Numer. Math. 39, 119–137 (1982)

Numerische Mathematik
© Springer-Verlag 1982

Partitioned Variable Metric Updates for Large Structured Optimization Problems

A. Griewank*, and Ph.L. Toint

Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, United Kingdom
Department of Mathematics, Facultés Universitaires de Namur, Namur, Belgium

Summary. This paper presents a minimization method based on the idea of partitioned updating of the Hessian matrix in the case where the objective function can be decomposed in a sum of convex "element" functions. This situation occurs in a large class of practical problems including nonlinear finite elements calculations. Some theoretical and algorithmic properties of the update are discussed and encouraging numerical results are presented.

Subject Classifications: AMS(MOS): 65H10 CR: 5.15.

Introduction

There has been, in the past few years, a growing interest in methods for unconstrained minimization that are applicable to problems involving a large number of variables. A number of the newly proposed algorithms are based on the quasi-Newton approach that was much studied in its application to lower dimensional problems (see, for example, [3, 12, 9, 4]). Recently, this approach has been extended to the case where sparsity is present in the problem description, therefore allowing experiments with larger number of independent variables ([21, 14, 18]). These large problems arise typically from discretizations as finite elements [13] and variational calculations [8], and are of increasing practical importance.

Another point of view on large scale unconstrained minimization was taken in [24] and [10], where consideration was given to partially separable functions which have a natural decomposition into partial and conflicting smaller objectives. It is the aim of the present paper to explore and extend this approach further and show that great numerical advantage can be taken of such a structure. Specifically, the possibility of updating approximations to the small Hessian matrices of the elementary objectives is considered and shown to

* Work supported by a research grant of the Deutsche Forschungsgemeinschaft, Bonn, FRG

be successful on some examples arising from discretized optimization calculations. The paper also includes some theoretical and algorithmic considerations related to this subject.

Section 2 presents the problem and the basic assumptions. Section 3 is concerned with the problem of updating semi-definite Hessian approximations while Sect. 4 contains some general algorithmic considerations. Section 5 suggests implementing a procedure that varies the discretization level of the underlying problem. Numerical results are presented in Sect. 6. Finally, Sect. 7 is devoted to concluding remarks and perspectives for future work.

2. Decomposition Into Element Functions and General Assumptions

We consider the problem of minimizing a twice continuously differentiable objective function depending on n real variables. If n is relatively large, greater than 200, say, we can only hope to solve this problem within the usual limitation on storage and computing time if the Hessian matrix of our objective function is sparse or otherwise structured. Gill and Murray [8] found that conjugate gradient and limited memory quasi-Newton methods can work quite well for a non quadratic function especially if the eigenvalues of its Hessian are clustered. However, as they point out in their report, the performance of these methods is rather unpredictable and may be at times unsatisfactory. One should also add that, except for some well known test functions, the distribution and spread of the eigenvalues of the Hessian matrix at the solution is not known.

Therefore we pursue the classical quasi-Newton approach of approximating the objective function by a quadratic model. Let us assume that we want to minimize

$$f: R^n \to R. \tag{1}$$

Then our approximating quadratic at the point x would be

$$q(x+s) = f(x) + s^T g(x) + \tfrac{1}{2} s^T B s \tag{2}$$

where $g(x)$ is the gradient vector of f at the point x and where B is a symmetric matrix that approximates the true Hessian of f, $G(x)$ say, at the same point. We then use the solution to the linear system

$$Bs = -g(x) \tag{3}$$

as a search direction.

Even if the number of variables is very large, this approach is applicable to many practical problems. In this paper, we consider the wide class of partially separable problems where the objective function is a sum

$$f(x) = \sum_{i=1}^{m} f_i(x), \tag{4}$$

of m smaller „element functions“ f_i, each of which is constant in all but $n_i \ll n$ components of x. A detailed definition of partial separability is given in [10].

As we are going to deal throughout the paper with quantities that are related to these element functions, we now introduce some notation. Depending on the context, f_i will be considered as a function of all the components of x or of its n_i internal variables in a given ordering. Correspondingly, we will denote by $g_i(x)$, or simply g_i, the gradient vector of the element function f_i at the point x and by x_i and s_i, the projections of the point x in R^n and of the step s in R^n on the subspace corresponding to the variables that are involved in f_i. Similarly, $G_i(x)$ will denote the true Hessian of f_i at x.

This decomposition arises naturally in many calculations that occur in the discretization of another (possibly infinite dimensional) problem. Nonlinear least squares, finite elements and other discretizations of variational problems are good examples where the decomposition (4) has a very natural interpretation. Moreover, it is often the case that a large number of the element functions f_i are of the same form and differ only by the assignment between the internal variables and the components of x. This is certainly the case in finite element applications where the number of structurally different elements is usually quite low.

For simplicity, we assume that f is already given as a certain sum of the form (4) and that it has an isolated minimizer x^* with $G(x^*)$ being nonsingular. Referring to a convex decomposition, we make the additional assumption that every element function f_i is globally convex and also that the null space of each element Hessian G_i does not depend on x in R^n. These assumptions are typically satisfied in finite element applications where the f_i are constant with respect to certain transformations of their variables, e.g. translations.

We then can ask the user to supply the nonzero entries of $g_i(\,.\,)$ as a vector of size n_i and possibly also the nonzero entries of $G_i(\,.\,)$. Whereas it is assumed that all g_i can be evaluated explicitly, some of the G_i may be too difficult to obtain so that we have to base the quadratic model (2) on approximations B_i to G_i with

$$B=\sum_{i=1}^{m} B_i \tag{5}$$

where the B_i denote, in this context, $n\times n$ matrices with $n_i\times n_i$ nonzero entries. Typically, many pairs of variables do not appear together in any element function so that the resulting Hessian $G(\,.\,)$ and its approximation B will be sparse. However, we do not make this assumption explicitly because the methods we will discuss are applicable in all cases where B can be multiplied cheaply by an arbitrary vector.

