessian. Coleman and Moré [2] connected these partition problems to various graph coloring problems and provided some partitioning algorithms which usually make the number of the gradient evaluations optimal or nearly optimal. A software package for sparse symmetric finite differences was developed by Coleman and Moré [4].

The definitions of a partition and a consistent partition of the columns of a matrix can be found in Li [8]. Now we give the definition of a symmetrically consistent partition.

DEFINITION 1.1. *A partition of the columns of a symmetric matrix B is symmetrically consistent if, for $(i,j) \in M$,*

(1.4)*The group containing column j has no other column with a nonzero in row i,*

or

(1.5)*The group containing column i has no other column with a nonzero in row j.*

Note that given a symmetrically consistent partition of the Hessian, if (1.4) is not satisfied for $(i,j) \in M$, then (1.5) must be satisfied for $(j,i) \in M$. Now the direct method can be formulated as follows: Given a symmetrically consistent partition of the columns of the Hessian, which divides the set $\{1,2,...,p\}$ into p subsets $c_1, c_2, ..., c_p$ (for convenience, c_l, $l = 1,2,...,p$ indicates both the sets of the columns and the sets of the indices of these columns), let

$$d^l = \sum_{j\in c_l} h^l e_j,$$

where $h^l \neq 0$ are scalars, and let

$$y^l = g(x + d^l) - g(x), l = 1,2,...,p,$$

where $g(x) = \nabla f(x)$. Let $(i,j) \in M$ and $i \in c_l$, $j \in c_m$. If only (1.4) is satisfied, then $b_{ij} = b_{ji}$ are determined uniquely by the equation

$$Bd^m = y^m,$$

i.e.,

$$b_{ij} = \frac{e_i^T y^m}{h^m}. \tag{1.6}$$

If only (1.5) is satisfied, then $b_{ij} = b_{ji}$ are determined uniquely by the equation

$$Bd^l = y^l,$$

i.e.,

$$b_{ij} = \frac{e_i^T y^l}{h^l}. \tag{1.7}$$

If both (1.4) and (1.5) are satisfied, then one can choose either (1.6) or (1.7), and the other one will be ignored. Alternatively, an averaging technique is possible.

In some cases, a good, symmetrically consistent partition of the columns of Hessian can take advantage of symmetry to reduce p. As an example, we consider a 6×6 matrix with the arrow structure:

$$A = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & 0 & 0 & 0 & 0 \\ \times & 0 & \times & 0 & 0 & 0 \\ \times & 0 & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & 0 \\ \times & 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

Since one row of the matrix is dense, six directions, $d^i = h^i e_i$, $i = 1, 2, ..., 6$, are needed, i.e., $p = 6$ if the symmetry is ignored. In this case, (1.4) and (1.5) are both satisfied. But if we have the partition: $c_1 = \{1\}$, $c_2 = \{2, 3, 4, 5, 6\}$, and the directions $d^1 = (h, 0, 0, 0, 0, 0)^T$ and $h^2 = (0, h, h, h, h, h)^T$, then B can be determined uniquely by (1.6) and (1.7). In this case, (1.4) is satisfied for c_1, and (1.5) is satisfied for c_2. Unfortunately, in many cases, Powell and Toint's direct symmetric method can not take the advantage of the symmetry to reduce p. This has been shown by Coleman and Moré[2] by means of a result on band matrices saying that $2\beta + 1$ differences are required for estimating the Hessian by a direct symmetric method, where β is the lower bandwidth of the Hessian. Since $2\beta + 1$ differences are also required by a direct method, it is clear that symmetry is of no use in this case. As an example, we consider the Hessian with a tridiagonal structure:

$$A = \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}. \tag{1.8}$$

A symmetrically consistent partition of the columns of the matrix is: $c_1 = \{1, 4, 7, ...\}$, $c_2 = \{2, 5, 8, ...\}$, $c_3 = \{3, 6, 9, ...\}$, $p = 3$, which is the same as the case when the symmetry is ignored, but which is optimal. In this case, both (1.4) and (1.5) are satisfied.

The indirect lower triangular method can sometimes exploit symmetry to a greater extent than the direct method. For example, a symmetric band matrix requires only $\beta+1$ differences using a substitution procedure. On the other hand, the computation of an element of B requires a sequence of substitutions, which makes the cost of obtaining B higher than in the direct method, and which may magnify rounding and truncation errors.

The sparse finite-difference methods referred to above can all be used within the context of a Newton-like method. At each iteration, a new finite-difference approximation to the Hessian is computed at the cost of p extra gradient evaluations, for some p usually much less than n. Still, p is usually considerably large than one and it may be computationally advantageous to further reduce the number of gradient evaluations per iteration. A possible strategy for doing this is at cost of not updating all elements was suggested by Feng and Li[7]. They proposed a successive element correction method, called the column-row update method. By this method, at each iteration one column of B is corrected by a vector obtained from differencing the gradient $\nabla f(x)$, and the corresponding row of B is corrected by the transpose of this column. Such a correction continues successively and periodically.

