Bima Journal of Science and Technology, Vol. 5(3) Jan, 2022. ISSN: 2536-6041

AN ALGORITHM FOR MINIMIZING A DIFFERENTIABLE FUNCTION THAT USES ONLY FUNCTION VALUES

Nagwai, A.P[1], Modi, B.[2]

[1]Gombe State University, Department of Mathematics,
Faculty of Science.
email:hodcomputerscienceou@gmail.com

[1]Gombe State University, Department of Mathematics,
Faculty of Science.
email: bmodi@gsu.edu.ng

ABSTRACT

The problem is to determine whether or not there exists a neighborhood of a given point in which a real-valued function of real-variables can be accurately approximated by a quadratic function. As such, this paper explores a better way to determine the local minimal point of the given function based on some critical assumptions and the algorithm that utilizes only function values. The result of the experiments show that MR1 with either search A or search C should be used because the terminal convergence is superior to that of the rank 2 schemes and the number of function evaluations required is only about half that required by Powell's no derivative scheme.

***Keywords*: MR1, local minimal point**

Introduction

Let f denote a real-valued function of n real-variables x. The problem then is to determine a point X^* such that: $f(X^*) \leq f(x)$, $X \in E^n$, under the assumptions that if (A – 1) X^* exists, then (A – 2) f is twice continuously differentiable. It then follows that in (A – 3) f, there is a neighborhood of X^* in which f can be accurately approximated by a quadratic function, which determines only a local minimal point of f. The critical assumption here is (A – 4) and as such, the algorithm can utilize only function values.

Related Works

Several algorithms have been proposed for this problem, however the ones that stand out include: Powell's no derivative scheme [1], Stewart's modification of Davidson-Fletcher–Powell scheme [2]. Furthermore, Powell's scheme if used as stated in [1] may generate even for quadratic functions [3], a non-minimal point. Powell suggested a modification [3] that corrects this problem, but the resulting scheme may degenerate into slow successive coordinate searches.

Zangwill [3] suggested an alternative modification by recommending inserting searches along coordinate lines at the beginning of each cycle. The coordinate lines are considered cyclically, and the searches continue until a line of descent is obtained. Clearly, if no coordinate line is a line of descent, then the algorithm has converged to a minimal point of *f*, since f is continuously differentiable. This idea of allowing searches along coordinate lines 'periodically' is used in the new algorithm.

Stewart's algorithm is quite different from Powell's. In Powell's algorithm no apparent approximation to the gradient of f are made. Stewart on the other hand, makes explicit approximations to the gradient of *f*, which are then used in the Davidson – Fletcher Powell (DEP) [4] recursive formula to obtain approximations H_i. The Approximations to the inverse of the Hessian of f iterates at X_i, i=1, 2,… At X_i, a search is made along the ray through X_i in the direction – H_i g_i^a, where g_i^a denotes the gradient approximation. In particular, the jth component of g_i^a, is given in Equation 1:

$(g_i^a,)^j = [f(X_i + h_i^a e_j) – f(X_i)] / h_i^j$………(1)

where, the function differences are computed along the coordinate lines $X = X_i + e_i$, $1 \leq j \leq n$, through X_j. Stewart has a clever formula for computing h_i^j. The matrices H_i, i=1, 2… are obtained formulas seen in Equation 2:

$H_i + 1 = H_i$- $(H_i \nabla g_i^a, >< H_i \Delta g_i^a,) / <\Delta g_i^a, H_i \nabla g_i^a > + (\Delta x_i >< \Delta x_i) / <\Delta x_i, \Delta g_i^a>$…..(2)

where, $\Delta g_i^{a=} g_{i+1}^a - g_i^a$, and $\Delta x_i = x_i + 1 - x_i$.

