International Journal of Pure and Applied Mathematics

Volume 28 No. 2 2006, 187-196

EXPLOITING FUNCTION VALUES IN MULTI-STEP METHODS

Issam A.R. Moghrabi
Department of Computer Science
Faculty of Science
Beirut Arab University
P.O. Box 11-5020, Beirut, LEBANON
e-mail: i_moghrabi@yahoo.com

Abstract: We present in this work a new perspective on the derivation of quasi-Newton algorithms for unconstrained optimization. The new algorithms are intended as an improvement on the multi-step methods presented by the author in previous work. The parameterization of the polynomials take into account the spacing between the points in the gradient space as well as exploiting the function values readily available from the previous $m+1$ iterations in an m-multi-step method. This employment of function values is done through utilizing such values in a nononlinear power model. Numerical results suggest that the new method improves over other existing ones that belong to the same class.

AMS Subject Classification: 65K10
Key Words: unconstrained optimization, quasi-Newton methods, multi-step methods

1. Introduction

The multi-step methods derived previously by the author [10], [9] are based on the so-called "Newton equation" (Ford and Saadallah [12]), which may be regarded as a generalization of the "secant equation" (Dennis and Schnabel [3]).

Let $f(\underline{x})$ be the objective function, where $\underline{x} \in R^n$, and let $\underline{g}$ and G denote the gradient and Hessian of f, respectively. Let $X = \{\underline{x}(\tau)\}$ denote a differentiable path in R^n, where $\tau \in R$. Then, if we apply the chain rule to

Received: December 21, 2005

$\underline{g}(\underline{x}(\tau))$ in order to determine its derivative with respect to τ, we obtain

$$G(\underline{x}(\tau))\underline{x}'(\tau) = \underline{g}'(\underline{x}(\tau)). \tag{1}$$

In particular, if we arrange for the path X to pass through the most recent iterate $\underline{x}_{i+1}$ (so that $\underline{x}(\tau_m) = \underline{x}_{i+1}$), then equation (1) provides a condition (termed the "Newton equation" in [12]) which the Hessian $G(\underline{x}_{i+1})$ must satisfy:

$$G(\underline{x}_{i+1})\underline{x}'(\tau_m) = \underline{g}'(\underline{x}(\tau_m)). \tag{2}$$

Therefore, if B_{i+1} denotes an approximation to $G(\underline{x}_{i+1})$, if

$$\underline{r}_i \overset{\text{def}}{=} \underline{x}'(\tau_m) \tag{3}$$

and $\underline{w}_i$ denotes an approximation to $\underline{g}'(\underline{x}(\tau_m))$, it is reasonable (by equation (2)) to require that B_{i+1} should satisfy the relation

$$B_{i+1}\underline{r}_i = \underline{w}_i \tag{4}$$

(the derivation, in particular, of the secant equation from the Newton equation is described in [10]). In [10, 9], it was proposed that X should be the vector polynomial which interpolates the $m+1$ most recent iterates $\{\underline{x}_{i-m+k+1}\}_{k=0}^{m}$ and that $\underline{w}_i$ should be obtained by constructing and differentiating the corresponding vector polynomial ($\underline{\hat{g}}(\tau)$) which interpolates the known gradient values $\{\underline{g}(\underline{x}_{i-m+k+1})\}_{k=0}^{m}$. Thus, the following explicit expressions for $\underline{r}_i$ and $\underline{w}_i$ may be derived:

$$\underline{r}_i = \underline{x}'(\tau_m) = \sum_{j=0}^{m-1} \underline{s}_{i-j}\{\sum_{k=m-j}^{m} \mathcal{L}_k'(\tau_m)\}; \tag{5}$$

$$\underline{w}_i = \underline{\hat{g}}'(\tau_m) = \sum_{j=0}^{m-1} \underline{y}_{i-j}\{\sum_{k=m-j}^{m} \mathcal{L}_k'(\tau_m)\} \approx \underline{g}'(\underline{x}(\tau_m)) \tag{6}$$