In this paper, we propose a successive element correction method which is an extension of both the column-row update method and the sparse symmetric direct method. Since it is based on a symmetrically consistent partition of the columns of the Hessian obtained from Coleman and Moré's partitioning algorithms, we call it the CM-element correction method (CMEC). This method needs only two gradient evaluations at each iteration. The differencing direction is chosen to allow for the direct determination of the nonzero elements induced by a group from a symmetrically consistent partition. In order to use the information sufficiently enough to obtain a better approximation to the Hessian, we also propose a secant modification of the CMEC method, which is a combination of the CMEC and a secant method. Our numerical results indicate that the new methods promise to be efficient in practice.

In section 2, we describe the CMEC algorithm and some of its properties. In section 3, we give some local convergence results for the CMEC algorithm. In section 4, we describe the modified algorithm. In section 5, we experimentally compare the algorithms mentioned above.

2. The CMEC method and its properties. Given a symmetrically consistent partition of the columns of the Hessian, which divides the set $\{1,2,...,n\}$ into p subsets $c_1, c_2, ..., c_p$, and given a scalar sequence $\{h^k\}$, where $h^k \neq 0,\ k=1,2,...$, let

$$d^k = \sum_{j\in c_{i_k}} h^k e_j, \tag{2.1}$$

where

$$i_k = \begin{cases} k(mod\, \text{p}), & \text{if k(mod } p) \neq 0 \\ \text{p}, & \text{otherwise,} \end{cases}$$

and let

$$y^k = g(x^k + d^k) - g(x^k). \tag{2.2}$$

Also let $B^k = [b_{ij}^k]$. Suppose that B^0 is a nonsingular matrix which has the same sparsity as the Hessian. Then $\{B^k\}$ can be obtained by the following procedure: When $k \le p$, for $(i,j) \in M$ and $j \in c_k$, if (1.4) is satisfied, the $b_{ij}^k = b_{ji}^k$ is determined by equation

$$e_i^T B^k d^k = e_i^T y^k. \tag{2.3}$$

As in (1.6), we have

$$b_{ij}^k = b_{ji}^k = \frac{e_i^T y^k}{h^k}, \tag{2.4}$$

and the other elements of B^k are equal to the corresponding elements of B^{k-1}. When $k > p$, the elements of B^k are corrected, as described above successively and periodically. In other words, for $(i,j) \in M$ and $j \in c_k$, if (1.4) is satisfied, then $b_{ij}^k = b_{ji}^k$ is determined by (2.4). The other elements of B^k are equal to the corresponding elements of B^{k-1}.

For (1.8), the element corrections at the first three iterations are shown below, where the elements at the '$\star$' positions are corrected.

At the first iteration,

$$B^1 = \begin{bmatrix} \star & \star & 0 & 0 & 0 & 0 \\ \star & \times & \times & 0 & 0 & 0 \\ 0 & \times & \times & \star & 0 & 0 \\ 0 & 0 & \star & \star & \star & 0 \\ 0 & 0 & 0 & \star & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

At the second iteration,

$$B^2 = \begin{bmatrix} \times & \star & 0 & 0 & 0 & 0 \\ \star & \star & \star & 0 & 0 & 0 \\ 0 & \star & \times & \times & 0 & 0 \\ 0 & 0 & \times & \times & \star & 0 \\ 0 & 0 & 0 & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \times \end{bmatrix}.$$

At the third iteration,

$$B^3 = \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \star & 0 & 0 & 0 \\ 0 & \star & \star & \star & 0 & 0 \\ 0 & 0 & \star & \times & \times & 0 \\ 0 & 0 & 0 & \times & \times & \star \\ 0 & 0 & 0 & 0 & \star & \star \end{bmatrix}.$$

At each iteration, at least half of the elements of B^k are corrected by some difference quotients, but only two gradient values are needed. In three iterative steps every nonzero element of B^k is corrected at least one time.

Next, we describe an enhancement of the basic algorithm which can make it more efficient. Note that we can correct more elements than those in just one group at each iteration. In general, we can expand a group of columns by adding some columns from other groups. The elements in this expanded group can be determined uniquely by only two gradient evaluations. The rule for adding columns from other groups is as follows: Columns from other groups can be added provided there are no nonzeros in the same row position in this expanded group. We may expand a group c_i to a group $\bar{c}_i$ with "maximal size"; that is if we add an additional column to $\bar{c}_i$, then there must be at least two nonzeros at the same row position.

As an example, we consider the problem with the following structure: a dense five-dimensional leading submatrix is followed by a tridiagonal matrix of arbitrary length. An eight-dimensional version of this structure is given in (2.5).

$$B = \begin{bmatrix} \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}. \tag{2.5}$$

