J. KSIAM Vol.15, No.4, 267–275, 2011

A SELF SCALING MULTI-STEP RANK ONE PATTERN SEARCH ALGORITHM

ISSAM A.R. MOGHRABI

DEPARTMENT OF M.I.S., SCHOOL OF BUSINESS, GULF UNIVERSITY FOR SCIENCE AND TECHNOLOGY, KUWAIT

E-mail address: moughrabi.i@gust.edu.kw

ABSTRACT. This paper proposes a new quickly convergent pattern search quasi-Newton algorithm that employs the multi-step version of the Symmetric Rank One (SRI). The new algorithm works on the factorizations of the inverse Hessian approximations to make available a sequence of convergent positive bases required by the pattern search process. The algorithm, in principle, resembles that developed in [1] with multi-step methods dominating the derivation and with numerical improvements incurred, as shown by the numerical results presented herein.

1. INTRODUCTION

We will consider two-step quasi-Newton methods (in contrast to the standard, more commonly-used one-step methods) for the unconstrained optimization problem

$$min\ f(x), \qquad \text{where } x \in R^n.$$

Denoting, the gradient and Hessian of f by g and G, respectively,. we note that such methods resemble standard (one-step) quasi-Newton methods, except that the approximation H_{i+1} to the inverse of the Hessian $G(x_{i+1})$ is now required to satisfy a condition of the following form:

$$s_i - \gamma_i s_{i-1} = H_{i+1}(y_i - \gamma_i y_{i-1}), \tag{1.1}$$

or

$$r_i = H_{i+1} w_i, \tag{1.2}$$

say, instead of the standard condition

$$s_i = H_{i+1} y_i, \tag{1.3}$$

commonly known as the Secant Equation. In (1) and (3), s_i and y_i are defined by

$$s_i = x_{i+1} - x_i; \tag{1.4}$$

Received by the editors October 1, 2010; Revised October 5, 2011; Accepted in revised form November 18, 2011.

2000 *Mathematics Subject Classification.* 65K10.

Key words and phrases. quasi-Newton methods, Pattern Search algorithms, multi-step methods.

$$y_i = g(x_{i+1}) - g(x_i), \tag{1.5}$$

where $\{x_i\}$ are the iterates produced by the method under consideration. The derivation of (1.1) is described by Ford and Moghrabi [2]. Quadratic curves $x(\tau)$ and $u(\tau)$ (where $\tau \in R$) are constructed to interpolate, respectively, the three most recent iterates x_{i-1}, x_i and x_{i+1}, and the three associated gradient evaluations (which are assumed to be available). The derivatives of these two curves at $\tau = \tau_2$, where τ_j is the value of τ for which

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

are then substituted into the relation (derived from applying the Chain Rule to $g(x(\tau))$:)

$$G(x_{i+1})x'(\tau_2) = g'(x(\tau_2)), \tag{1.6}$$

where primes denote differentiation with respect to τ . It is important, at this point, to note particularly that

$$w_i \stackrel{def}{=} u\prime(\tau_2) = \sum_{j=0}^{m-1} \underline{s}_{i-j}\{ \sum_{k=m-j}^{m} \mathcal{L}'_k(\tau_m)\}; \tag{1.7}$$

is, in general, only an approximation to the term $g'(x(\tau_2))$ that is required in (1.6), whereas

$$r_i \stackrel{def}{=} x'(\tau_2) = \sum_{j=0}^{m-1} \underline{y}_{i-j}\{ \sum_{k=m-j}^{m} \mathcal{L}'_k(\tau_m)\} \tag{1.8}$$

may be computed exactly. The quantity $\mathcal{L}_j(\tau)$ 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 $X = \{\underline{x}(\tau)\}$, where in this work we have chosen $m = 2$.

On making these substitutions into (1.6) and removing a common scaling factor. we obtain a relation of the form (1.1) for $H_{i+1} \approx G^{-1}(x_{i+1})$ to satisfy. H_{i+1} may then be obtained (for example) by use of an appropriately modified version of the SRI formula :

$$H_{i+1} = H_i + \frac{(r_i - H_i w_i)(r_i - H_i w_i)^T}{(r_i - H_i w_i)^T r_i} \tag{1.9}$$

$$\stackrel{def}{=} SRI(H_i, r_i, w_i). \tag{1.10}$$

The term γ_i in (1.1) is an expression that depends on the choice of the three values τ_0, τ_1 and τ_2 and hence it is necessary to choose these three values with some care, since the updating of the Hessian inverse approximation (and, therefore, the numerical performance of such an algorithm) is determined by the value of γ_i in (1.1). Some successful choices for $\{\tau_k\}_{k=0}^2$ were described by Ford and Moghrabi [2].