(in the equations above,

$$\underline{s}_i \overset{\text{def}}{=} \underline{x}_{i+1} - \underline{x}_i, \tag{7}$$

$$\underline{y}_i \overset{\text{def}}{=} \underline{g}(\underline{x}_{i+1}) - \underline{g}(\underline{x}_i) \tag{8}$$

and $\mathcal{L}_j(\tau)$ is the j-th Lagrange polynomial of degree m corresponding to the set of values $\{\tau_k\}_{k=0}^{m}$, so that $\mathcal{L}_j(\tau_j) = 1$ and $\mathcal{L}_j(\tau_i) = 0$ for $i \neq j$. The scalars $\{\tau_k\}_{k=0}^{m}$ are the values of τ associated with the iterates $\{\underline{x}_{i-m+k+1}\}_{k=0}^{m}$ on the path $X = \{\underline{x}(\tau)\}$:

$$\underline{x}(\tau_k) = \underline{x}_{i-m+k+1}, \text{ for } k = 0, 1, \ldots, m). \tag{9}$$

In this paper, we will investigate a class of parameterized models (employing a nonlinear scaling factor) for the path X. The free parameter in such models can be viewed as providing a means by which more information can be utilized in updating the Hessian approximation (or its inverse), as in the methods of Ford and Ghandari [6, 7, 8]. We describe, in the next section, the particular technique (which essentially involves making use of the function-values at our disposal from the m most recent iterations) that will be used in determining the free parameter.

2. The Nonlinear Model

Preliminary experiments on algorithms derived in [10, 9] indicated that, of the methods considered, those for which $m = 2$ were generally superior to those corresponding to larger values of m. We therefore confine our attention to the case $m = 2$ in what follows. We use values of τ which are based on the algorithm **A2** described in [9]. In particular, this means that the origin for values of τ is taken to be τ_1:

$$\tau_1 = 0, \tag{10}$$

while the remaining values of τ are given by

$$\tau_0 = -[\underline{s}_{i-1}^T \underline{y}_{i-1}]^{1/2}; \quad \tau_2 = [\underline{s}_i^T B_i \underline{s}_i]^{1/2}.$$

We will consider paths $X = \{\underline{x}(\tau)\}$ in the space R^n which (like **A2**) interpolate the iterates $\underline{x}_{i-1}$, $\underline{x}_i$ and $\underline{x}_{i+1}$, respectively, but which have the following more general form:

$$\underline{x}(\tau, \theta) \equiv [\underline{a}_0 + \underline{a}_1 \tau + \underline{a}_2 \tau^2] \Upsilon(\tau, \theta), \tag{11}$$

where $\{\underline{a}_j\}_{j=0}^2$ are constant vectors and $\Upsilon(\tau, \theta)$ is a suitably-chosen, nonzero scaling function (a corresponding model, using the same scaling function and the same value of θ, will be employed for the gradient approximation $\underline{\hat{g}}(\tau, \theta)$). If we define the vector $\underline{z}(\tau, \theta)$ by

$$\underline{z}(\tau, \theta) \equiv \underline{x}(\tau, \theta) \Upsilon^{-1}(\tau, \theta), \tag{12}$$

then, evidently, $\underline{z}(\tau, \theta)$ is quadratic in τ and hence may be written in its "Lagrangian" form:

$$\underline{z}(\tau, \theta) \equiv \mathcal{L}_0(\tau) \underline{z}_{i-1} + \mathcal{L}_1(\tau) \underline{z}_i + \mathcal{L}_2(\tau) \underline{z}_{i+1}, \tag{13}$$

where $\underline{z}_{i-1}$, $\underline{z}_i$ and $\underline{z}_{i+1}$ are the values of $\underline{z}$ corresponding to the abscissae τ_0, τ_1 and τ_2, respectively. Therefore, from (12), we may write $\underline{x}(\tau,\theta)$ in the form

$$\underline{x}(\tau,\theta) \equiv \Upsilon(\tau,\theta)[\mathcal{L}_0(\tau)\underline{z}_{i-1} + \mathcal{L}_1(\tau)\underline{z}_i + \mathcal{L}_2(\tau)\underline{z}_{i+1}], \tag{14}$$

from which it follows that

$$\underline{x}'(\tau,\theta) \equiv \Upsilon(\tau,\theta)[\mathcal{L}_0'(\tau)\underline{z}_{i-1} + \mathcal{L}_1'(\tau)\underline{z}_i + \mathcal{L}_2'(\tau)\underline{z}_{i+1}] + \Upsilon'(\tau,\theta)\underline{z}(\tau,\theta). \tag{15}$$