The following symmetrically consistent partition of the columns of the matrix is optimal: $c_1 = \{1\}$, $c_2 = \{2\}$, $c_3 = \{3, 6, 9, ...\}$, $c_4 = \{4, 7, 10, ...\}$ and $c_5 = \{5, 8, 11, ...\}$. Hence, the number of gradient evaluations for the direct algorithm at each iteration is six. Note that only five elements of the Hessian are estimated for each of the gradient values $g(x + he_1)$ and $g(x + he_2)$. When n is large, this is an inefficient use of gradient evaluations. However, we can expand each group so that close to n elements of the Hessian are estimated with each gradient evaluation. Specifically, we can expand c_1 to $\bar{c}_1 = \{1, 7, 10, 13, ...\}$ and c_2 to $\bar{c}_2 = \{2, 8, 11, 14, ...\}$; and let $\bar{c}_3 = c_3$, $\bar{c}_4 = c_4$ and $\bar{c}_5 = c_5$. Note that there are some overlaps between $\bar{c}_i$, $i = 1, ..., 5$, i.e., $\bar{c}_1 \cap \bar{c}_4 = \{7, 10, 13, ...\}$ and $\bar{c}_2 \cap \bar{c}_5 = \{8, 11, 14, ...\}$. The elements corrected in the first three iterations are shown below. Again, the elements at the "$\star$" positions are corrected. At the first iteration:

$$B = \begin{bmatrix} \star & \star & \star & \star & \star & 0 & 0 & 0 \\ \star & \times & \times & \times & \times & 0 & 0 & 0 \\ \star & \times & \times & \times & \times & 0 & 0 & 0 \\ \star & \times & \times & \times & \times & 0 & 0 & 0 \\ \star & \times & \times & \times & \times & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & \times & \star & 0 \\ 0 & 0 & 0 & 0 & 0 & \star & \star & \star \\ 0 & 0 & 0 & 0 & 0 & 0 & \star & \times \end{bmatrix}.$$

At the second iteration:

$$B = \begin{bmatrix} \times & \star & \times & \times & \times & 0 & 0 & 0 \\ \star & \star & \star & \star & \star & 0 & 0 & 0 \\ \times & \star & \times & \times & \times & 0 & 0 & 0 \\ \times & \star & \times & \times & \times & 0 & 0 & 0 \\ \times & \star & \times & \times & \times & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \star \\ 0 & 0 & 0 & 0 & 0 & 0 & \star & \star \end{bmatrix}.$$

Note that corresponding to the gradient values $g(x + he_1)$ and $g(x + he_2)$, n and $n - 1$ elements of the lower triangular part of the Hessian are estimated at the first and the second iteration, respectively.

ALGORITHM 2.1. *Given a symmetrically consistent partition of the columns of the Hessian which divides the set $\{1, 2, ..., n\}$ into p subsets $c_1, c_2, ..., c_p$, expand each group c_i to a maximal size group $\bar{c}_i$. For a given $x^0 \in R^n$, and a nonsingular matrix B^0, which is symmetric and has the same sparsity as the Hessian, do the following: At the initial step:*

1. *Set $l = 0$.*
2. *Solve $B^0 s^0 = -g(x^0)$.*
3. *Choose x^1 by $x^1 = x^0 + s^0$ or by a global strategy.*

At each iteration $k > 0$:

1. *Choose a scalar h^k.*
2. *If $l < p$, then set $l = l + 1$, otherwise set $l = 1$.*
3. *Set*

$$d^k = \sum_{j \in \bar{c}_l} h^k e_j.$$

4. *If $j \in \bar{c}_l$ and $(i, j) \in M$, then set*

$$b_{ij}^k = \frac{1}{h^k} e_i^T (g(x^k + d^k) - g(x^k)), \tag{2.6}$$

and set

$$b_{ji}^k = b_{ij}^k,$$

otherwise set

$$b_{ij}^k = b_{ij}^{k-1},$$

5. *Solve $B^k s^k = -g(x^k)$.*
6. *Choose x^{k+1} by $x^{k+1} = x^k + s^k$ or by a global strategy.*
7. *Check convergence.*

Now, we discuss some properties of the CMEC algorithm.

LEMMA 2.1. *If $\{B^k\}$ is obtained by Algorithm 2.1, then all elements in nonzero positions of B^k are corrected at least one time in every p steps of the iteration.*

Proof. Consider the first p steps. Let $(i,j) \in M$. and $j \in c_m, i \in c_l, 1 \leq m \leq p, 1 \leq l \leq p$. If (1.4) is satisfied, then b_{ij}^{m-1} is corrected by $b_{ij}^m = (e_i^T y^m)/h^m$. Otherwise, (1.4) is satisfied for $(j,i) \in M$. Then b_{ij}^{l-1} is corrected by $b_{ij}^l = (e_j^T y^l)/h^l$. Therefore, Lemma 2.1 holds for $k \leq p$. Thus, the lemma holds in the first p steps, and by periodicity, the result follows.

Let

$$J^k = \int_0^1 H(x^k + td^k)dt. \tag{2.7}$$

and $J^k = [J_{ij}^k]$. Note that J^k is symmetric and it has the same sparsity as the Hessian.

LEMMA 2.2. *If b_{ij}^{k-1} is corrected at the kth iteration, then*

$$b_{ij}^k = b_{ji}^k = J_{ij}^k. \tag{2.8}$$

Proof. Note that

$$y^k = J^k d^k. \tag{2.9}$$

If (1.4) is satisfied for $(i,j) \in M$, then

$$e_i^T y^k = e_i^T J^k d^k = h^k e_i^T J^k e_j = h^k J_{ij}^k. \tag{2.10}$$

Thus, (2.8) follows from (2.6) and (2.10). Otherwise, (1.4) is satisfied for $(j,i) \in M$, and then (2.9) implies that

$$e_j^T y^k = e_j^T J^k d^k = h^k J_{ji}^k = h^k J_{ij}^k. \tag{2.11}$$

Thus, (2.8) follows from (2.6) and (2.11).