For the above algorithms, the new iterate is determined by an inexact line search along

$$p_i = -H_i g_i \tag{1.11}$$

as

$$x_{i+1} = x_i + \alpha_i p_i$$

where α_i is chosen such that

$$f_{i+1} - f_i \leq \beta \alpha_i g_i^T H_i g_i$$

and

$$|g_{i+1}^T H_i g_i| \leq \lambda g_i^T H_i g_i$$

hold. The parameters λ and β satisfy $\lambda \in (\lambda, 1)$ and $\beta \in (0, 1/2)$.

The SRI possesses finite termination on quadratic functions without requiring exact line searches and regardless of what the vectors r_i and w_i are. However, it suffers from numerical instability as the matrix may become singular or undefined. Recent studies [1] have shown that, if the SRI is stabilized somehow then it could outperform well with the standard BFGS method.

One particularly successful attempt to guard the updated matrix from failing to be positive definite, at a given iteration, is to use a self-scaling SRI (see [3]) as follows:

$$H_{i+1} = \rho H_i + \frac{(s_i - \rho H_i y_i)(s_i - \rho H_i y_i)^T}{(s_i - \rho H_i y_i)^T s_i}, \tag{1.12}$$

with a similar structure for the Multi-step formula (1.9)

$$H_{i+1} = \widehat{\rho} H_i + \frac{(r_i - \widehat{\rho} H_i w_i)(r_i - \widehat{\rho} H_i w_i)^T}{(r_i - \widehat{\rho} H_i w_i)^T r_i}. \tag{1.13}$$

For (1.12), Osborne and Sun [4] propose a new algorithm (OCSSR1) that exploits Davidon's optimal conditioning with two possible values for the scaling parameter ρ as

$$\rho = \theta/\beta \pm (\theta^2/\beta^2 - \theta/\lambda)^{1/2}$$

where $\theta = s_i^T H_i^{-1} s_i$, $\beta = s_i^T y_i$, and $\lambda = y_i^T H_i y_i$.Their numerical tests show that their algorithm compares well with the standard BFGS.

Using similar reasoning, it is straightforward to show that, by analogy,

$$\widehat{\rho} = \widehat{\theta}/\widehat{\beta} \pm (\widehat{\theta}^2/\beta^2 - \theta/\lambda)^{1/2}$$

The focus in this paper is on problems for which the gradient is not available for which we consider multi-step direct search methods This is true in quite few engineering problems as well as industry, economics and a variety of domains. In such situations, classic implementation of quasi-Newton methods does not work well and other methods such as direct search

methods display better numerical convergence. The next Section introduces the direct search template. In Section 3, the new method is derived and then numerical results are presented and discussed.

2. DIRECT SEARCH METHODS

Direct Search is a direct search strategy, developed first Box in the 50s [5] and Hooke-Jeeves [6] in the early 60s. Such methods rely only on available function evaluations. They start from a base point and explores various step sizes in every possible direction, evaluating the objective function for each move. The aim is to yield a lower function value and once such a value is found, the new iterate is defined and hence it becomes the new base point for the next iteration. If such a trial point is not successful, other trial points will be tested. Direct Search methods have received much attention, especially the Multidirectional Search Patterns of Dennis and Torczon [7] and the Generalized Pattern Search Methods of Torczon [8]. In specific, Torczon [8] defines a pattern P_i and a positive real number Δ_i. The pattern P_i consists a nonsingular basis matrix $A \in \mathbb{R}^{nxn}$ and a generating matrix $Z_i \in \mathbb{Z}^{nxp}$ $(p > 2n)$ such that Z_i can be partitioned as

$$Z_i = [N_i \quad -N_i \quad D_i]$$

where $N_i \in N \subset \mathbb{Z}^{nxn}$, N is a finite set of non-singular matrices, $D_i \in \mathbb{Z}^{nx(p-2n)}$ with at least one zero column.