Finally, therefore, we can express $\underline{x}'(\tau,\theta)$ in the form

$$\underline{x}'(\tau,\theta) \equiv \Upsilon(\tau,\theta)[\mathcal{L}_2'(\tau)\Delta\underline{z}_i - \mathcal{L}_0'(\tau)\Delta\underline{z}_{i-1}] + \Upsilon'(\tau,\theta)\underline{z}(\tau,\theta), \tag{16}$$

where $\Delta\underline{z}_j \overset{\text{def}}{=} \underline{z}_{j+1} - \underline{z}_j$ and where we have used the identity

$$\sum_{j=0}^{2} \mathcal{L}_j'(\tau) \equiv 0.$$

The question of how to determine a suitable value for θ, given a particular choice for Υ, now arises. The methods of Ford and Ghandhari [6, 7, 8] provide an illustration that efficient utilization of function-values in updating the Hessian approximation (or its inverse) can be profitable. This motivates us to consider (in the new context of multi-step methods) exploiting the function-values available from the last three iterates in order to determine a suitable value for θ. Although a number of approaches to attacking this problem might be considered, we will discuss only one of them here. The fundamental idea is to employ the three available function-values f_{i-1}, f_i and f_{i+1} (where $f_j = f(\underline{x}_j)$) to estimate the derivative (with respect to τ) of $f(\underline{x}(\tau,\theta))$ at the three abscissae $\{\tau_k\}_{k=0}^2$. We therefore approximate $f(\underline{x}(\tau,\theta))$ with the interpolating quadratic $\phi(\tau)$:

$$\phi(\tau_j) = f(\underline{x}(\tau_j,\theta)) = f_{i-1+j}, \text{ for } j = 0, 1, 2. \tag{17}$$

Then, on defining the additional constants

$$\mu = \tau_2 - \tau_0\ ; \quad \delta = (\tau_2 - \tau_1)/(\tau_1 - \tau_0) = -\tau_2/\tau_0\ ,$$

we may derive the following expressions for the required derivatives of ϕ:

$$\begin{aligned}
\phi'(\tau_0) &= [-\delta^{-1} f_{i+1} + (\delta^{-1} + 2 + \delta) f_i - (2+\delta) f_{i-1}]/\mu; && (18)\\
\phi'(0) &= [\delta^{-1} f_{i+1} + (\delta - \delta^{-1}) f_i - \delta f_{i-1}]/\mu; && (19)\\
\phi'(\tau_2) &= [(2+\delta^{-1}) f_{i+1} - (\delta^{-1} + 2 + \delta) f_i + \delta f_{i-1}]/\mu. && (20)
\end{aligned}$$

On the other hand, the derivative (with respect to τ) of $f(\underline{x}(\tau,\theta))$ may be obtained analytically from the chain rule:

$$df(\underline{x}(\tau,\theta))/d\tau \equiv [d(\underline{x}(\tau,\theta))/d\tau]^T \underline{g}(\underline{x}(\tau,\theta)).$$

Therefore, using the derivative estimates obtained above, we derive the following conditions (where $\underline{g}_j = \underline{g}(\underline{x}_j)$):

$$\begin{aligned} \underline{x}'(\tau_0,\theta)^T \underline{g}_{i-1} &= \phi'(\tau_0); && (21)\\ \underline{x}'(0,\theta)^T \underline{g}_{i} &= \phi'(0); && (22)\\ \underline{x}'(\tau_2,\theta)^T \underline{g}_{i+1} &= \phi'(\tau_2); && (23) \end{aligned}$$

and observe that any one these relations may (in principle) be used to determine θ.

