American Journal of Applied Mathematics
2015; 3(2): 47-50
Published online February 25,2015 (http://www.sciencepublishinggroup.com/j/ajam)
doi: 10.11648/j.ajam.20150302.13
ISSN: 2330-0043 (Print); ISSN: 2330-006X (Online)

On Quasi-Newton Method for Solving Unconstrained Optimization Problems

Felix Makanjuola Aderibigbe, Kayode James Adebayo, Adejoke O. Dele-Rotimi

Department of Mathematical Sciences, Ekiti State University, Ado Ekiti, Nigeria

Email address:
fm_aderibigbe@yahoo.co.uk (F. M. Aderibigbe), akj7070@yahoo.com (K. J. Adebayo), delerotimi@ymail.com (A. O. Dele-Rotimi)

To cite this article:
Felix Makanjuola Aderibigbe, Kayode James Adebayo, Adejoke O. Dele-Rotimi. On Quasi-Newton Method for Solving Unconstrained Optimization Problems. *American Journal of Applied Mathematics*. Vol. 3, No. 2, 2015, pp. 47-50. doi: 10.11648/j.ajam.20150302.13

Abstract: This paper discusses the use of quasi-Newton method algorithm employed in solving unconstrained optimization problems. The method is aimed at circumventing the computational rigours undergone using the Newton's method. The Quasi –Newton method algorithm was tested on some benced mark problems with the results compared with the Conjugate Gradient Method. The results gotten using the Quasi-Newton Method compared favourably with results of existing CGM algorithm.

Keywords: Optimization Problems, Conjugate Gradient Method, Control Operator

1. Introduction

The general optimization problem to be considered is of the form by [10] and [11] as

$$\text{Optimize: f(x)} \qquad \text{(objective function)} \quad (1.1)$$

$$\text{Subject to: } h_i(x) = 0,\ i = 1, \dots, m_1 \qquad \text{(equality constraint)} \quad (1.2)$$

$$g_j(x) \geq 0,\ j = 1, \dots, m_2 \qquad \text{(inequality constraint)} \quad (1.3)$$

$$x \geq 0 \qquad \text{(non negativity conditions)} \quad (1.4)$$

where $\boldsymbol{x}$ is a vector of n variables $(x_1, x_2, \dots, x_n)$, $h(x)$ is a vector of equations of dimension m_1, and $g(x)$ is a vector of inequalities of dimension m_2, and the total number of constraints $m = (m_1 + m_2)$. If $h_i(x) = g_j(x) = 0$ then the problem becomes an unconstrained optimization problem otherwise, it is a constrained problem. Methods for solving this model have been developed, tested and successfully applied to many important problems of scientific and economic interest. However, in spite of the proliferation of the methods, there is no universal method for solving all optimization problems which calls for application of quasi-Newton method in solving (1.1) through (1.4).

2. The Conjugate Gradient Method Algorithm

The gradient method uses the iterative scheme

$$x_{i+1} = x_i - \alpha \nabla F(x) \qquad (2.1)$$

to generate a sequence $\{x_i\}_{i=1}^{n}$ of vectors which converges to the minimum of $F(x)$. The parameter α appearing in (2.1) denotes the step length of the descent sequence. In particular, if F is a functional on a Hilbert space $\mathcal{H}$ such that in $\mathcal{H}$, F admits a taylor series expansion

$$F(x) = F_0 + \langle a, x \rangle_H + \langle x, Ax \rangle_H \qquad (2.2)$$

where A is an $n \times n$ symmetric, positive definite operator and H is a Hilbert space, $a, x \in H$.