Thus,

$$P_i = AZ_i = [AN_i \quad -AN_i \quad AD_i],$$

where the columns of P_i can span $\mathbb{R}^n$.

Therefore, if $Z_i = [z_i^{(1)}, z_i^{(2)}, ..., z_i^{(p)}]$ then a trial step at some iteration is given by $s_i^{(k)} = \Delta_i A z_i^{(k)}$, so that $x_i^{(k)} = x_i + s_i^{(k)}$, where k indicates the kth trial. The scalar Δ_i serves as a step length.

The algorithm framework is outlined as:
Given the base point $x_0 \in \mathbb{R}^n$, $A \in \mathbb{R}^{nxn}$, $Z_0 \in \mathbb{Z}^{nxp}$, $\Delta_i > 0$, $i = 0$

while not converged do

1. find step s_i using the space (Δ_i, AZ_i);
2. if $f(x_i + s_i) < f(x_i)$, then $x_{i+1} = x_i + s_i$ else $x_{i+1} = x_i$;
3. update Δ_i to Δ_{i+1} and Z_i to Z_{i+1}, $i = i + 1$.

Various variations to the generalized pattern search methods (GPSM) have been devised that differ in the degree of freedom in choosing the matrices and the scalings. For example, the methods of Coope and Price [9] replaced the matrix A with a positive bases and only updated it at some iterations. They also allowed the integer matrix Z to vary from iteration to iteration as they also tried points that are not restricted to the directions of the positive bases. They

also permitted irrational mesh size, as apposed to only rational size in GPSM. All the above methods proved effective and while one particular method behaves well on some problems, others are more competitive on other problems.

3. MULTI-STEP RANK ONE PATTERN SEARCH ALGORITHM

The derivation will proceed in a fashion similar to [1]. Formula (1.13) can be expressed in its product form as follows:

$$H_{i+1} = (I + z_i v_i^T)\widehat{\rho} H_i (I + z_i v_i^T)^T \tag{3.1}$$

where

$$\begin{aligned} z_i &= \mu(r_i - \widehat{\rho} H_i w_i) \\ v_i &= \frac{B_i r_i - \widehat{\rho} w_i}{\widehat{\rho}} \\ \mu &= \frac{-\widehat{\rho} \pm \sqrt{(\theta' \widehat{\rho} - \widehat{\rho}^2 \beta')/\beta' - \lambda' \widehat{\rho})}}{\theta' - 2\beta' + \lambda' \widehat{\rho}^2} \end{aligned}$$

for $\theta' = r_i^T H_i^{-1} r_i$, $\beta' = r_i^T w_i$, and $\lambda' = w_i^T H_i w_i$.
If

$$H_i = L_i L_i^T,$$

where L_i is an nxn nonsingular matrix, then (1.13) can be expressed as

$$L_{i+1} = \widehat{\rho}^{1/2}[I + z_i v_i^T] L_i. \tag{3.2}$$

By analogy with what Osborne and Sun [4] for their methods, It is straightforward to show that if $\rho > 0$ and $r_i^T w_i > 0$, then H_{i+1} is positive definite if an only if

$$\widehat{\rho} \notin \left[\frac{r_i^T w_i}{w_i^T H_i w_i}, \frac{r_i^T B_i r_i}{r_i^T w_i} \right].$$

The quasi-Newton direction is given as

$$p_i = -L_i \widehat{g}_i,$$

where, if $L_i = [l_i^{(1)}, l_i^{(2)}, ..., l_i^{(n)}]$, then $p_i = -l_i^T \widehat{g}_i$ are the directional derivatives of $f(x)$ in the directions $l_i^{(j)}$, $j = 1, 2, ..., n$. The components of the vector $\widehat{g}_i$ can be calculated using standard finite difference methods.