The general outline of our minimization procedure is then as follows. Provided that B is positive definite, which can easily be ensured in case of a convex decomposition, the search direction s is well defined by (3) and is downhill. We then perform a line search along s to achieve a sufficient reduction in f and to obtain an average positive curvature over the step. Each approximating Hessian B_i is then updated separately to B_i^+ such that

$$B_i^+ s_i = y_i = g_i(x+s) - g_i(x) \tag{6}$$

and symmetry is preserved. With a convex decomposition, we can also maintain positive definiteness because all inner products $y_i^T s_i$ are nonnegative.

3. Updating Semi-definite Hessian Matrices

Dropping the subscript "i", we consider in this section the problem of updating a dense approximation B to the Hessian $G(x)$ of an element function f which is convex in some convex region D in R^n. The convexity assumption is equivalent to the condition that $G(x)$ is positive semi-definite for all x in D. We are particularly interested in the case where the nullspace of $G(x)$ is known to contain a certain subspace N of R^n independent of x. This situation arises if f is constant with respect to certain translations of its variables so that

$$f(x+s)=f(x) \quad \text{for all } x \text{ in } D \text{ and all } s \text{ in } N \tag{7}$$

where we have assumed that $x+s$ also belongs to D. In many cases the subspace N is the same for several or even all the element functions so that we only need to store one set of vectors spanning N to perform elementary manipulations like projecting a vector into N.

After an arbitrary step from x in D to $x+s$ in D, the current approximation B is updated to B^+ such that the secant condition (6) is satisfied, where the subscript "i" is again omitted. Since G is known to be positive semi-definite with a nullspace containing N we want to maintain the same property for the approximating matrices B. Because f is only one of several element functions, the step s is not directly determined by B and g so that we cannot ensure that $y^T s$ is positive by imposing suitable conditions on the line search. However, by assumption on G, we have always that

$$\bar{G}=\int_0^1 (G(x+ts)\,dt) \tag{8}$$

has a nullspace containing N and that

$$y=\bar{G}s \quad \text{is in } N^* \tag{9}$$

where N^* denotes the orthogonal complement of the subspace N. This equations and $y^T s=s^T\bar{G}s$ being non-negative show that the secant condition (6) is consistent with the requirement that B^+ should be positive semi-definite with a nullspace containing N. It is well known ([4]) that the only rank two updates for which $y^T s>0$ is sufficient for hereditary positive definiteness are the formulae in the convex Broyden class spanned by the BFGS and the DFP updates. Therefore we examine the properties of these two formulae in the semi-definite case where $y^T s$ and $s^T Bs$ can be arbitrarily small or even zero.

Whenever

$$y^T s>0 \tag{10}$$

the DFP update

$$B^+_{\mathrm{DFP}}=B-\frac{Bsy^T+ys^TB}{y^Ts}+\frac{yy^T}{y^Ts}\left(1+\frac{s^TBs}{y^Ts}\right) \tag{11}$$

is well defined for arbitrary symmetric positive semi-definite B. An elementary calculation shows that for, all z in R^n,

$$z^T B^+_{\text{DFP}} z = \left\{ \sqrt{z^T B z} - \sqrt{s^T B s}\, \frac{|y^T z|}{y^T s} \right\}^2 + \frac{(y^T z)^2}{y^T s}$$
$$+ 2\{\sqrt{z^T B z s^T B s} - \operatorname{sign}(y^T z) z^T B s\} \frac{|y^T z|}{y^T s}. \tag{12}$$

By using the Schwarz inequality, one can see that the bracket occuring in the last term of the right hand side is always non-negative, and hence we can conclude to hereditary positive semi-definiteness. It follows furthermore by standard arguments that

$$y^T z = 0 \quad \text{implies} \quad z^T B^+_{\text{DFP}} z = z^T B z \tag{13}$$

and

$$\operatorname{range}(B^+_{\text{DFP}}) = \operatorname{range}(B) + \operatorname{span}[y]. \tag{14}$$

The implication (13) means in particular because of (9) that the restriction of B to N remains unchanged throughout the calculations. Thus we see that if the nullspace of the initial B does not contain N, the discrepancy $G - B$ cannot become arbitrarily small, which will in general prevent superlinear convergence. Therefore, the initial nullspace, if it cannot be made exactly equal to N should be chosen rather too large than too small, and, in theory, we could even initialize B to the zero matrix. Then it follows from (14) that the rank of B is increased by one whenever y is not in its range until B has the right nullspace N. However, besides the obvious difficulty that the low rank of the early B's is likely to yield a singular system for the quasi-Newton step, the DFP update may still have serious instabilities which will be investigated later.