The Conjugate Gradient Method (CGM) described by Hestenes and Stiefel [5,pp 409-436] is an iterative method that can solve (2.2) by simply generating a sequence $\{x_n\}$ of approximation of the solution x from an arbitrary initial approximation x_0. The guessed x_0 is sequentially improved upon untill a desired accuracy is attained. The CGM algorithm for iteratively locating the minimum x^* of $F(x)$ in H is described in [9] as follows:

a: Guess the first element $x_0 \in \mathcal{H}$

b. Compute $p_0 = -g_0$ (2.3a)

c. Set $x_{i+1} = x_i + \alpha_i p_i$; $where\ \alpha_i = \frac{\langle g_i, g_i \rangle_{\mathcal{H}}}{\langle p_i, Ap_i \rangle_{\mathcal{H}}}$ (2.3b)

d. Compute $g_{i+1} = g_i + \alpha_i A p_i$ (2.3c)

e. Set $p_{i+1} = -g_{i+1} + \beta_i p_i$ $where\ \beta_i = \frac{\langle g_{i+1}, g_{i+1} \rangle_{\mathcal{H}}}{\langle g_i, g_i \rangle_{\mathcal{H}}}$ (2.3d)

f. if $g_i = 0$ for some i, then, terminate the sequence: else set i

= i +1 and go over to step b.

In the iterative steps a – f above, p_i denotes the descent direction at the *ith* step of the algorithm, α_i denotes the step length of the descent sequence $\{x_i\}$ and g_i denotes the gradient of *f* at x_i. Steps *c, d* and *e* of the algorithm reveal the crucial role of the linear operator A in determining the step length of the descent sequence and also in generating a conjugate direction of search. Applicability of the algorithm thus rests heavily on explicit knowledge of A.

3. Quasi Newton Method

Nonlinear problems in finite dimensions are generally solved by iteration. According to [2], for the minimization problem, and [1], for systems of equations, introduced new methods which although iterative in nature, were quite unlike any others in use at the time. These papers together with very important modification and classification of Davidon's work by [4] have sparked a large amount of research in the late sixties and early seventies.

This work has led to a new class of algorithm which has been called by the names quasi-Newton or modification methods. The methods have proved themselves in dealing with systems of *n* equations in *n* unknowns, and the unconstrained minimization of functionals. [3]

The basic idea behind any quasi-Newton method is to eliminate computation of the Hessian in every iteration and the methods are based on Newton's method to find the stationary point of a function, where the gradient is zero or near zero. The Hessian is updated by analyzing successive gradient vectors instead. Detailed overviews of quasi-Newton methods are presented in [8] and [7]

The BFGS method named after Broyden, Fletcher, Goldfarb and Shanno who discovered it in 1970. It is numerically stable and has a very effective "self-correcting properties" accounts for its superior performance in practice [8]. If the matrix H_k incorrectly estimates the curvature in the objective function, and if this bad estimate slows down the iteration, then the Hessian approximation will tend to correct itself within a few steps.

Since the search direction $p = -H_k \nabla f(x_k)$, this has the advantage that we don't need to solve a linear system to get the search direction, but only do a matrix/vector multiply.

The BFGS update formula is as follows:

$$H_{k+1} = H_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} \quad 3.1$$

By taking the inverse, the BFGS update formula for $B_{k+1}(i.e\ H_{k+1}^{-1})$ is obtained:

$$H_{k+1}^{-1} = B_{k+1} = B_k + \left(\frac{1+y_k^T B_k y_k}{y_k^T s_k}\right)\frac{s_k s_k^T}{s_k^T y_k} - \frac{s_k y_k^T B_k + B_k y_k s_k^T}{y_k^T s_k} \quad 3.2$$

The BFGS, preserve positive definiteness of the Hessian approximations if and only if $y_k^T s_k > 0$.(see theorem 1 for the proof)

Theorem 1. Let B_i be a symmetric positive-definite matrix, and assume that B_{i+1} is obtained from B_i using the BFGS update formula. Then B_{i+1} is positive definite if and only if $y_i^T s_i > 0$.