In the following context, we assume that D is an open convex set, and sometimes we assme that the Hessian astisfies the following Lipschitz condition: For $(i,j) \in M$, there exists $\alpha_{ij} > 0$ such that

$$|e_i^T(H(x) - H(y))e_j| \leq \alpha_{ij}\|x - y\|, \; \forall x, y \in D. \tag{2.12}$$

Let $\alpha = (\sum_{(i,j)\in M} \alpha_{ij}^2)^{1/2}$. Then (2.12) implies that

$$\|H(x) - H(y)\|_F \leq \alpha\|x - y\|, \; \forall x, y \in D. \tag{2.13}$$

LEMMA 2.3. *Let $\{x^k\}$ and $\{B^k\}$ be generated by Algorithm 2.1. Assume that $x^k \in D$ and $x^k + d^k \in D$. If b_{ij}^{k-1} is corrected at the kth iterative step, then*

$$|e_i^T(B^k - H(x^k))e_j| \leq \frac{\sqrt{n}}{2}\alpha_{ij}|h^k|. \tag{2.14}$$

Proof. It follows from (2.8) and (2.12) that

$$\begin{aligned}|e_i^T(B^k - H(x^k))e_j| &= |e_i^T(J^k - H(x^k))e_j| \\ &= |e_i^T(\int_0^1 (H(x^k + td^k) - H(x^k))dt)e_j| \\ &\leq \alpha_{ij}\|d^k\| \int_0^1 tdt = \frac{\alpha_{ij}}{2}\|d^k\| \leq \frac{\sqrt{n}}{2}\alpha_{ij}|h^k|.\end{aligned}$$

THEOREM 2.4. *Assume $H(x)$ satisfies Lipschitz condition (2.12). Let $\{x^j\}_{j=1}^k \subset D$ and $\{B^j\}_{j=0}^k$ be generated by Algorithm 2.1 with B^0 satisfying $\|B^0 - H(x^0)\|_F < \delta$. If $\{x^j + d^j\}_{j=1}^k \subset D$, then for $k < p$,*

$$\|B^k - H(x^k)\|_F \leq \alpha(2\bar{e}_k + \bar{h}_k) + \delta, \tag{2.15}$$

and for $k \geq p$,

$$\|B^k - H(x^k)\|_F \leq \alpha(\bar{e}_k + \bar{h}_k), \tag{2.16}$$

where

$$\bar{e}_k = \max_{1\leq j\leq m(k)}\{\|x^k - x^{k-j}\|\}, \ \bar{h}_k = \frac{\sqrt{n}}{2}\max_{0\leq j\leq m(k)}\{h^{k-j}\},$$

where $m(k) = \min\{k, p-1\}$, and $h^0 = 0$.

Proof. We first consider the case where $k \geq p$. By Lemma 2.1 for any $(i,j) \in M$, there exists at least one integer $0 \leq q \leq p-1$ such that b_{ij}^{k-q} is corrected at the $(k-q)$th step. Let m be the smallest one among all such q's. Then

$$e_i^T B^k e_j = e_i^T B^{k-m} e_j.$$

It can be obtained from Lemma 2.3 that

$$\begin{aligned}|e_i^T(B^k - H(x^k))e_j| &= |e_i^T(B^{k-m} - H(x^k))e_j| \\ &= |e_i^T(B^{k-m} - H(x^{k-m}))e_j| + |e_i^T(H(x^{k-m}) - H(x^k))e_j| \\ &\leq \alpha_{ij}(\frac{\sqrt{n}}{2}|h^{k-m}| + \|x^k - x^{k-m}\|) \\ &\leq \alpha_{ij}(\bar{e}_k + \bar{h}_k). \qquad (2.17)\end{aligned}$$

Therefore,

$$\|B^k - H(x^k)\|_F^2 = \sum_{(i,j)\in M} |e_i^T(B^k - H(x^k))e_j|^2 \leq \alpha^2(\bar{e}_k + \bar{h}_k)^2.$$

Thus, (2.16) follows from (2.17).

Now we consider the case where $1 \leq k < p$. In this case, it may happen that for some $(i,j) \in M$,

$$e_i^T B^k e_j = e_i^T B^0 e_j.$$

Let M_2^k be the set of all such (i,j) pairs, and let $M_1^k = \{(i,j) \in M : (i,j) \notin M_2^k\}$. Also let

$$E^k = B^k - H(x^k),\ E_1^k = \sum_{(i,j)\in M_1^k} e_i^T E^k e_j e_i e_j^T,\ E_2^k = \sum_{(i,j)\in M_2^k} e_i^T E^k e_j e_i e_j^T.$$

Then, using (2.17), we have that

$$\|E_1^k\|_F \leq \alpha(\bar{e}_k + \bar{h}_k),$$

and therefore,

$$\begin{aligned} \|E^k\|_F &= \|E_1^k + E_2^k\|_F \leq \|E_1^k\|_F + \|E_2^k\|_F \\ &\leq \alpha(\bar{e}_k + \bar{h}_k) + \|H(x^k) - B^0\|_F \\ &\leq \alpha(\bar{e}_k + \bar{h}_k) + \alpha\|x^k - x^0\| + \|H(x^0) - B^0\|_F \\ &\leq \alpha(2\bar{e}_k + \bar{h}_k) + \delta. \end{aligned}$$