The BFGS update

$$B^+_{\text{BFGS}} = B - \frac{B s s^T B}{s^T B s} + \frac{y y^T}{y^T s} \tag{15}$$

is only well defined if we have (10) and

$$s^T B s > 0 \qquad \text{or equivalently} \quad B s \neq 0. \tag{16}$$

At first sight, this may appear a disadvantage since it can also be seen that B^+_{BFGS} is highly discontinuous for $s^T B s$ near zero. In analogy to (12), we find for any z in R^n,

$$z^T B^+_{\text{BFGS}} z = \left\{ z^T B z - \frac{(z^T B s)^2}{s^T B s} \right\} + \frac{(y^T z)^2}{y^T s} \geq 0, \tag{17}$$

where it follows again from the Schwarz inequality that the bracket on the right hand side is non-negative. Thus we have again heriditary positive semi-definiteness, but, this time, the range of B is not simply enlarged to include y, but modified such that always

$$\operatorname{rank}(B^+_{\text{BFGS}}) = \operatorname{rank}(B). \tag{18}$$

This equation follows from the fact that the rank drops by one when the second term on the right hand side of (15) is substracted, and then increases by one when the last term is added. Thus we see that, if B was initialized with a rank less than $n-\dim(N)$, then its nullspace can never be contained in N, which is again likely to prevent superlinear convergence. If, on the other hand,

$$\operatorname{rank}(B) \geqq n-\dim(N), \tag{19}$$

then the BFGS update performs very well, since, by (17),

$$y^T z=0 \quad \text{implies } z^T B^+_{\text{BFGS}} z = z^T B z - \frac{(z^T B s)^2}{s^T B s}. \tag{20}$$

This means that even if the initial nullspace differs from N, the restriction of B to N will be reduced whenever Bs is not in the orthogonal complement of N. To a lesser extend, this is also true for all other updates in the convex Broyden class, with the exception of DFP as noted above. If already $Bz=0$ for some z in N, then it follows from (9), (12) and (17) that also $B^+z=0$ for all updates in the convex class. However, if the nullspace of $G(x)$ varies with x, or N is not known beforehand, then only the BFGS update can be expected to yield good approximations B. To see this, we establish the following boundedness result.

Proposition 1. Let the Hessian $G(x)$ of $f(x)$ be continuous and positive semi-definite at all points x in some convex compact subset D of R^n. Given any symmetric positive semi-definite $n\times n$ matrix B, let the set S consist of all vector pairs (s, y) in R^{2n} such that, for some x in D with $x+s$ in D,

$$y=g(x+s)-g(x) \quad \text{and} \quad y^T s>0, \tag{21}$$

where $g(.)$ is the gradient of $f(.)$. Then the matrix B^+_{BFGS} as defined in (15) is uniformly bounded over all (s, y) in S with $s^T B s \neq 0$. For B^+_{DFP} as defined in (11) to have the same property, it is necessary that, for all x in D,

$$\text{nullspace}[G(x)] \subseteqq \text{nullspace}[B] \qquad \text{or } G(x)=0. \tag{22}$$

Furthermore, if, for some x in D,

$$\text{nullspace}[B] \nsubseteqq \text{nullspace}[G(x)], \tag{23}$$

then there are pairs (s, y) in S such that

$$\operatorname{rank}(B^+_{\text{DFP}}) = \operatorname{rank}(B)+1. \tag{24}$$

Proof. It was shown by Powell in [17] that, for all (s, y) in S,

$$\frac{\|y\|^2}{y^T s} \leqq c_0 = \max\{\|G(x)\| \text{ for } x \text{ in } D\}. \tag{25}$$

Hence, it follows from the positive semi-definiteness of

$$B - \frac{B s s^T B}{s^T B s} \tag{26}$$

and the triangle inequality that

$$\|B^+_{\mathrm{BFGS}}\| \leqq \|B\| + c_0, \tag{27}$$

which proves the first assertion. It is well known (see [4]) that

$$B^+_{\mathrm{DFP}} = B^+_{\mathrm{BFGS}} + (u-v)(u-v)^T, \tag{28}$$

where

$$u = \sqrt{s^T B s}\,\frac{y}{y^T s} \quad \text{and} \quad v = \frac{Bs}{\sqrt{s^T B s}} \tag{29}$$

Since always

$$\|v\| \leqq \sqrt{\frac{s^T B B s}{s^T B s}} \leqq \sqrt{\|B\|}, \tag{30}$$

it follows from (27) and (28) that B^+_{DFP} is uniformly bounded if and only if it is true for

$$\|u\| = \frac{\|y\| \sqrt{s^T B s}}{y^T s} \leqq \sqrt{c_0 \frac{s^T B s}{y^T s}}, \tag{31}$$

where the last inequality holds by (25). Now suppose there are vectors z and d with norm equal to one half such that

$$G(x) z = 0 \neq G(x) d \quad \text{and} \quad Bz \neq 0. \tag{32}$$

Because G is, by assumption, continuous, there is a monotonically decreasing sequence r_j converging to zero with

$$r_0 \leqq 1 \tag{33}$$

such that, for all s in R^n,

$$\|s\| \leqq \frac{1}{j} \quad \text{implies} \quad \|G(x+s) - G(x)\| \leqq r_j. \tag{34}$$

Considering the vector sequences

$$s_j = \frac{1}{j}(z + d r_j^{1/3}) \tag{35}$$

and correspondingly

$$\begin{aligned} y_j &= g(x+s_j) - g(x) \\ &= G(x) s_j + O[r_j \|s_j\|] = \frac{1}{j} r_j^{1/3} G(x) d + O\left[\frac{1}{j} r_j\right], \end{aligned} \tag{36}$$

we find that

$$\|y_j\| = \frac{1}{j} r_j^{1/3} \{\|G(x) d\| + O[r_j^{2/3}]\}, \tag{37}$$

$$y_j^T s_j = \frac{1}{j^2} r_j^{2/3} \{d^T G(x) d + O[r_j^{1/3}]\}$$

and

$$s_j^T B s_j = \frac{1}{j^2}\{z^T B z + O[r_j^{1/3}]\}. \tag{38}$$

Thus (21) is satisfied for sufficiently large j and we have

$$\lim_j (y_j^T s_j)^2/(y_j^T y_j s_j^T Bs) = 0, \tag{39}$$

which implies that $\|u\|$ as defined in (31) and consequently B_{DFP}^+ are unbounded. To prove the last assertion, suppose there is a vector z in R^n with

$$G(x)z \neq 0 \quad \text{and} \quad Bz = 0. \tag{40}$$

By making s a sufficiently small multiple of z, we can ensure that (21) holds with y outside the range of B, so that (24) holds by (14). Q.E.D.