Let

$$u_i = L_i^T v_i,$$

then (see (3.1))

$$z_i = \widehat{\rho}\mu L_i^T u_i$$

and

$$L_{i+1} = \widehat{\rho}^{1/2}[I + \widehat{\rho}\mu u_i u_i^T]L_i.$$

However, u_i can be expressed as

$$u_i = -L_i^T[y_i + (\alpha_i/\widehat{\rho})g_i]$$

As with the method developed in [1], the key point of the implementation is how to choose $\widehat{\rho}$. It should be chosen such that $\kappa(L_{i+1}L_{i+1}^T)$ is minimal, where $\kappa(.)$ indicates the condition number of some matrix. Given that $L_0 = I$, the condition number can then be estimated using the recurrence

$$Tr(L_{i+1}L_{i+1}^T) = \widehat{\rho}\left[Tr(L_iL_i^T) + \frac{(L_iu_i)^TL_iu_i}{u_i^TL_iw_i}\right]. \tag{3.3}$$

4. THE ALGORITHM OUTLINE

The initial search direction set Φ is initially the positive basis generated from the unit matrix. This positive basis is updated at each iteration to generate subsequent positive bases used in generating the search directions set Φ. Inexact line search can be carried out along the generated quasi-Newton search direction in order to settle on the next iterate, named as pattern search point. If such a point is not found at a given iteration, the current point is used to serve as the pattern search point.

Algorithm MSSR1P

1. $i = 0, m = 0$, $L_0 = I$. Define x_0 and build a positive basis V^+ from L_0 and define $V_0^+ = [V^+, -V^+e]$. Let $F^m = \xi$ (some positive scalar)
2. while not converged do
   - 2a. set grid mesh size h_i; $k = 1$; $num = 0$ (record of the number of failed searches); $noc_i = |V_i^+|$ (number of columns in the matrix V_i^+).
   - 2b. while $(num < noc_i)$ do
     - i. evaluate $f(x)$ on a chosen number of points ς on the grid $\Sigma^{(m)}$ (generated from x_i, grid size h_i and a positive basis V_i^+) including the point $x_i + h_iv_i^+$
     - ii. if for a point in the grid $x_i^{(t)}$ for $t = 1..\varsigma$, $f(x_i^{(t)}) < f(x_i) - h_i^2$, $x_{i+1} = x_i^{(t)}$; $h_{i+1} = \phi h_i$, where ϕ is some positive integer

iii. If $h_{i+1} > F^m$, set $h_{i+1} = h_i$, $V_{i+1}^+ = V_i^+$, $i = i+1, num = 0$
else $num+ = 1$
iv. $k = k+1$
v. if $k > noc_i$ set $k = 1$

3. Let $\widetilde{x}^m = x_i$ and $\widetilde{h}^m = h_i$
4. Build a directional derivatives $\widehat{g}^{(m)}$ at using $\widetilde{x}^m$in every direction in L_i using some finite difference method
5. $p^{(m)} = -L^{(m)}\,\widehat{g}^{(m)}$
6. find a new iterate x_{i+1} using an inexact line search such that is 1.11 satisfied for $\beta = 10^{-4}$
7. if $||\widehat{g}^{(m+1)} - \widehat{g}^{(m)}|| \geq \beta^2$ go to 8 else stop
8. if $\widehat{g}^{(m)T}(\,\widehat{g}^{(m+1)} - \widehat{g}^{(m)})$ is too small set $L^{(m+1)} = L^{(m)}$, $V_{m+1}^+ = V_m^+$; Go to 11
9. calculate $\widehat{\rho}$
10. Update $L^{(m)}$ to build $L^{(m+1)}$ using (3.1) (1.13).
11. Build $V_{i+1}^+ = [L_{m+1}, -L_{m+1}e]$; set $F^{m+1} = \beta F^m$, $k+ = 1$, $m+ = 1$

The remarks made here are consistent with those in [1]. The central difference formula is more accurate to use in the implementation of the above algorithm. However, more function evaluations would be needed in that case (see [1].for more details) Also, adjusting the mesh size at each iteration for the above algorithm is much simpler than in other algorithms as it is self-adjusting. The size of h is scaled up when an acceptable pattern point is found or scaled down when a new iterate is located. Such an adjustment controls the efficiency of the algorithm as it avoids too large or too small mesh sizes. The algorithm is actually halted as soon as the mesh size gets too small.

5. NUMERICAL RESULTS AND CONCLUSIONS

A pool of test functions of a variety of moderate dimensions was used for testing the algorithm. The algorithm is compared to that developed in [1] in terms of iterations and function evaluations. The main observation here is that **MSSR1P** outperforms **SRIP** as the dimension of the problem increases while it loses on small problems as convergence gets slower. The improvement incurred on larger problems is visible as it exceeds 8% on quite few problems.