3. Local convergence results for the CMEC algorithm. To study the local convergence of our algorithm, we assume that $g : D \subset R^n \to R^n$ has the following property:

(3.1) *There is an $x^* \in D$, such that $g(x^*) = 0$ and $H(x^*)$ is nonsingular.*

THEOREM 3.1. *Assume that $g : D \subset R^n \to R^n$ satisfies (3.1) and H satisfies Lipschitz condition (2.12). Assume that $\{x^k\}$ is generated by Algorithm 2.1 without any global strategy. Then there exist ϵ, δ, $h > 0$ such that if $0 < |h^k| < h$ and $x^0 \in D$ and $B^0 \in D$ satisfy*

$$\|x^0 - x^*\| \leq \epsilon,\ \ \|B^0 - H(x^0)\|_F \leq \delta,$$

then $\{x^k\}$ is well defined and converges q-linearly to x^*. If $\lim_{k\to\infty} |h^k| = 0$, then the convergence is q-superlinear. If there exists some constant C such that $|h^k| \leq C\|g(x^k)\|$, then the convergence is p-step q-quadratic.

Proof. Since $x^* \in D$ and D is an open convex set, we can choose ϵ so that $S(x^*, 2\epsilon) \equiv \{x : \|x - x^*\| < 2\epsilon\} \subset D$. Also, We can chose ϵ, δ and h so that

$$\sqrt{n}h < \epsilon,\ \ 2\beta(\alpha(\frac{9\epsilon}{2} + h) + \delta) < \frac{1}{2}.$$

where $\beta > 0$ satisfies $\|H^{-1}(x^*)\|_F < \beta$.

We first show, by induction on k, that

$$\|x^{k+1} - x^*\| \leq \frac{1}{2}\|x^k - x^*\|. \tag{3.2}$$

For $k = 0$, we first show that B^0 is nonsingular. Notice that

$$\|H^{-1}(x^*)(B^0 - H(x^*))\|_F \ \leq\ \|H^{-1}(x^*)\|_F[\|B^0 - H(x^0)\|_F + \|H(x^0) - H(x^*)\|_F]$$

$$\leq \ \beta(\delta + \alpha\epsilon) < \frac{1}{2}.$$

Thus, by Dennis and Schnabel's Theorem 3.1.4 [6],

$$\|(B^0)^{-1}\|_F \leq \frac{\beta}{1-\frac{1}{2}} = 2\beta.$$

Therefore x^1 is well defined, and

$$\begin{aligned}
\|x^1 - x^*\| &\leq \|(B^0)^{-1}\|_F[\|g(x^*) - g(x^0) - H(x^0)(x^* - x^0)\| \\
&\qquad +\|B^0 - H(x^0)\|_F\|x^* - x^0\|] \\
&\leq 2\beta[\frac{\alpha}{2}\|x^* - x^0\| + \delta]\|x^* - x^0\| \\
&\leq 2\beta(\frac{\alpha\epsilon}{2} + \delta)\|x^* - x^0\| \leq \frac{1}{2}\|x^* - x^0\|.
\end{aligned}$$

This means that (3.2) holds for $k = 0$. Now suppose (3.2) holds for $k = 1, 2, ..., m-1$. We show that it also holds for $k = m$. By (3.2),

$$\|x^m + d^m - x^*\| \leq \|x^m - x^*\| + \|d^m\| \leq \|x^0 - x^*\| + \sqrt{n}h < 2\epsilon.$$

Thus, $\{x^k + d^k\}_{k=1}^m \subset S(x^*, 2\epsilon) \subset D$. By Theorem 2.4, there exists an integer $1 \leq j_0 \leq \min\{m, p-1\}$ such that

$$\begin{aligned}
\|B^m - H(x^m)\|_F &\leq \alpha(2\|x^m - x^{m-j_0}\| + \bar{h}_m) + \delta \\
&\leq \alpha(2\|x^m - x^*\| + \|x^* - x^{m-j_0}\| + \bar{h}_m) + \delta \\
&\leq \alpha(4\|x^* - x^{m-j_0}\| + \bar{h}_m) + \delta.
\end{aligned} \tag{3.3}$$

Thus,

$$\|H^{-1}(x^*)(B^m - H(x^*))\|_F \leq \beta(\alpha(5\epsilon + h) + \delta) < \frac{1}{2},$$

and therefore,

$$\|(B^m)^{-1}\|_F \leq 2\beta,$$

which shows that x^{m+1} is well defined. By (3.3)

$$\begin{aligned}
\|x^{m+1} - x^*\| &\leq \|(B^m)^{-1}\|_F[\|g(x^*) - g(x^m) - H(x^m)(x^* - x^m)\| \\
&\qquad +\|B^m - H(x^m)\|_F\|x^* - x^m\|] \\
&\leq 2\beta[\frac{\alpha}{2}\|x^m - x^*\| + \|B^m - H(x^m)\|_F]\|x^* - x^m\| \\
&\leq 2\beta[\frac{\alpha\epsilon}{2} + \alpha(4\epsilon + \bar{h}_m) + \delta]\|x^* - x^m\| \\
&= 2\beta[\alpha(\frac{9}{2}\epsilon + h + \delta]\|x^* - x^m\| \leq \frac{1}{2}\|x^m - x^*\|,
\end{aligned} \tag{3.4}$$

which completes the induction step. It follows from (3.2) that $\{x^k\}$ converges to x^* at least q-linearly.