Considering the proposition, we note that even though the BFGS update is not defined if either $y^T s$ or $s^T Bs$ are exactly zero, it does not „blow up" if either of these quantities is very small. The same can only be true for all other formulae in the convex class as long as condition (22) is satisfied. However, if, for some x in D, the nullspace of $G(x)$ is properly contained in that of B, then it follows from (24) that the nullspace of B is likely to be reduced in dimension until it is contained in that of $G(x)$ for all x in D. Unless we are lucky in that, for all x in D,

$$\text{nullspace}\,[G(x)] = N = \text{nullspace}\,[B], \tag{41}$$

the subsequent updates are then prone to blow up if we do not use the BFGS formula throughout.

We feel that these properties show interesting differences between the DFP and BFGS updates that uniformly favour BFGS in our context. These observations may also cast some light on the difference between the two formulae in the usual framework of unconstrained minimization. Since the numerical results reported in Sect. 6 do also indicate a clear superiority of the BFGS formula, we will only consider this update in the rest of this section.

We now turn to the important question of how to use in practice the information about the nullspace N that is known a priori while maintaining numerical stability in our calculations. In order to preserve the theoretically excellent properties of the BFGS formula in the presence of rounding errors, it is best to store and update B in factored form (see [7]). Because the next quasi-Newton step depends on all approximating element Hessians, there is no need to solve a linear system in B. However, since $G(x)$ is typically singular and $y^T s$ and $s^T Bs$ can be arbitrarily small, the problem of numerically maintaining positive semi-definiteness is much more serious than in the case of standard variable metric updates.

The representation of B can be simplified if the user is prepared to supply a $p \times n$ matrix Q with full row rank $p = \dim(N)$ such that

$$N = \text{range}(Q^T). \tag{42}$$

By a suitable ordering of the variables of f, we can always ensure that the last p columns of Q form a nonsingular matrix Q'. Since postmultiplication of the transposed Q^T by $-(Q')^{-T}$ does not change its range N, we may assume without loss of generality that Q takes the form

$$Q=(Q_0, -I) \tag{43}$$

where Q_0 is an arbitrary $p\times(n-p)$ matrix. Then it can be easily checked that for any symmetric B

$$\text{range}(B) \text{ is in } N^* \tag{44}$$

if and only if

$$B=(I, Q_0^T)^T B_0 (I, Q_0^T) \tag{45}$$

where B_0 is the leading $(n-p)\times(n-p)$ submatrix of B. Thus we can maintain (44) and positive semi-definiteness by updating the factorization

$$B_0=L_0 D_0 L_0^T \tag{46}$$

where L_0 and D_0 are the leading $(n-p)\times(n-p)$ submatrices of L and D respectively. As in the case of the minimal surface problem discussed in Sect. 6, Q_0 is often the same for several element functions and also sparse, which allows savings in storage and the number of multiplications per matrix vector product.

Therefore we conclude that if f is convex but $G(x)$ not necessarily nonsingular and if the steps are not directly related to g and B, then the BFGS update seems the only stable way of satisfying the secant condition while maintaining positive semi-definiteness. In practice B should be stored and updated in factored form which guarantees semi-definiteness and allows savings in storage and computing time, especially if a basis of the constant nullspace N is provided by the user. Provided the approximating element Hessians are suitably initialized, the resulting quasi-Newton update is invariant with respect to all those linear transformations on the variables that leave the decomposition (4) unaffected.

4. Algorithmic Considerations

We now turn to some questions that arise in the implementation of the method outlined at the end of Sect. 2.

The first obvious observation is that B_i, g_i and f_i do not fully determine s_i because s_i is just the projection of the overall step s, the solution of (3) with B defined in (5), on the relevant subspace. As a consequence, a number of familiar relations and properties of the steps in classical Quasi-Newton procedures are not true anymore. In particular conjugacy and even linear independence of successive search directions are lost. It is not unusual indeed that, for the same i, the projections s_i and s_i^+ of two successive steps s and s^+ are nearly linearly dependent.

A second interesting observation is that f is usually evaluated by summing the element function values in a certain order. This fact can be exploited in the linesearch because it may happen that the computed values of a few element functions make it impossible for the overall value of f to yield the reduction we ask for. At this stage, one may argue that there is no need to evaluate the remaining element functions and we can therefore, in theory, spare some computing time. In practice however, a lot depends upon the order in which the element functions are evaluated: using the given ordering one often detects an overshoot of the function value too late and other orderings may be costly to compute. In our experience, we can nevertheless achieve a saving in computing that lies between 20 and 40% of the time spent evaluating f. This is using an ordering that measures how "active" a particular element function is. Since savings of that magnitude are worthwhile when the cost of evaluating element functions is high, we believe that further thought should be devoted to the topic and conclusions will be presented at a later stage.