If f is a positive definite quadratic function f = $<Q_x$ x >/2+<c, x>, then the DFP scheme converges to the minimal point of f in n iterations. This finite convergence depends upon three relationships namely: G_i (which denotes the gradient of f). R-1 (which denotes the directions of search d_i) and $H_i g_i$(R-2) which indicates that the linear searches are accurate, with $< g_{i+1}, d_i > = 0$ and (R-3)$(\Delta g)_I = Q(\Delta x)_i$. Clearly, if $g_i^a \neq g_i$, then (R-1) is not satisfied. Moreover, if h_i varies with i, then (R-3) is not satisfied either. Hence, if the gradient approximation are inaccurate Stewart's scheme may converge very slowly or even terminate prematurely since the inner product $< g_i$ $H_i g_i^a>$ may vanish even if $g_i \neq 0$.

Furthermore, unless $g_i^a \to g_i$ as $i \to \infty$, the terminal convergence of Stewart's scheme is weak even when g_i^a represents an accurate approximation. However, for the case tested in [2] Stewart's scheme required half as many function evaluations considered worthy for further consideration. Actually, rank I scheme [5] is quite worthy of consideration, with $H_i+1=H_i-(W_i >< w_i)/ <\Delta g_i, w_i>$,where $W_i \equiv H_i(\Delta g_i)$ - Δx_i. The finite convergence properties of this scheme depends largely upon the following two conditions: That, (R – 3) and (R – 4) H_i are invertible for each i. As such, the directions of search can be arbitrary as long as they are independent, while minimizing linear searches are not required. (R – 3) is satisfied whenever h_I^j is constant over i (not j). Hence, searches along coordinate lines can be inserted without introducing errors into the Hessian approximation, and perhaps a further reduction in the number of function evaluations required can be obtained by using a non-minimizing linear search.

Methodology

Rank 1 recursion formula and the coordinate searches were combined with several other new ideas to obtain a new algorithm, MR1. The differences between this algorithm and Stewart's algorithm other than the recursion formula used for computing the matrices H_i, where i=1, 2 …are given in Table1. However, Table 1 requires some explanatory c on subscripts i which refer to the iteration number scripts j on the component of the vest consideration.

In some cases the iteration j has been omitted for simplicity. Moreover, it has been used on the coordinate direction, since directions of search other than used in MRI, and the algorithm must keep track of H_i and its inverse. Also, the approximation to the Hessian of f, G_i a^{jj} in the approximation to the jth component of the gradient of f at x_i is the jth diagonal element of G_i. This second order correction does more than just improve the terminal convergence of the algorithm.

In fact if f is a quadratic function and G_i = Q then g_i^a in MIR equals the gradient of f, g_i, for any size h^j. In MRI lines of search, not rays are used.

Table 1: **MRI** against **Stewart's** Algorithm

MR1	Stewart
Approximation to the jth component of the gradient	
$g^i = (\Delta f)^j / h^i - h_a^j \, {}^{ij}/_2$	$G^j \, ((\Delta f)^j / h^j$
Formula for h_j^i	
$H_{ji} = 21\ f(xi)n/aijj \mid \frac{1}{2}$ or it is fixed apriori for all i	Depends on several factors. Basically there are two formulas h_j^i and a central differencing option too
Directions of search, d_i	
$\pm H_i g_i^a$, and the coordinate lines through x_i	$-H_i g_i^a$
Initial step size in the linear searches	
Min $\{2, < g_i^a, d_i > / < G_i d_i d_j > \}$	Min $\{1, -2(f(x + i) - f_i) / < g_i^a, d_i > \}$
Restarting rules	
The number of function evaluations per linear search exceeds a given value for several successive searches, and (n+1) iterations have been completed since the latest restart	$A_i^{ij} < 0$ for some j <u>OR</u> $< g_i^a , H_i g_i^a > < 0$ or $+H_i g_i^a >$ is a direction of descent at x_i
Restart directions at x_i	
The coordinate lines through x_i	$-g_i^a$
$\pm H_i g_i^a$ are not directions of descent at x_i	
Searches along the coordinate lines through x_i	Algorithm terminates
Stopping rule	
Each coordinate line through x_i is not a line of	$\lVert d_i \rVert < €$

descent	$\lVert \Delta x_i \rVert$ € specified apriori

Implementation

As stated earlier, the rank one scheme allows arbitrary directions of search. The coordinate lines, which are any appropriate set of independent directions, would work just as well to prevent premature convergence of the algorithm.