Proof. If B_i is positive definite, then it can be factored as $B_i = LL^T$ where L is a nonsingular matrix (Cholesky factorization of B_i). If this factorization is substituted into the BFGS formula for B_{i+1}, then

$$B_{i+1} = LWL^T \quad 3.3$$

where $B_{i+0} = I - \frac{\hat{s}\hat{s}^T}{\hat{s}^T\hat{s}} + \frac{\hat{y}\hat{y}^T}{\hat{y}^T\hat{y}}$, $\hat{s} = L^T s_i$, and $\hat{y} = L^{-1} y_i$

B_{i+1} will be positive definite if and only if W is. To test if W is positive definite, we test if $v^T W v > 0$ for all $v \neq 0$. let θ_1 be the angle between v and $\hat{s}$, θ_2 the angle between v and $\hat{y}$, and θ_3 is the angle between $\hat{s}$ and $\hat{y}$. Then

$$v^T W v = v^T v - \frac{(v^T\hat{s})^2}{\hat{s}^T\hat{s}} + \frac{(v^T\hat{y})^2}{\hat{y}^T\hat{s}}$$

$$= \|v\|^2 - \frac{\|v\|^2\|\hat{s}\|^2 cos^2\theta_1}{\|\hat{s}\|^2} - \frac{\|v\|^2\|\hat{y}\|^2 cos^2\theta_2}{\|\hat{y}\|\,.\,\|\hat{s}\| cos\theta_3}$$

$$= \|v\|^2\left[1 - cos^2\theta_1 + \frac{\|\hat{y}\| cos^2\theta_2}{\|\hat{s}\| cos\theta_3}\right]$$

$$= \|v\|^2\left[sin^2\theta_1 + \frac{\|\hat{y}\| cos^2\theta_2}{\|\hat{s}\| cos\theta_3}\right] \quad 3.4$$

If $y_i^T s_i > 0$, then $\hat{y}^T\hat{s} > 0$ and $cos\theta_3 > 0$; hence $v^T W v > 0$ and W is positive definite. If $y_i^T s_i < 0$, then $cos\theta_3 < 0$; in this case, v can be chosen so that $v^T W v < 0$ and so W is not positive definite. This completes the proof.

4. Quasi-Newton Method Algorithm

Given a quadratic functional

$$f(x) = f_0 + \langle a, x\rangle + ½\langle x, Ax\rangle$$

for x, a in Hilbert space with A being a positive, symmetric linear operator. The quasi- Newton algorithm is described in the following steps:

Step 1: Guess the initial element, x_0

Step 2: Compute the gradient, g_0

Step 3: Compute the descent direction, $p_0 = -B_0 g_0$; $B_0 = I$; $p_i = -B_i g_i$

Step 4: Compute the step length, $\alpha_i = \frac{g_i^T g_i}{p_i^T A p_i}$

Step 5: Update the descent sequence, $x_{i+1} = x_i + \alpha_i p_i$

Step 6: Update the gradient $g_{i+1} = \nabla f(x_{i+1})$

Step 7: Test for convergence $f(x_i)\ and\ \|g_1\|$

Step 8: Determine the vector updates

$$s_i = x_{i+1} - x_i$$

$$y_i = \nabla f(x_{i+1}) - \nabla f(x_i)$$

Step 9: Compute the new Hessian approximate

$$B_{i+1} = B_i + \left(\frac{1+y_i^T B_i y_i}{y_i^T s_i}\right)\frac{s_i s_i^T}{s_i^T y_i} - \frac{s_i y_i^T B_i + B_i y_i s_i^T}{y_i^T s_i}$$

Step 10: Compute the next descent direction

$$p_{i+1} = -B_{i+1}\nabla f(x_{i+1})$$

Step 11: Return to step 3

In the iterative steps 2 to 10 above, p_i denotes the descent direction at the i^{th} step of the algorithm, α_i denotes the step length of the descent sequence $\{x_i\}$, I denotes the identity $(n \times n)$ matrix, f denotes the objective function, A is the linear operator, s_i is the difference between two consecutive variable values, y_i is the difference between two consecutive gradient values and g_i denotes the gradient of f at x_i.

5. Computational Results

The following problems were investigated using both the CGM and QNM algorithms.