A third question that has to be considered is how Eq. (3) should be solved in practice. In our first implementation for the convex decomposition case, we use a conjugate gradient technique with diagonal preconditioning. The choice of a conjugate gradient procedure is, in our opinion, supported by several reasons:

1. It has the advantage that the overall matrix B of (5) needs never be assembled explicitly from the B_i. All that is needed is a procedure to obtain the product Bz for a given vector z in R^n, which can trivially be achieved by summing the $B_i z_i$, where z_i is again the projection of z on the relevant subspace.
2. Furthermore, the solution of (3) in this manner creates no fill-in and is usually fast on systems arising from discretized problems, especially when dealing with three dimensional finite elements. We also believe that a more sophisticated preconditioning technique like SSOR (see [1] or [11]) can further improve the procedure.
3. An iterative solver such as conjugate gradients has the intrinsic advantage of providing better and better approximations to the solution. In practice, one is often satisfied with a rather crude approximation of the step defined by (3), especially in the early stages of the minimization, when the quadratic model (2) is still quite poor. It will be shown in the next section that, on our test examples, the inexact solution of (3) is very satisfactory from the computing point of view.
4. The coding of a conjugate gradient solver is simple. Although these advantages are significant, we do not feel that the last word has been said. Much depends on the particular structure and on the actual positive definiteness of B. This is a question where further numerical testing is needed.

We end these algorithmic remarks by a few comments on first iteration scaling. The question of making the sequence of points generated by a quasi-Newton method independent of the scaling of the objective function by a constant factor has been examined already by several authors ([15, 19]). In [19], Shanno and Phua recommended the premultiplication of the initial approximation to the Hessian by a scalar factor that would correct the "lack of scale" of this first matrix (often chosen as the identity matrix). We adopted this point of view and we comment below on its usefulness. We were even able to go further in this direction, as every single element Hessian can be scaled with a specific appropriate factor. Therefore, the overall result of this scaling is not a multiple of the initial matrix but incorporates information about the different scales that may appear in different element functions. We used the factor

$$\frac{y_i^T s_i}{s_i^T B_i s_i} \tag{47}$$

for scaling up the initial Hessian approximation corresponding to the element function f_i.

5. A Mesh Refining Method

In this section, we briefly explore the concept of "mesh refinement" applied in an unconstrained minimization context. As mentioned already, many of the practical problems come from the calculus of variations, with an objective function f defined on a domain in some infinite dimensional Banach space X. Then the problem must be discretized, for instance using finite differences or finite element techniques, before it can be solved on a digital computer. For certain values of a discretization parameter $h>0$, one attains an objective function f_h defined on a finite dimensional linear space X_h which can be mapped by a "prolongation" P_h into X. If the discretization is stable and consistent at a minimum x^* of f as defined by Stetter in [20], then the stationary conditions

$$\operatorname{grad}[f_h(x)]=0 \tag{48}$$

have solutions x_h^* in X_n such that for some "order of accuracy" m

$$P_h x_h^* = x^* + O(h^m). \tag{49}$$

Here we have assumed that h is in fact a discretization width, e.g. the maximal diameter of any finite element. Then the number of variables $\dim(X_h)$ grows typically with h^{-2} or h^{-3}, so that the solution of the linear system (3) via the full Cholesky decomposition of B or $G(x)$ requires some $O(h^{-6})$ or $O(h^{-9})$ arithmetic operations. By exploiting sparsity of the coeficient matrix, or using an iterative scheme to solve the system only approximately, the computational effort for each Newton-like step can be substantially reduced. However, in general, it will still grow rapidly as h becomes smaller. Therefore the number of iterations at some minimal discretization width h^*, which depends on the desired accuracy on the solution, should be kept as low as possible.

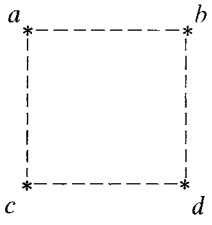
Even if f is quadratic, the close relationship between the Hessian of f_{h^*} and that of f_h for $h>h^*$ can be used to solve the linear system corresponding to h^* efficiently by an iterative scheme, for instance by successive overrelaxation. The multigrid approach reviewed by Brandt in [2] alternates between several discretization levels whereas the older idea of mesh refinement is based on a monotonically decreasing sequence $h(k) \to h^*$. The latter approach is particularly useful in the nonlinear case where one can prolong the solution $x^*_{h(k)}$ from $X_{h(k)}$ to a starting point in $X_{h(k+1)}$. In case of the partitioned variable metric approach proposed in this paper, we can also adjust the approximating element Hessians to the next discretization level. In our minimum surface and minimum volume examples (and for functionals having certain homogeneity properties), this adjustment is a trivial scaling. In the nonhomogeneous case, one may scale on the basis of the leading term, i.e. of highest power in h^{-1}, but this requires some analysis of the problem at hand.

If the minimization method is superlinearly convergent, it would seem appropriate to solve each discretized problem up to the finally required accuracy and to use a simple prolongation procedure, e.g. piecewise linear interpolation. A few additional steps at the coarser level may not make much difference with respect to the resulting starting point at the finer grid, but could substantially improve the approximating element Hessians.

6. Some Numerical Results

In this section, we present some preliminary numerical results. These results show, in our opinion, the potential power and usefulness of the approach of partitioned updating. On the other hand, they are far from being extensive and the conclusions that we can draw from them are to be taken with caution.