Two linear search strategies were used in this approach. The first scheme, search crude, C is a non-minimizing scheme that attempts only to determine a point x on the line of search through x_i such that $f(x) < f(x_i)$. The other scheme, search `accurate` A, is a minimizing scheme that attempts to bracket the minimal point of f on the line of search and then fits a parabola to the data on this line. In both searches, the initial step is the one given in Table 1. Usually, this step is 1 unit in size and takes one to the stationary point of the current quadratic approximation as seen in Equation 3:

$$g_i(X) = \langle G_i x_i\ x\rangle /2 + \langle g_i^a - G_i x_i, x\rangle \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

Stewart used the F – P search scheme [5] which requires an apriori lower bound f_i on the function f. Such a bound may not be easy to determine after all.

Stewart's restarting strategy parallels that of the DFP strategy, by checking the positive definiteness of the matrices H_i. The strategy in MRI is totally different, because, the initial step size in each of the linear searches, utilizes the current approximations to the gradient and the Hessian of f. As such, it seems reasonable to connect the restarting procedure to the number of function evaluations required to perform the linear searches. Hence, if at each of q successive iterations, more than r function evaluations are required, where q and r depend upon the search strategy used, and n+1 iterations have been completed since the latest restart, then it is assumed that the current approximation to the Hessian is inaccurate and the algorithm is restarted.

When the algorithm is restarted, $G_i = H_i = 1$ and a coordinate line of descent is used, where the coordinate lines are considered cyclically inclined; the stopping rule in MRI, that a successive check of the coordinate lines through x_i yields no lines of descent. This however, prevents premature convergence, but does not provide sharp termination if g_i^a is not an accurate approximation to g_i . Furthermore, the algorithm was programmed and run on a computer with the standard test functions Rosenbrock, Powell, Fletcher – Powell, Chebyquad, and a 7–dimensional quadratic function. Stewart's paper [2] contains results only for cases where the compared with results obtained using MR1 with search A, or search C, MDFP with search A or Search C and MDFP1 with search A or search C. MDFP denotes the algorithm obtained from MR1, if the DFP scheme is used in updating the approximation to the Hessian. MDFP1 denotes the algorithm obtained from MDFP, if the gradient approximation is replaced by those used by Stewart. Hence, MDFP1 is a rank 2-type algorithm with a Stewart type gradient approximation, but with the MR1 rules with respect to the linear search schemes, restarting, directions of search, formula for h_i^j and stopping rule applied. For the other tests with inaccurate gradient approximations, MR1 was compared with MDFP and MDFP1.

Among those cases with g g_i^a and accurate approximation to the gradient, and $g_i^a \to g_i$ as $i \to \infty$; Stewart's scheme, and MR1, MDFP, and MDFP1 all with search A behaved quite similarly. This is with the exception of the one test function whose Hessian is singular at the minimal point, using Powell's function of four variables.

The function MR1 with search A or C, outperformed Stewart's scheme and the other rank 2 schemes. When g_i^a was not and 'accurate' approximation to the gradient of f, $g_i^a \nrightarrow g_i$ as $i \to \infty$; MR1 with either search A or search C produced significantly better reductions in $f^* \equiv f - f(m - m)$ than either rank 2 scheme before it degenerated in to pure coordinate searches.

Whether or not the number of function evaluations required to reduced f^* to a specified value can be reduced by use of a non-minimizing search needs to be answered, although not always. However, in certain cases significant reductions were obtained. Algorithm MR1 with search C generally required at each iteration, 0 or at most 1 extra function evaluation, other than the values $f(x_i)$ and $f(x_i+1))$ to perform a crude linear search. The first n iterations after a start or restart each generally required 2 or 3 function evaluations. Moreover, this decrease was balanced by a corresponding increase in the number of iterations, and Table 2 demonstrates this balancing effect.

Discussion of Results

The test function was the Chebyquad function with n = 8. The results are compared with Powell's no derivative scheme because Stewart [2] did not test this function. In Table 1, g_i^a is an accurate approximation to the gradient of f; the simplest version of Stewart's formula was used to compute each h_i^j. Observe that $h_i \to 0$ $i \to \infty$, because f(min)=.0035… >0. $f^* \equiv f(x_i) - f(\min)$, FE denotes the number of function evaluation and ITN denotes the iteration number. R denotes the numbers of times say n, where $n + H_i g_i^a$ is the direction of descent. CD denotes the number of coordinate lines used, and ND denotes the number of times the line $x = x_i + a H_i g_i^a$ was not a line of descent at x_i.