(P1) Quadratic example, n = 2

$$\min_{(x)} f(x) = 4(x_1 - 5)^2 + (x_2 - 6)^2, at\ x_0$$

$$= x_2^2 + 4x_1^2 - 12x_2 - 40x_1 + 136, at\ x_0 = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$$

(P2) Quadratic example, n = 2

$$\min_{(x)} f(x) = (x_1 - 3)^2 + 9(x_2 - 5)^2, at\ x_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$= 9x_2^2 + x_1^2 - 90x_2 - 6x_1 + 234, at\ x_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

(P3) Rosenbrock's Banana Shaped Valley function, n = 2

$$\min_{(x)} f(x) = (1 - \mathrm{x}_1)^2 + 100(\mathrm{x}_2 - \mathrm{x}_1^2)^2, \text{at } \mathrm{x}_0 = \begin{pmatrix} -1.2 \\ 1 \end{pmatrix}$$

$$= 100x_1^4 - 200x_1^2x_2 + 100x_2^2 + x_1^2 - 2x_1 + 1, at\ x_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

(P4) Quadratic example, n = 3

$$\min_{(x)} f(x) = (x_1 - 2)^2 + (x_2 - 4)^2 + (x_3 - 6)^2, at\ x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$= x_3^2 + x_2^2 + x_1^2 - 12x_3 - 8x_2 - 4x_1 + 56, at\ x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

(P5) Beale's Function, n = 2

$$\min_{(x)} f(x) = [1.5 - x_1(1 - x_2)]^2 + [2.25 - x_1(1 - x_2^2)]^2 + [2.625 - x_1(1 - x_2^3)]^2$$

$$= x_1^2x_2^2 - 2x_1^2x_2 + x_1^2 + 3x_1x_2 - 3x_1 + 2.25,\ \ at\ x_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

$$f(z) = z_1^2 + z_2^2 + z_3^2\ at\ z_0 = (1.5, 2.25, 2.625)$$

(P6) Powell's Quartet function, n = 4

$$\min_{(x)} f(x) = (x_1 - 10x_2)^2 + (x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4\ at\ x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$= x_1^2 + x_2^4 + 5x_3^2 + 5x_4^2 + 100x_2^2 + 16x_3^4 + 10x_1^4 + 10x_4^4 + 20x_1x_2$$

$$= -10x_3x_4 - 8x_2^2x_3 + 24x_2^2x_3^2 - 32x_2x_3^2 - 40x_1^3x_4 + 60x_1^2x_4^2 - 40x_1x_4^3, at\ x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Table 1. *Numerical Comparison of Solutions to P1.*

Test Problem	Iteration	CGM Function Value	CQNM Function Value	CGM Gradient Norm	CQNM Gradient Norm
1	0	85	85	772	772
	1	25.4016	11.4016	125.504222	11.2028668
	2	0.1e-19	0	0.53642542e-28	0.71454274e-14

From the table above, it can be seen that, while CGM converged in the expected 2nd iteration, due to the presence of two variables, the CQNM also converged at the 2nd iteration using the gradient norm of the two methods as the stopping criteria. Considering the gradient norms, though the CQNM converged with 0.7145274e-14, CGM does better with the value 0.53642542e-28 which is smaller than that of CQNM. This shows that though the CQNM does the job, CGM has an edge.

Table 2. *Numerical Comparison of Solutions to P2.*

Test Problem	Iteration	CGM Function Value	CQNM Function Value	CGM Gradient Norm	CQNM Gradient Norm
2	0	148	148	5200	5200
	1	3.15941035	-68.8405896	12.6723007	3.55981751
	2	-2.8421709e-14	0	2.4802298	0.88817842e-15

From table 2, it is observed that the norm of the gradient of CQNM reached the optimum faster than that of the CGM. Here the CQNM has shown a better performance. This could be due to the spherical contour shape of the problem and an indication of the efficiency of the of the CQNM in solving this class of problems. This point to the fact that, the CQNM compares favourably with the CGM.