We considered, for testing purposes, two problems that arise from the discretization of the Euler variational equation. The first one is a discretization on the unit square of the well-known "minimal surface" problem: given a function on the boundary of the unit square, we want to find a surface that interpolates the given function on the boundary and that has minimum area. This problem is rather classical (see [6], for example) and can be discretized in the following way. The unit square is subdivided into m small squares and we compute the surface height at every corner of these small squares. If the small squares have the structure



where a, b, c and d represent the height of the surface at the four corners, then the surface of that element can be approximated by

$$s(a, b, c, d) = \frac{1}{m}\sqrt{1 + \frac{m}{2}\left[(a-d)^2 + (b-c)^2\right]}. \tag{50}$$

The overall objective function is then the sum of those small areas over the m squares. Each of the element functions thus involve four variables and has a full 4×4 Hessian matrix that has a fixed nullspace of dimension 2 (it is clear that the pairs $[a, d]$ and $[b, c]$ can be translated by a constant without modifying the value of $s(a, b, c, d)$). The boundary condition on the unit square is taken as

$$b(x, y) = 4x - 8y + 9 \tag{51}$$

and the starting values are set to zero for all the variables that are not constrained by the boundary condition. This problem will be called the "linear minimal surface" problem (LMS) in the sequel.

The second test example is the direct generalization of the first one to a higher dimension: find the minimal volume in the four dimensional hypercube subject to a three dimensional boundary condition. The discretization is obtained along the same lines as for the previous example and is therefore not described explicitly. In this new setting, each element function involves eight variables and has a full 8×8 Hessian matrix with a five dimensional nullspace. We used two different boundary conditions with this problem, i.e.

$$b(x, y, z) = 2x + 4y + 10z + 1 \tag{52}$$

and

$$b(x, y, z) = 10x^2 + 2x + 4y + 10z + 1. \tag{53}$$

The starting point is also given by setting to zero all the free variables. The resulting problems will be referred as the "linear minimal volume" (LMV) and "nonlinear minimal volume" (NLMV) problems depending whether condition (52) (for LMV) or condition (53) (for NLMV) is used.

We now list the different algorithms that were tested:

1. N: Newton's algorithm where the second derivative matrices are obtained by using forward differences in the gradients of the element function.
2. CG: conjugate gradient algorithm as described by Powell in [16].
3. NQN: a mixed Newton+quasi-Newton algorithm as proposed by Toint in [23]. This algorithm uses a sparse generalization of the BFGS updating formula that was proposed in [18] and [22]. The computation of the step does not require a very accurate solution of the involved linear system.
4. SPSB: a quasi-Newton algorithm implemented along the lines of NQN, except that it uses a sparse PSB update (see [21]) at every iteration. The initial approximation to the Hessian matrix is computed from forward differences in the gradient vector.
5. SBFGS: the same algorithm as SPSB, but with the sparse generalization of BFGS updating at each iteration.
6. PBFGS1: an algorithm that uses the partitioned BFGS update proposed above with the first approximate Hessian computed by finite differences in the element gradient vectors.
7. PBFGS2: the same algorithm as PBFGS1, but with the first approximate Hessian matrix chosen in the following way: each element Hessian is set to $I - NN^T$ where N is a matrix whose columns form an orthonormal basis of the nullspace associated with the corresponding element function.

8. PBFGS3: the same algorithm as PBFGS1, but with the first approximate Hessian set to an identity matrix that is scaled using (47).

9. PBFGS4: the same algorithm as PBFGS2, but with each element Hessian scaled at first iteration (as in PBFGS3).

10 PDFP1, PDFP3, PDFP4: as PBFGS1, PBFGS3 and PBFGS4 respectively, but with a partitioned DFP update.

All tests presented below but one were run on the DEC 2060 computer of the Facultes Universitaires de Namur (Belgium) in double length arithmetic. The numbers concerning the conjugate gradient algorithm were obtained on the Cambridge University (U.K.) IBM 370/165 in double precision.

We now present the numerical results. The first comparison (Table 1) is between Newton's algorithm (N), the conjugate gradient algorithm (CG), sparse updating algorithms using sparse counterparts of the PSB and BFGS formulae (SPSB and SBFGS), the hybrid Newton+quasi-Newton algorithm (NQN) and partitioned updating algorithms using DFP and BFGS formulae (PDFP1 and PBFGS1). These algorithms were tested on the minimum surface problem (LMS) for moderately large number of variables. In this Table, n stands for the number of free variables in the problem, n' for the total number of discretization variables (including the fixed boundary values), nit is the number of iterations needed to reach an accuracy of 10^{-7} on the function values, and nf is the overall number of calls to the gradient. The number nf also incorporates the gradient calls that are needed for the finite difference schemes when the true Hessian matrix is estimated.

As one can seen in Table 1, the partitioned update algorithms have a clear advantage and this is especially true for the partitioned BFGS. It can also be observed that sparse updating formulae may be pretty bad when used alone, but that their performance is improved by a combination with estimations of the true Hessian matrices as discussed in [23].

The apparent superiority of the partitioned BFGS led the authors to investigate different possible initializations of the approximate Hessian in this algorithm. This comparison is presented in Table 2 below. In this Table, the symbols n, n', nit and nf have the same meaning as above.