In this paper a new quasi-Newton Pattern Search algorithm is developed. The algorithm is similar in spirit to that in [1] as it uses the same idea of generating search spaces using some positive basis matrix generated by utilizing the symmetric rank one update. The new method utilizes a more general form of the SRI that uses past step vectors and corresponding gradient difference vectors in order to exploit more accumulated information generated from past iterations. The methods do not require any gradient or Hessian availability but rather use difference formulae to estimate the gradient. The results obtained are encouraging especially as the size of the problem grows. The convergence analysis of the methods is quite similar to that done in [1].

TABLE 1. The test function test and respective dimensions

Function Name (dimension)
1. Watson function $(3 \leq n \leq 31)$
2. Extended Rosenbrock $(2 \leq n \leq 40, n$ even$)$
3. Extended Powell $(2 \leq n \leq 40, n$ divisible by $4)$
4. Penalty function I $(2 \leq n \leq 40)$
5. Variably dimensioned function $(2 \leq n \leq 20)$
6. Trigonometric function $(2 \leq n \leq 60)$
7. Modified Trigonometric function $(2 \leq n \leq 60)$
8. Broyden Tridiagonal function $(2 \leq n \leq 40)$
9. Discrete Boundary value function $(2 \leq n \leq 40)$
10. Oren and Spedicato Power function $(2 \leq n \leq 40)$
11. Full Set of Distinct Eigen Values Problem $(2 \leq n \leq 20)$
12. Tridiagonal function $(2 \leq n \leq 20)$
13. Wolfe function $(2 \leq n \leq 40)$
14. Generalized Shallow function $(2 \leq n \leq 80, n$ even$)$
15. Rosenbrock $(n = 2)$
16. Quadratic function $(n = 2)$
17. Freudenstein & Roth$(n = 2)$
18. Box three-dimensional $(n = 3)$
19. Powell Singular $(n = 4)$
20. Biggs EXP6 $(n = 6)$

REFERENCES

[1] T. Wu, L.P. Sun, A new quasi-Newton pattern search method based on symmetric rank-one update for unconstrained optimization, Computers and Math with applications 55 (2008) 1201-1214.

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

[3] L.P. Sun, A Self-scaling symmetric rank one updating algorithm for unconstrained optimization, Journal of Chinese Universities (English Series) 3 (1984) 15-25.

[4] M.R Osborne, L.P. Sun, A new approach to symmetric rank-one updating, IMA Journal of Numerical Analysis 19 (1999) 497-507.

[5] G.E.P. Box, Evolutionary operation: A method for increasing industrial productivity, Applied Statistics 6 (1957) 81-101.

[6] R. Hooke, T.A. Jeeves, Direct Search Solution of numerical and statistical problems, Journal of the Association for Computing Machinery (ACM) 8 (1961) 212-219.

[7] J.E. Dennis, V. Torczon, Direct Serach methods on parallel machines, SIAM Journal on Optimization 1 (1991) 448-474.

[8] V. Torczon, On the convergence of Pattern Search algorithms, SIAM Journal on Optimization 7 (1997) 1-25.

[9] I.D. Coope, C.J. Price, On the convergence of grid-based methods for unconstrained minimization, SIAM Journal on Optimization 11 (2001) 859-869.

TABLE 2. Comparison of iteration and function evaluation count for both algorithms

	MSSR1P		**SRIP**	
Problem	**fcount**	**icount**	**fcount**	**icount**
1	1983	933	1898	992
2	1717	693	1968	733
3	1963	501	2184	583
4	1781	443	1880	528
5	1877	599	1856	581
6	3018	1568	3198	1597
7	3999	1487	5001	1548
8	1142	579	998	476
9	1766	509	1886	623
10	1811	836	1932	978
11	831	299	870	340
12	951	342	990	394
13	1689	829	1753	901
14	8932	2056	9999	2468
15	259	87	251	79
16	251	81	249	79
17	288	86	281	81
18	240	60	297	62
19	176	68	166	64
20	187	68	181	58
Totals	34861	12124	37838	13165