Table 3. *Numerical Comparison of Solutions to P3.*

Test Problem	Iteration	CGM Function Value	CQNM Function Value	CGM Gradient Norm	CQNM Gradient Norm
3	0	24.2	24.2	7763.63	7763.63
	1	4.74356541	108982.74	19.0212218	63702.0884
	2	7.5721439e-29	1.19506261e21	0.30288575e-25	7.4916118e16

From table 3, we see that the norm of the gradient of the CQNM diverged instead of converging. This proves the fact that a single method cannot solve all optimization problems hence the need for various methods. Here the CGM has shown a better performance.

Table 4. *Numerical Comparison of Solutions to P4.*

Test Problem	Iteration	CGM Function Value	CQNM Function Value	CGM Gradient Norm	CQNM Gradient Norm
4	0	56	56	224	224
	0	0	-12	0	0

From table 4, here, both methods converged at the first iteration with norm of the gradient being zero for both the CGM and CQNM. This shows which implies that the two methods do well in solving this problem.

Table 5. *Numerical Comparison of Solutions to P5.*

Test Problem	Iteration	CGM Function Value	CQNM Function Value	CGM Gradient Norm	CQNM Gradient Norm
5	0	14.203125	14.203125	56.8125	56.8125
	1	0	5.25	0	0

From table 5, here, both methods converged at the first iteration with norm of the gradient being zero for both the CGM and CQNM. This shows which implies that the two methods do well in solving this problem.

Table 6. *Numerical Comparison of Solutions to P6.*

Test Problem	Iteration	CGM Function Value	CQNM Function Value	CGM Gradient Norm	CQNM Gradient Norm
6	0	215	215	6700	6700
	1	41.5455951	121.137296	192.064671	13.9086897
	2	11.2793734	13.2595664	57.093599	1.66251563e13

From table 6, the function values at the second iteration are close but the gradient of the CQNM began to diverge showing that the method has broken down.

6. Conclusion

In this paper, we apply the quasi-Newton method on a number of unconstrained optimization problems as shown in Tables 1 to 6. Considering the results gotten in each case compared with the results when CGM was used, it clearly shows that the Quasi-Newton Method does fine hence, can be used to solve unconstrained optimization problems. With this development, the authors of this paper wish to extend the Quasi-Newton Method to Optimal Control Problems.

References

[1] BROYDEN, C.G. (1965). "A Class of Methods for Solving Nonlinear Simultaneous Equations". Math. Comp., 19,pp. 577-593.

[2] DAVIDON, W.C. (1959)."Variable Metric Method for Minimization", Rep.ANL-5990 Rev, Argonne National Laboratories, Argonne, I11

[3] DENNIS, J.E. JR and MORÉ,J.J. (1977). "Quasi Newton Methods, Motivation and Theory" SIAM Review 19(1), pp 46-89

[4] FLETCHER, R., and POWELL, M.J.D.,(1963), "A rapidly Convergent Descent Method for Minimization." Comput. J.,6, pp 163-168.

[5] HESTENES, M. R., and STIEFEL, E., (1952), "Method of Conjugate Gradients for solving Linear Systems," J. Res. Nat. Bur. Standards 49, pp 409- 436.

[6] IBIEJUGBA, M. A. and ONUMANYI, P., (1984), "A Control Operator and Some of its Applications," Journal of Mathematical Analysis and Applications, Vol. 103, No.1, pp 31-47.

[7] IGOR, G., STEPHEN, G. N. and ARIELA, S., (2009), Linear and Nonlinear Optimization, 2nd edition. George Mason University, Fairfax, Virginia, SIAM, Philadelphia.

[8] Jorge NOCEDAL and Stephen J. WRIGHT. (2006), Numerical Optimization. 2nd edition. Springer-Verlag, New York.

[9] LAWRENCE, Hasdorff, (1976), Gradient Optimization and Nonlinear Control. J. Wiley and Sons, New York.

[10] RAO, S. S., (1978), Optimization Theory and Applications, Wiley and Sons. New York.

[11] THOMAS, F. E., and DAVID, M. H., (2001), Optimization of Chemical Processes, McGraw Hill Comp.