We can clearly conclude from Table 2 that it is quite important from the numerical point of view that the first approximation to the element Hessian matrices have the right nullspace associated with the corresponding element functions. Otherwise the correct nullspace is approximately obtained in the end, the convergence is slowed down considerably. The second conclusion that can be drawn from Table 2 is that initialization to the true Hessian is not that

Table 1. Comparison with existing algorithms

		N		CG		SPSB		SBFGS		NQN		PDFP1		PBFGS1	
n	n'	nit	nf	nit	nf	nit	nf	nit	nf	nit	nf	nit	nf	nit	nf
25	49	8	43	21	38	>70	>100	>70	>100	28	50	34	40	12	19
121	169	10	57	48	79	not run		not run		47	71	50	60	13	23

Table 2. Initialization of partitioned BFGS

		PBFGS1		PBFGS2		PBFGS3		PBFGS4	
n	n′	nit	nf	nit	nf	nit	nf	nit	nf
25	49	12	19	11	13	21	24	10	12
121	169	13	23	13	17	35	43	13	18
400	464	16	31	16	23	46	70	14	20
841	961	18	36	19	32	not run		21	39

helpfull and does not justify its cost. Finally, first iteration scaling does not seem to have much effect on this problem. This is because the scaling of the true Hessian matrix at the starting point is rather different from that at the solution. We will see later that it might nevertheless be important.

The next question to look at was whether or not a very accurate solution of (3) was needed in order to obtain good convergence properties for the method. Indeed, if weak accuracy is sufficient, the computing cost for the whole procedure can be substantially reduced. The next table (Table 3) presents the results of a comparison between high and low accuracy requirements for the algorithms N and PBFGS4. High accuracy means that the residual appearing in the preconditioned conjugate gradient method has to be reduced by a factor $t=10^{12}$ while low accuracy uses a factor $t=100$. Again, n, n′, nit and nf have the same meaning as above. The parameter ns now represents the total number of matrix-vector products needed in the iterative solution of the systems (3).

Table 3. Accuracy in the step definition

		N (high acc)			N (low acc)			PBFGS4 (h·acc)			PBFGS4 (l·acc)		
n	n′	nit	nf	ns	nit	nf	ns	nit	nf	ns	nit	nf	ns
25	49	8	43	158	8	43	40	10	12	208	11	13	66
121	169	10	57	541	15	92	285	13	18	801	13	15	135
400	464	10	55	983	18	126	777	14	20	1663	14	21	290
841	961	11	66	1709	17	95	679	21	39	3812	18	32	843

As one can see in Table 3, the choice of a relatively weak accuracy for the solution of (3) is quite interesting from the point of view of reducing the amount of computation per iteration. This allows the solution of reasonably high dimensional problems in a moderate computing time. One can also observe that, although lowering the accuracy requirements does not seem to affect the number of function evaluation too much in PBFGS4, this is not the case for Newton's method (N) where the reduction in the work for the solution of (3) is obtained at the price of a slight increase in the number of function and gradient calls. One can also observe that the performance of PBFGS4 is reasonably close to that of Newton's method, as should be expected from a quasi-Newton algorithm. This is especially true when $t=100$ is used.

At this stage, it was interesting to verify that these modifications did not affect the superiority of the partitioned BFGS algorithm compared to partitioned DFP. Therefore, both algorithms (with $t=100$) were run on the two small LMS problems, with results shown in Table 4. This Table also shows the behaviour of these two algorithms but with the "wrong" initialization, i.e. algorithms PBFGS3 and PDFP3.

Table 4. Partitioned BFGS versus partitioned DFP

		PBFGS3		PDFP3		PBFGS4		PDFP4	
n	n′	nit	nf	nit	nf	nit	nf	nit	nf
25	49	21	24	>75	>86	11	13	21	23
121	169	35	43	>75	>82	13	15	46	47

A brief examination of Table 4 shows that the conclusions we had above about the superiority of the partitioned BFGS are still valid. It is also clear from the comparison between PDFP3 and PBFGS3 that, in agreement with the theory, the partitioned BFGS algorithm is much less sensitive to an error in the initialization of the first approximate Hessian than the partitioned DFP.

We now turn to some tests that were performed with the mesh refining partitioned BFGS algorithm (MPBFGS) mentioned in Sect. 5. These tests all use the three dimensional minimum volume problem with its different boundary conditions and are presented in Table 5. In this Table, the parameters n, n′ and nit have the same meaning as above. The parameters ne, nef and cpu stand for the number of element functions, the number of calls to an element function and the cpu time (in seconds on DEC2060) respectively. These new quantities must be introduced because nf and ns become meaningless for MPBFGS.

Table 5. Mesh refining partitioned BFGS

				PBFGS4			MPBFGS		
probl.	n	ne	n′	nit	nef	cpu	nit	nef	cpu
NLMV	343	512	729	18	11776	190	17	4080	21
NLMV	1331	1728	2197	22	69120	1377	23	12258	67
NMLV	3375	4096	4913	not run			19	23984	155
LMV	343	512	729	23	14336	266	5	40	3
LMV	2197	2744	3375	35	115248	3480	not run		
LMV	3375	4096	4913	not run			5	40	3

There is no doubt, when looking at Table 5, that the mesh refinement approach is extremely efficient. Although the very low numbers obtained for this method for the problem LMV come from the linearity of its solution, it is clear from the numbers obtained from NLMV that worthwhile savings can be achieved by applying a multigrid type algorithm when possible. It is also encouraging to observe that a rather nonlinear problem as NLMV in more than 3000 variables can be solved in less than 3 min on an average mainframe computer.

Finally, tests have been performed to analyze the influence of first iteration scaling in the multigrid approach. It turned out that this influence was rather important: a gain of about 50% on the cpu time and the number of function calls can be achieved by a proper scaling of the first approximate Hessian in this context.

7. Concluding Remarks and Perspectives

The main conclusion one can draw at this stage is that the approach of partitioned updating looks promising when applied to a convex decompositon of the objective function. It is also clear that additional experience is needed to establish its practical value.