In Table 3, f = Powell's function of 4 variables, and the simplest version of Stewart's formula was used to compute h_i. Observe that $h_i \to o$ as $i \to \infty$ and h_i is small everywhere. Observe that MR1 with search A outperformed MR1 with search C and Stewart's scheme. In Table 4, again f = Powell's function of 4 variables. Here, h_i is fixed apriori; $h_i^j =. 05$, i =1, 2,…j = 1,…. J = 1,…,n. Four algorithms are compared, MR1 with search A, MR1 with search C, MDFP with search A, and MDFP with search A.

Clearly, MDFP1 which uses the Stewart gradient approximation is out of the running. The other three algorithms performed very similarly, where the basis of comparison is the number of function evaluations required to obtain a specified reduction in f^*, down to $f^* = 3x10^{-3}$. Both MR1 with search A and MR1with search C outperformed MDFP with search A. clearly, MDFP1 which uses the Stewart gradient approximation is out of the running. The other three algorithms performed very similarly, where the basis of comparison is the number of function evaluations required to obtain a specified reduction in f*, down to $f^*=3\times 10^{-3}$. Afterwards, MR1 with Search A and Search C both outperformed MDEFP with Search A.

However, one unsolved problem is the question of termination of the algorithm when g_i^a is not necessarily an accurate gradient approximation. Each of the algorithms considered degenerates into ordinary coordinate searches after some unspecified number of iterations, for example: MR1 with Searches C in table 4 degenerated into coordinate searches at step 42.

Table 5, considers the Fletcher-Powell function with $h_i^j = .05$, j=1,…,n, i=1,2…. In this case MDFP1 degenerated into coordinate searches at iteration 46 and the decrease to 6×10^{-2} was due to coordinate searches. MDFP died at iteration 42 and the decrease to 6×10^{-5} was attained before the algorithm degenerated. The numerical results clearly demonstrated that the rank 1 scheme MR1 will outperform a rank 2 scheme with Stewart's gradient approximations cannot be made arbitrarily small.

However, the scheme also demonstrate that the number of function evaluations required by MR1 with search A or C to achieve a specified reduction in f^* depends critically upon the degree of accuracy of the gradient approximation. For instance, looking at Tables 3, 4 with f = Powell's function of 4 variables. MR1 with search A, and accurate gradient approximations required 71 function evaluations to reduce f to 10^{-4}, while Powell's no derivative scheme on the other hand required 138 function evaluations to make the same reduction with a ratio of almost 2:1. However, $h_i^j \equiv .05$ MR1 with search A, required 160 function evaluations to make the same reduction and the balance was in favor of Powell's no derivative scheme.

Conclusion and Future Work

If restrictions are really not very restrictive, for example where $h_i^j = 10^{-4}$ for f = PF4V, then the numerical results as seen in Table 2 - 5 indicate that MR1 with either search A or search C should be used because the terminal convergence is superior to that of the rank 2 schemes and the number of function evaluations required is only about half that required by Powell's no derivative scheme. More detailed computational results and a few theoretical results are available in reference [6]. Determining which algorithm to use when h_i^j is restricted remains an open question to be addressed very soon.

TABLE 2: Table values of Iterations for Chebyquad function.