Note that for $k \geq p$, by (2.16) inequality (3.3) is changed to

$$\begin{aligned} \|B^k - H(x^k)\|_F &\leq \alpha(\|x^k - x^*\| + \|x^* - x^{k-p+1}\| + \bar{h}_k) \\ &\leq \alpha(2\|x^* - x^{k-p+1}\| + \bar{h}_k), \end{aligned}$$

and therefore, (3.4) is changed to

$$\|x^{k+1} - x^*\| \leq 2\alpha\beta(\frac{5}{2}\|x^* - x^{k-p+1}\| + \bar{h}_k)\|x^k - x^*\|. \tag{3.5}$$

Since $\{\bar{h}_k\}$ is a sub-sequence of $\{h^k\}$, $h^k \to 0$ implies $\bar{h}_k \to 0$. Therefore, by (3.5), $\{x^k\}$ converges to x^* q-superlinearly if $h^k \to 0$. By Dennis and Schnabel's Theorem 5.4.1 [6],

$$|h^k| \leq C\|g(x^k)\|$$

is equivalent to

$$|h^k| \leq C_1\|x^k - x^*\|,$$

where $C_1 > 0$ is a constant. Therefore, if $|h^k| \leq C\|g(x^k)\|$, inequality (3.5) can be rewritten as

$$\|x^{k+1} - x^*\| \leq C_2\|x^* - x^{k-p+1}\|\|x^k - x^*\| \leq C_2\|x^* - x^{k-p+1}\|^2, \tag{3.6}$$

where $C_2 > 0$ is a constant, which implies that $\{x^k\}$ converges to x^* at least p-step q-quadratically.

THEOREM 3.2. *Assume that g satisfies the hypotheses in Theorem 3.1 and that $|h^k| \leq C\|g(x^k)\|$. Then the r-convergence order of Algorithm 2.1 is not less than τ_p, where τ_p is the unique positive root of the equation*

$$t^p - t^{p-1} - 1 = 0.$$

Proof. Inequality (3.6) can be rewritten as

$$\|x^{k+1} - x^*\| \leq C_3\|x^k - x^*\| \sum_{j=0}^{p-1} \|x^{k-j} - x^*\|,$$

where $C_3 > 0$ is a constant. Thus, the desired result follows from Ortega and Rheinboldt's Theorem 9.2.9 [10].

It is well known that the efficiency index is one of the ways to evaluate an algorithm. It is defined by $E = ln\tau/\mu$, where τ is the convergence order of the algorithm and μ is the number of function evaluations at each iteration counted in number of vectors ($g(x)$). To compare Algorithm 2.1 with Powell and Toint's direct finite-difference algorithm which needs $p+1$ function evaluations per iteration, and the local convergence order of which is 2 under some assumptions, we computed the ratio of the efficiency indices of these two algorithms, i. e.,

$$r_p = \frac{(p+1)ln\tau_p}{2ln2}.$$

The values of r_p various with different p are shown in Table 1. The results show that the ratio is always not less than one and it increases as p increases, however, the increase speed slows down as p becomes larger. This means that Algorithm 2.1 may be more competitive in the sense of the efficiency index when p is relatively large.

TABLE 1

p	1	3	5	7	9	19	29	39	100
r_p	1.00	1.10	1.21	1.31	1.39	1.68	1.86	2.00	2.48

4. A secant modification of the CMEC method.

As mentioned by the author in a previous paper [8], at step k, the information $g(x^{k-1})$ has not been used in the CMEC method. Therefore, it is reasonable to consider to use this information and make a secant update to B^k to get a better approximation to the Hessian, say $\bar{B}^k$, and then solve the linear system

$$\bar{B}^k s^k = -g(x^k).$$

To implement this, we apply Toint's [12] sparse symmetric secant (SPSB) update to have the following modified CMEC method.

ALGORITHM 4.1. *Given a symmetrically consistent partition of the Hessian, x^0 and B^0 as in Algorithm 2.1, do the following: At the initial step :*

```
    1. Set $l = 0$ and $\bar{B}^0 = B^0$.
    2. Solve $\bar{B}^0 s = -g(x^0)$.
    3. Choose $x^1$ by $x^0 + s$ or by a global strategy.
At each iteration $k > 0$:
    1. Update $B^{k-1}$ by Algorithm 2.1 to get $B^k$.
    2. Update $B^k$ by SPSB update to get $\bar{B}^k$.
    3. Solve $\bar{B}^k s = -g(x^k)$.
    4. Choose $x^{k+1}$ by $x^{k+1} = x^k + s$ or by a global strategy.
    5. Check convergence.
```

Using the same analysis as that for the modified CM-successive column correction method (see Li [8]), we have the following local convergence result for the modified CMEC method.