3. A Power Model

We now make the particular choice

$$\Upsilon(\tau,\theta) \equiv (\theta+1)^\tau \tag{24}$$

for the scaling function Υ. From (12), we may deduce the relevant values of $\underline{z}$:

$$z((\tau,\theta) = (\theta+1)^{-\tau} x(\tau,\theta),$$

which may be substituted into equation (16) to provide the required expressions for the derivatives of $\underline{x}$: (for $\lambda \equiv \theta + 1$)

$$\begin{aligned} &\underline{x}'(\tau_0,\theta)\\ &\quad = \ln\lambda \underline{x}_{i-1} + \mathcal{L}_2'(\tau_0)\lambda^{-\mu} x_{i+1} - [L_2'(\tau_0) + L_0'(\tau_0)]\lambda^{\tau_0} x_i + L_0'(\tau_0) x_{i-1}, \end{aligned} \tag{25}$$

$$\underline{x}'(0,\theta) = \ln\lambda \underline{x}_i + \mathcal{L}_2'(0)\lambda^{-\tau_2} x_{i+1} - [L_2'(0) + L_0'(0)] x_i + L_0'(0)\lambda^{\tau_0} x_{i-1}, \tag{26}$$

$$\begin{aligned} &\underline{x}'(\tau_2,\theta)\\ &\quad = \ln\lambda \underline{x}_{i+1} + \mathcal{L}_2'(\tau_2) x_{i+1} - [L_2'(\tau_2) + L_0'(\tau_2)]\lambda^{\tau_2} x_i + L_0'(\tau_2)\lambda^{\mu} x_{i-1}. \end{aligned} \tag{27}$$

Hence, defining

$$\beta_{ij} = \underline{x}_i^T \underline{g}_j, \tag{28}$$

we can calculate (from (21), (22) and (23)) the expressions that are required for the equations which will be employed to determine θ:

$$\phi'(0,v) \\ = \ln\lambda\beta_{ii} + L_2'(0)\lambda_2^{-\tau}\beta_{i+1,i} - \left[L_2'(0)+L_0'(0)\right]\beta_{ii} + L_0'(0)\lambda^{-\tau_0}\beta_{i-1,i}\,, \quad (29)$$

$$\phi'(\tau_0,v) = \ln\lambda\beta_{i-1,i-1} + L_2'(\tau_0)\lambda_2^{-\tau}\beta_{i+1,i-1} \\ - \left[L_2'(\tau_0)+L_0'(\tau_0)\right]\lambda^{\tau_0}\beta_{i,i-1} + L_0'(\tau_0)\beta_{i-1,i-1}\,, \quad (30)$$

and

$$\phi'(\tau_2,v) = \ln\lambda\beta_{i+1,i+1} + L_2'(\tau_2)\beta_{i+1,i+1} - \left[L_2'(\tau_2)+L_0'(\tau_2)\right]\lambda^{\tau_2}\beta_{i,i+1} \\ + \lambda^{\mu}L_0'(\tau_2)\beta_{i-1,i+1}\,. \quad (31)$$

We now define three algorithms (to be denoted by **E1**, **E2** and **E3**, respectively), corresponding to the three equations (29), (30), (31) defining θ. For example, in algorithm **E2**, θ is defined (on each iteration) to be the solution of the equation:

$$\left[\ln\lambda\beta_{i,i} + L_2'(0)\right]\lambda^{-\tau_2}\beta_{i+1,i} - \left[L_2'(0)+L_0'(0)\right]\beta_{i,i} + L_0'(0)\lambda^{-\tau_0}\beta_{i-1,i} \\ - \mu^{-1}(f_{i+1}-f_{i-1}) = 0\,. \quad (32)$$

In each of the three methods, once the value of θ has been determined, the Hessian approximation B_{i+1} is updated to satisfy the relation (compare (2) and (4)):

$$B_{i+1}\underline{x}'(\tau_2,\theta) = \underline{\hat{g}}'(\tau_2,\theta), \quad (33)$$

where $\underline{x}'(\tau_2,\theta)$ is obtained from equation (27) and $\underline{\hat{g}}'(\tau_2,\theta)$ is the corresponding expression with each iterate $\underline{x}_j$ replaced by the related gradient $\underline{g}_j$.