F= Chebyquad (n=8), $h_i^j = 2| f(\times_i)10^{-8}/G^j{}_i{}^j |^{\frac{1}{2}}$

Itn	MRI-C			MRI-A			Powell ND		
	F^*		FE	F*		FE	F*		FE
0	4	10^{-2}	1	4	10^{-2}	1	4	10^{-2}	1
4	3	10^{-2}	46	1	10^{-2}	60	7	10^{-3}	91
8	5	10^{-3}	85	4	10^{-3}	107	2	10^{-3}	194
16	2	10^{-3}	161	3	10^{-4}	201	2	10^{-5}	385
24	4	10^{-6}	238	2	10^{-12}	294	6	10^{-13}	537
30	3	10^{13}	292						
R	0			0			-		
SW	2			5			-		
CD	0			1			-		
DN	0			1			-		

TABLE 3: Table values of Iterations for Powell's function with $h_i^j = 2| f(x_i)10^{-8}/G^j{}_i{}^j |^{\frac{1}{2}}$

Itn	MRI-C		MRI-A		Powell ND	
	F*	FE	F*	FE	F*	FE
0	2 10^{-2}	1	2 10^{2}	1	2 10^{2}	1
4	5 10^{-1}	29	2 10^{-1}	31	7 10^{-2}	37
8	1 10^{0}	49	9 10^{-4}	36	3 10^{-3}	69
12	1 10^{-2}	69	8 10^{-8}	94	3 10^{-5}	104
16	1 10^{-4}	89	2 10^{-10}	127	1 10^{-8}	139
18	2 10^{-5}	99	8 10^{-11}	148	9 10^{-9}	158
24	3 10^{-8}	129	5 10^{-12}	212		
R	1		1			
SW	2		3			
CD	1		1			
DN	0		0			

F= Powell's function 0 (n=4), $h_i^j = 2| f(x_i)10^{-8}/G^j{}_i{}^j |^{\frac{1}{2}}$

TABLE 4: Table values of Iterations for Powell's function with $h_i^j = .05$

F= Powell's function (n=4), $h_i^j = .05$

Itn	MRI-A		MRI-C		MDFP-A		MDFPI-A	
	F	FE	f	FE	F	FE	f	FE
0	2 10^{2}	1	2 10^{2}	1	2 10^{2}	1	2 10^{2}	1
4	6 10^{-1}	30	5 10^{0}	21	6 10^{-1}	33	6 10^{-1}	29
88	1 10^{-1}	68	1 10^{0}	41	1 10^{-1}	71	5 10^{-1}	57
15	5 10^{-3}	137	8 10^{-2}	76	3 10^{-3}	140	4 10^{-1}	107
25	2 10^{-6}	245	3 10^{-3}	126	3 10^{-4}	222	8 10^{-2}	199
30	3 10^{-7}	305	9 10^{-4}	151	2 10^{-4}	270	1 10^{-2}	246
	↓		↓		↓		↓	
R	2		1		5		1	
SW	1		2		0		0	
CD	7		6		7		1	
ND	5		5		2		0	

TABLE 5: Table values of Iterations for Fletcher- Powell function

F= Fletcher- Powell function (n=3), $h_i^j = .05$

Itn	MRI-A		MRI-C		MDFP-A		MDFPI-A	
	f	FE	f	FE	F	FE	f	FE
0	2 10^{3}	1	2 10^{2}	1	2 10^{3}	1	2 10^{3}	1
4	3 10^{1}	26	1 10^{0}	17	3 10^{1}	26	2 10^{1}	26
88	4 10^{0}	77	5 10^{0}	55	7 10^{0}	89	9 10^{0}	81

15	2 10^{-2}	138	2 10^{-2}	95	4 10^{0}	150	6 10^{0}	140
25	3 10^{-6}	218	2 10^{-3}	143	2 10^{0}	217	2 10^{0}	203
30	3 10^{-7}	319	1 10^{-4}	192	2 10^{-1}	296	5 10^{-1}	272
	↓		7 10^{-5}	229	1 10^{-1}	332	3 10^{-1}	322
			↓		↓			
R	1		1		4		3	
SW	4		4		0		0	
CD	9		7		8		5	
ND	7		4		4		2	

References

[1] M. J. D. Powell, Comp. J. 7, 155, 1994.
[2] G. W. Stewart, J. Assoc. Mach. 14. 75, 1997.
[3] W. L. Zangwill, Comp, J. 11, 293, 2007.
[4] R. Fletcher and M.J. D Powell, Comp. J. 6, 163, 2003.
[5] M.J.D. Powell, Integer Progamming, ed J. Abadle, North-Holland, 139, 2012.
[6] J.Cullum,IBNJ.Res.Dev.,2014.