THEOREM 4.1. *The modified CMEC method has at least the same local convergence properties as the CMEC method.*

5. The numerical results.

We computed six examples by Powell and Toint's direct method (PTD), Powell and Toint's indirect method (PTID), Toint's sparse PSB method (SPSB), the CM-element correction methods (with and without expanding the groups of a partition) and the modified CM-element correction method. In this section we compare the numerical results from these six methods.

The 'global strategy' used to force convergence from far away points was a simple line search backtracking strategy as described by Dennis and Schnabel[6]. If a direction p is found to be an increase direction, i.e. $p^T \nabla f > 0$, then the negative direction $-p$ will be used. According to Dennis and Schnabel[6], we choose the step length in finite differences for each element as

$$h_j^k = \sqrt{macheps}\, x_j^k,$$

where *macheps* is the machine precision. The stopping tests we used are the ones given by Dennis and Schnabel [6] and all tests were run with the same accuracy requirement ($\epsilon = 10^{-5}$). For all functions and all methods, the global minimum was found. For the SPSB method, Algorithm 2.1 and Algorithm 4.1, the initial approximations to the Hessian were computed by the PTID method for Example 5.1-5.4 and by the PTD method for Example 5.5-5.6. All tests were run on the Jilin University Honeywell DPS-8 in double precision.

Example 5.1 is an extension of Example 9.2.2 in [6], where the dimension is only two. Example 5.2, Example 5.3 and Example 5.4 are variations of the Broyden Banded Function (see [9]). Here we only made some changes on the lower half bandwidth and the upper half bandwidth to have five diagonal, seven diagonal and nine diagonal structures. The results for these four examples are shown in Table 2 where NI is the number of iterations and NG is the total number of the gradient evaluations needed for solving these problems.

Example 5.5 and Example 5.6 are created to see the effect of expanding groups in Algorithm 2.1. Example 5.5 has the same structure as (2.5), i. e. a dense 5-dimensional leading sub-matrix followed by a tridiagonal matrix, which we call the tadpole structure. The partition of the columns of the Hessian we used for the PTID method is $c_1 = \{1\}$, $c_2 = \{2\}$, $c_3 = \{3\}$, $c_4 = \{4, 6, 8, ...\}$ and $c_5 = \{5, 7, 9, ...\}$, which is an optimal partition. The symmetrically consistent partition of the columns of the Hessian we used for other methods is $c_1 = \{1\}$, $c_2 = \{2\}$, $c_3 = \{3, 6, 9, ...\}$, $c_4 = \{4, 7, 10, ...\}$ and $c_5 = \{5, 8, 11, ...\}$, which is also optimal. For Algorithm 2.1 and Algorithm 4.1 we expand group c_1 to $\bar{c}_1 = \{1, 7, 10, 13, ...\}$ and c_2 to $\bar{c}_2 = \{2, 8, 11, 14, ...\}$. Example 5.6 has also the tadpole structure. However, the dimension of the leading sub-matrix is six instead of five. The partition of the columns we used for the PTID method for this example is $c_1 = \{1\}$, $c_2 = \{2\}$, $c_3 = \{3\}$, $c_4 = \{4\}$, $c_5 = \{5, 7, 9, ...\}$, $c_6 = \{6, 8, 10, ...\}$, which is an optimal partition. The symmetrically consistent partition of the columns of the Hessian we used for other methods is $c_1 = \{1\}$, $c_2 = \{2\}$, $c_3 = \{3\}$, $c_4 = \{4, 7, 10, ...\}$, $c_5 = \{5, 8, 11, ...\}$, and $c_6 = \{6, 9, 12, ...\}$, which is also optimal. For Algorithm 2.1 and Algorithm 4.1 we expand group c_1 to $\bar{c}_1 = \{1, 8, 11, 14, ...\}$, c_2 to $\bar{c}_2 = \{2, 9, 12, 15, ...\}$ and c_3 to $\bar{c}_3 = \{3, 10, 13, 16, ...\}$. In Table 3, we compare the PTD method, PTID method, the CMEC method without expanding any groups of a partition (CMECW), Algorithm 2.1 and Algorithm 4.1.

Example 5.1 (Three diagonal).

$$\begin{aligned} f_i(x) &= (x_i - 2)^4 + (x_i - 2)^2 x_{i+1}^2 + (x_{i+1} + 1)^2,\ i = 1, 2, ..., n-1, \\ f_n(x) &= (x_n - 2)^4, \\ x^0 &= (-1, -1, ..., -1)^T, \\ n &= 36,\ p = 2\, for\ PTID\ and\, p = 3\ for\ others. \end{aligned}$$

Example 5.2 (Five diagonal).

$$f(x) = \sum_{i=1}^{n} f_i(x),$$

Table 2

Algorithms	Example 5.1		Example 5.2		Example 5.3		Example 5.4	
	NI	NG	NI	NG	NI	NG	NI	NG
PTD	7	29	7	43	7	57	7	71
PTID	7	22	7	29	7	36	7	43
SPSB	29	32	31	35	39	44	34	40
Alg.2.1	11	24	14	31	17	38	19	43
Alg.4.1	10	22	11	25	13	30	14	33

where

$$f_i(x) = x_i(2 + 5x_i^2) + 1 - \sum_{j \in J_i} x_j(1 + x_j),$$

where

$$J_i = \{j : j \neq i, max(1, i - m_l) \leq j \leq min(n, i + m_u)\},$$

and

$$m_l = 1, \;\; m_u = 1, \;\; x^0 = (-1, -1, ..., -1)^T,$$

$n = 36$, $p = 3$ for PTID and $p = 5$ for others.