4. Numerical Experiments

The algorithms **E1**, **E2** and **E3** developed in Section 2 were first compared with each other. Eleven test functions (taken from the literature) were used, each with either one or two starting-points, giving a total of twenty problems (full details of the test functions and starting points may be found in Ford and Moghrabi [11]). Many of the functions employed in the tests may be used with varying dimensions – in such cases, we have carried out the tests on a range of suitable dimensions and summed the results, since lack of space precludes a tabulation of the individual figures for each dimension.

In all the methods considered here, the new point $\underline{x}_{i+1}$ was computed from $\underline{x}_i$ via a line-search algorithm which accepted the predicted point if the two standard stability conditions (Fletcher [5]) given below were satisfied and which, otherwise, used step-doubling and safeguarded cubic interpolation, as appropriate. To be acceptable, $\underline{x}_{i+1}$ was required to satisfy the following conditions:

$$f(\underline{x}_{i+1}) \leq f(\underline{x}_i) + 10^{-4}\underline{s}_i^T \underline{g}(\underline{x}_i); \quad \underline{s}_i^T \underline{g}(\underline{x}_{i+1}) \geq 0.9\{\underline{s}_i^T \underline{g}(\underline{x}_i)\}.$$

In the actual implementations, the matrices $H_i \overset{def}{=} B_i^{-1}$ (instead of B_i) were stored and updated (using the appropriate form of the BFGS (Broyden [2], Fletcher [4], Goldfarb [13], Shanno [14]) formula for inverse Hessian approximations), in order to reduce the computational expense of calculating the search-direction on each iteration. Here the dimension of the problem was ten or higher, the initial inverse Hessian approximation $H_0 = I$ was scaled by the method of Shanno and Phua [15] before the first update was performed. The solution of the nonlinear equation which arises on each iteration of these new methods was carried out by means of Brent's algorithm [1].

It is easy to show (by analogy with standard theory for the BFGS method) that a necessary and sufficient condition for preserving positive-definiteness in the successive matrices $\{H_i\}$ is that $\underline{r}_i^T \underline{w}_i > 0$. In practice, we have imposed (in the implementations) the following requirement:-

$$\underline{r}_i^T \underline{w}_i > 10^{-4} \parallel \underline{r}_i \parallel_2 \parallel \underline{w}_i \parallel_2 ,$$

in order to ensure that $\underline{r}_i^T \underline{w}_i$ is "sufficiently" positive and thus avoid possible numerical instability in computing H_{i+1}. If this condition on $\underline{r}_i^T \underline{w}_i$ was not satisfied, the algorithm reverted to the choice $\theta = 0$.

The results of this first set of experiments are presented in Table 1. For each problem, the number of function/gradient evaluations required to solve the problem is given, followed (in brackets) by the number of iterations. The best performance for each problem (assessed by the number of function/gradient evaluations, with ties resolved on the basis of iterations) is indicated by the symbol "†". Totals (including "scores", indicating the number of best performances) are given at the foot of Table 1.

On the basis of these results (which indicate that **E3** gives the best numerical performance of the three new methods), and by comparing the results in Table 1, all three of the new methods perform significantly better than the standard, single-step, BFGS method.