Some important issues related to this new approach still remain unexplored and should be analysed in the near future, the main one being the case when no convex decomposition of f is available. This problem is not easy because the convexity assumption plays an important role in the theory presented above since otherwise not all approximating element Hessians can be updated by the BFGS formula. Then, it is not clear what updating formula could be applied with success. Preliminary experiments show, for instance, that using the PSB update does not result in an appreciable gain compared to the sparse updating techniques. An interesting idea is to "convexify" the decomposition at hand by shifting quadratic terms as discussed in [10] or to reestimate a true element Hessian when convexity is not achieved.

Other questions of importance are

1. The theory of convergence and rate of convergence of partitioned update algorithms. A proof of local and superlinear convergence of the partitioned BFGS algorithm will be presented in a forthcoming paper.

2. The application of the technique of partitioning to parallel computations in minimization. It can be seen that the proposed method lends itself quite naturally to this approach. A possibly efficient parallel algorithm for optimization could be developped in this framework.

3. The application of the decomposition idea to other decompositions than sums of element function. The idea is more to update pieces of information from which the approximate Hessian can be deduced: nothing a priori precludes other simple combinations of element functions like composition or products.

4. The introduction of constraints in the problem. This is certainly a nontrivial extension and will require more knowledge of the process, but it can be dreamed that the second derivative matrix of the Lagrangian function can be partitioned in smaller pieces involving only few variables. For example,

constraint functions that involve only a few variables could be treated separately.

Acknowledgements. It is a pleasure to acknowledge here the numerous and helpful suggestions that were made by L.C.W. Dixon, especially on the aspects of parallel computing (see [5]). The authors are also indebted to A. Buckley who read an early version of the paper, to M.J.D. Powell who pointed out an error in a proof and to a number of nice people present at the 1981 meeting on Mathematical Programming in Oberwolfach.

References

1. Axelsson, O.: Solution of linear systems of equations: Iterative Methods. In: Sparse matrix techniques, V.A. Baker ed. Copenhagen 1976. Springer Verlag Berlin 1976
2. Brandt, A.: Multi-level adaptative solutions to boundary value problems. Math. Comput. **31**, 333-390 (1977)
3. Davidon, W.C.: Variable metric method for minimization. Technical Report #ANL-5990 (Rev.), Argonne National Laboratory, Research and Development 1959
4. Dennis, J.E., Moré, J.J.: Quasi-Newton methods: Motivation and theory. SIAM Rev. **19**, 46-89 (1977)
5. Dixon, L.C.W.: The solution of the Navier-Stokes equations via finite elements and optimization on a parallel processor. Presented at the CEC/CREST International Summer School on Numerical Algorithms for Parallel Processors. Bergamo, Italy June 1981
6. Ekeland, I., Temam, R.: Convex analysis and variational problems. North-Holland: Amsterdam 1976
7. Fletcher, R., Powell, M.J.D.: On the modification of LDL^T factorisations. Math. Comput. **28**, 1067-1087 (1974)
8. Gill, P., Murray, W.: Conjugate gradient methods for large scale nonlinear optimization. Technical Report SOL 79-15, Department of Operations Research, Stanford University, Stanford 1979
9. Greenstadt, J.: Variations upon variable metric methods. Math. Comput. **24**, 1-18 (1970)
10. Griewank, A., Toint, Ph.L.: On the unconstrained optimization of partially separable functions. In: Nonlinear optimization, M.J.D. Powell ed. Academic Press: New York (1981)
11. Gustafsson, I.: Stability and rate of convergence of modified incomplete factorisation methods. Research Report 79.02 R Department of Computer Science, University of Göteborg (1979)
12. Huang, H.Y.: Unified Approach to quadratically convergent algorithms for function minimization. J. Optimization Theory Appl. **5**, 405-423 (1970)
13. Kamat, M.P., Vanden Brink, D.J., Watson, L.T.: Non-linear structural analysis using quasi-Newton minimization algorithms that exploit sparsity. Presented at the International Conference on Numerical Methods for Nonlinear Problems, Swansea, U.K. (1980)
14. Marwil, E.: Exploiting sparsity in Newton-like methods. PhD thesis, Cornell University, Ithaca, New York (1978)
15. Oren, S.S., Spedicato, E.: Optimal conditioning of self-scaling variable metric algorithms. Math. Programming **10**, 70-90 (1976)
16. Powell, M.J.D.: Restart procedures for the conjugate gradient method. Math. Programming **12**, 241-254 (1977)
17. Powell, M.J.D.: Some global convergence properties of a variable metric algorithm for minimization without exact line searches. SIAM-AMS Proceedings **9**, 53-72 (1976)
18. Shanno, D.F.: On variable metric methods for sparse hessians. Math. Comput. **34**, 499-514 (1980)
19. Shanno, D.F., Phua, K.H.: Matrix conditionning and nonlinear optimization. Math. Programming **14** (1978)
20. Stetter, H.J.: Analysis of discretization methods for ordinary differential equations. Springer tracts on Natural Philosophy **23** (1973)

21. Toint, Ph.L.: On sparse and symmetric matrix updating subject to a linear equation. Math. Comput. **31**, 954-961 (1977)
22. Toint, Ph.L.: A note on sparsity exploiting quasi-Newton methods. Math. Programming **21**, 172-181 (1981)
23. Toint, Ph.L.: Towards an efficient sparsity exploiting Newton method. In: Sparse matrices and their uses. I.S. Duff ed. Academic Press: New York 1981
24. Toint, Ph.L., Strodiot, J.J.: An algorithm for unconstrained minimization of large scale problems by Decomposition in subspaces. Technical Report 76/1, Department of Maths, FUN Namur, Belgium 1976

Received September 16, 1981/March 2, 1982