Example 5.3 (Seven diagonal).

The same as Example 5.2 except for that $m_l = 2$, $m_u = 2$, $n = 36$, $p = 4$ for PTID and $p = 7$ for others.

Example 5.4 (Nine diagonal structure).

The same as Example 5.2 except for that $m_l = 3$, $m_u = 3$, $n = 36$, $p = 5$ for PTID and $p = 9$ for others.

Example 5.5 (Tadpole structure 1).

$$f(x) = \hat{f}(x) + 0.5(x_1 - x_2 + x_3 - x_4 + x_5 - 1)^4,$$

where $\hat{f}(x)$ is defined by Example 5.1,

$$x1 = (-1, -1, ..., -1)^T, \; x2 = (3, 3, ..., 3)^T, \; n = 36, \; p = 5.$$

Example 5.6 (Tadpole structure 2).

$$f(x) = \hat{f}(x) + 0.5(x_1 - x_2 + x_3 - x_4 + x_5 - x_6)^4,$$

where $\hat{f}(x)$ is defined by Example 5.1,

$$x^0 = (-1, -1, ..., -1)^T, \; x2 = (3, 3, ..., 3)^T, \; n = 36, \; p = 6.$$

It can be seen from the numerical results that for most of the cases, Algorithm 4.1 takes the least number of gradient evaluations and it takes less number of iterations than Algorithm 2.1. For all the cases, Algorithm 2.1 takes less number of gradient evaluations than the PTD method.

From Table 5 we can see that when the groups in a symmetrically consistent partition of the columns can be expanded, both Algorithm 2.1 and Algorithm 4.1 may use much fewer gradient values than those for the PTD and PTID methods.

TABLE 3

	Example 5.5				Example 5.6			
Algorithms	$x^0 = x1$		$x^0 = x2$		$x^0 = x1$		$x^0 = x2$	
	NI	NG	NI	NG	NI	NG	NI	NG
PTD	6	37	8	49	6	43	8	57
PTID	6	37	8	49	6	43	8	57
SPSB	36	42	25	31	42	49	19	26
CMECW	13	31	17	39	15	36	19	44
Alg. 2.1	12	29	15	35	14	34	17	40
Alg. 4.1	11	27	13	31	11	28	11	28

6. Concluding remarks. We have given two algorithms for solving unconstrained optimization problems which use gradient evaluations efficiently. The local convergence properties established for Algorithm 2.1 in section 3 are quite satisfactory, and the numerical results suggest the algorithms have some promise to be efficient in practice. When the gradient evaluation is not very expensive, we may consider a variation of Algorithm 2.1 or Algorithm 4.1 such that instead of correcting of just one group, two or more groups are chosen at each iteration. This technique may increase the r-convergence order and reduce the number of the iterations required for convergence, but, of course, it will take more gradient evaluations at each iteration than those required by Algorithm 2.1 or Algorithm 4.1.

Acknowledgement.

The author would like to thank Professor Thomas Coleman for his many important and helpful suggestions and corrections on the preliminary draft of this paper.

REFERENCES

[1] T. COLEMAN AND J. M. É, *Estimation of sparse jacobian matrices and graph coloring problems*, SIAM Journal on Numerical Analysis, 20 (1983), pp. 187–209.

[2] ——, *Estimation of sparse hessian matrices and graph coloring problems*, Mathematical Programming, 28 (1984), pp. 243–270.

[3] ——, *Software for estimation of sparse jacobian matrices*, ACM Transaction on Mathematical software, 10 (1984), pp. 329–345.

[4] ——, *Software for estimation of sparse hessian matrices*, ACM Transaction on Mathematical software, 11 (1985), pp. 363–377.

[5] A. CURTIS, M. POWELL, AND J. REID, *On the estimation of sparse jacobian matrices*, Journal of Applied Mathematics, 13 (1974), pp. 117–119.

[6] J. DENNIS AND R. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[7] G. FENG AND G. LI, *The column-row update method*, Numerical Mathematics, Journal of Chinese Universities, 5 (1983).

[8] G. LI, *Successive column correction algorithms for solving sparse nonlinear systems of equations*, Mathematical Programming, 43 (1989), pp. 187–207.

[9] J. MORÉ, B. GARBOW, AND K. HILLSTROM, *Testing unconstrained optimization software*, ACM Transaction on Mathematical software, 7 (1981), pp. 17–41.

[10] J. ORTEGA AND W. RHEINBOLDT, *Iterative Solution of Nonlinear Equations*, Academic Press, New York and London, 1970.

[11] M. Powell and P. Toint, *On the estimation of sparse hessian matrices*, SIAM Journal on Numerical Analysis, 16 (1979), pp. 1060–1074.

[12] P. Toint, *On sparse and symmetric matrix updating subject to a linear equation*, Mathematics of Computation, 31 (1977), pp. 954–961.