Problem	**E1**	**E2**	**E3**	BFGS
1(a)	463(446)	443(429)†	443(429)†	542(530)
1(b)	2012(1127)	1511(1183)	1392(995)†	1653(1293)
2(a)	392(373)†	475(444)	500(469)	631(612)
2(b)	444(422)†	772(734)	596(467)	825(806)
3(a)	345(134)	301(224)	188(123)†	159(122)
3(b)	320(105)	124(89)†	150(92)	807(386)
4(a)	873(344)	865(242)	765(341)†	637(507)
4(b)	505(384)†	577(450)	632(385)	634(596)
5(a)	103(76)†	390(330)	171(130)	2390(2331)
5(b)	153(139)†	733(668)	593(535)	2300(1786)
6(a)	2488(1358)	1479(860)†	1836(928)	2136(2001)
6(b)	2014(1231)	1308(1150)	1181(1012)†	280(226)
7(a)	480(381)	287(241)†	298(243)	1967(965)
7(b)	1941(639)†	2043(975)	2091(1095)	5099(2272)
8(a)	3214(2350)†	4918(1939)	5830(2175)	2822(1969)
8(b)	2606(2247)†	3575(2174)	2640(1755)	1034(987)
9(a)	1403(791)	780(667)†	798(677)	1779(1559)
9(b)	3011(1618)	1735(1382)†	1769(1423)	132(126)
10	841(341)	155(136)†	169(140)	1505(1262)
11	256(181)†	532(391)	375(347)	162(156)
TOTALS	23864(142887)	23003(14908)	22417(13461)	27494(20492)
SCORES	9	7	5	0

Table 1: Comparison of **E1, E2** and **E3**

5. Summary and Conclusions

A new family of models (using a scaling function) for the interpolating curve used in constructing multi-step quasi-Newton methods has been introduced. The methods have a free parameter and it has been shown how this parameter may be determined by using values of the objective function to produce numerical estimates of the derivative of the function at the latest three iterates. From a particular choice of the scaling function, three new algorithms based on this general approach have been derived. All three algorithms show a significant improvement, in numerical terms, over the standard (single-step) BFGS method, and the best of the three new methods (namely, **E2**) also yielded gains when compared with an earlier successful multi-step method also based on function-values (**A1F**).

References

[1] R. Brent, *Algorithms for Minimization without Derivatives*, Prentice-Hall, Englewood Cliffs, New Jersey (1973).

[2] C.G. Broyden, The convergence of a class of double-rank minimization algorithms - Part 2: The new algorithm, *J. Inst. Math. Applic.*, **6** (1970), 222-231.

[3] J.E. Dennis, R.B. Schnabel, Least change secant updates for quasi-Newton methods, *SIAM Review*, **21** (1979), 443-459.

[4] R. Fletcher, A new approach to variable metric algorithms, *Comput. J.*, **13** (1970), 317-322.

[5] R. Fletcher, *Practical Methods of Optimization*, Second Edition, Wiley, New York (1987).

[6] J.A. Ford, R.A. Ghandhari, On the use of function-values in unconstrained optimisation, *J. Comput. Appl. Math.*, **28** (1989), 187-198.

[7] J.A. Ford, R.A. Ghandhari, Efficient utilisation of function-values in quasi-Newton methods, *Colloquia Mathematica Societatis János Bolyai*, **59** (1990), 49-64.

[8] J.A. Ford, R.A. Ghandhari, On the use of curvature estimates in quasi-Newton methods, *J. Comput. Appl. Math.*, **35** (1991), 185-196.

[9] J.A. Ford, I.A. Moghrabi, Alternative parameter choices for multi-step quasi-Newton methods, *Optimization Methods and Software*, **2** (1993), 357-370.

[10] J.A. Ford, I.A. Moghrabi, Multi-step quasi-Newton methods for optimization, *J. Comput. Appl. Math.*, **50** (1994), 305-323.

[11] J.A. Ford, I.A. Moghrabi, Using function-values in multi-step quasi-Newton methods, *J. Comput. Appl. Math.*, **66** (1996), 201-211.

[12] J.A. Ford, A.F. Saadallah, A rational function model for unconstrained optimization, *Colloquia Mathematica Societatis János Bolyai*, **50** (1986), 539-563.

[13] D. Goldfarb, A family of variable metric methods derived by variational means, *Math. Comp.*, **24** (1970), 23-26.

[14] D.F. Shanno, Conditioning of quasi-Newton methods for function minimization, *Math. Comp.*, **24** (1970), 647-656.

[15] D.F. Shanno, K.H. Phua, Matrix conditioning and nonlinear optimization, *Math. Programming*, **14** (1978), 149-160.