International Journal of Mathematics and
Computer Science, **9**(2014), no. 2, 83–93

Fibonacci Scheme for Quadratic Functionals

**I. Abdullahi, J. O. Omolehin, K. Rauf, P. L. Dickmu,
O. T. Wahab**

Department of Mathematics
University of Ilorin
Ilorin, Nigeria

email: iabdullahi94@gmail.com, jomolehin@gmail.com,
krauf@unilorin.edu.ng, pawahndukum@yahoo.co.uk,
wahab4math@gmail.com

(Received Nov. 27, 2013, Revised June 19, 2014, Accepted June 23, 2014)

Abstract

It has been proved that the conventional conjugate gradient Method Algorithm (CGM) converges in at most n iterations in a well posed quadratic functional problems. This work used Fibonacci Scheme on Gradient Method (GM) to solve quadratic functional problems and the results are favorable and comparable with CGM algorithm.

1 Preliminaries

Optimization can be best described as an act, process, or methodology of making something (as a design, system, or decision) as fully perfect, functional, or effective as possible In particular, it is the mathematical procedures (as finding the maximum of a function) involved in a particular problem as it was used in 1857 [2, 3, 6].

In the field of applied mathematics, the principles and methods are used to solve quantitative problems in disciplines including physics, biology, engineering, and economics. The questions of minimizing or maximizing functions arising in all disciplines and can be solved using the same mathematical

Key words and phrases: Convergence, Perturbation, Gradient Method, Fibonacci Scheme.
AMS (MOS) Subject Classifications: 49J20, 47N10 34M60, 11B39, 46N10, 65K05.
ISSN 1814-0432, 2014, http://ijmcs.future-in-tech.net

optimization tools (maximum and minimum) [7]. In a typical optimization problem, the aim is to find the values of controllable factors determining the behavior of a system (a physical production process, an investment scheme) that minimize waste or maximize productivity. The simplest problems involve functions (systems) of a single variable (input factor) and may be solved with differential calculus. Thus, linear programming was developed to solve optimization problems involving two or more input variables.

1.1 Fibonacci Scheme

The Fibonacci scheme is based on the observation that a distance-2 code, aside from detecting errors, can also correct errors that occur at known positions in the code block; therefore, in a concatenated code, detected errors at one coding level become located errors that can be corrected at the next level up. The Fibonacci method can also be used to find the minimum of a function of one variable even if the function is not continuous. This method makes use of the sequence of Fibonacci numbers. These numbers are defined as $F_0 = F_1 = 1$ $F_n = F_{n-1} + F_{n-2}$, n = 2 , 3 , 4 , . . . which yield the sequence 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ... Knill's numerical simulations indicate that the Fibonacci scheme, too, can tolerate independent stochastic noise with an error rate of about one percent [12]. In this paper, we provide a perturbation term λ, as the reciprocal of the usual Fibonacci number which remove the difficulty involved in constructing an operator as obtained in the conjugate gradient method algorithm. The method ensures optimum solution with the correct initial guess.

1.2 Conjugate Gradient Method

Below is the conjugate gradient algorithm for minimizing

$$F(x) = F_0 + \langle a, x\rangle_H + \frac{1}{2}\langle x, Ax\rangle_H \tag{1.1}$$

Where A is an $n * n$ symmetric, positive definite operator on the Hilbert Space H, and a is a vector in H.

1. The first element x_0 of the descent sequence is guessed, while the remaining members of the sequence are computed with aid of the formulae

2. $p_0 = -g_0 = -(a + Ax_0)$

3. $x_{i+1} = x_i + \alpha_i p_i$ where $\alpha_i = \dfrac{\langle g_i, g_i \rangle}{\langle p_i, Ap_i \rangle}$
   $g_{i+1} = g_i + \alpha_i Ap_i$
   $p_{i+1} = -g_{i+1} + \beta_i p_i$ where $\beta_i = \dfrac{\left\langle g_{(i+1)}, g_{(i+1)} \right\rangle}{\langle g_i, g_i \rangle}$

4. if $g_i = 0$ for some i Set $i = i + 1$ go to step 3

Observer that the convergence rate of CGM algorithm has been proved [2] to be

$$E(x_n) = \left\{ \frac{1 - \frac{m}{M}}{1 + \frac{m}{M}} \right\}^{2n} E(x_0),$$

where m and M are the smallest and greatest values of spectral radius of the operator A respectively.

1.3 Gradient Method

Consider the function $f : \mathbb{R}^n \to \mathbb{R}$ which is continuously differentiable in some domain $D \in \mathbb{R}^n$ and it is assumed that f assumes a local minimum value in D at a point $x \in \mathbb{D}^o$, where $\mathbb{D}^o$ is the interior of D [8, 10, 2].

In order to construct the formula used in what is commonly called the gradient method, we let λ be a positive number and consider the tailor series expansion truncated at the second term

$$f(x - \lambda \frac{\partial f}{\partial x}(x)^{'}) = f(x) + \frac{\partial f}{\partial x}(x)(-\lambda \frac{\partial f}{\partial x}(x)^{'}) + \theta(\lambda)$$

$$= f(x) - \lambda || \frac{\partial f}{\partial x}(x)||_e^2 + \theta(\lambda).$$

If $\frac{\partial f}{\partial x}(x) \neq 0$ then for sufficiently small $\lambda > 0$ we clearly have $f(x - \lambda \frac{\partial f}{\partial x}(x)') < f(x)$

Thus, if we are searching for a minimum of f the point $x - \lambda \frac{\partial f}{\partial x}(x)'$ is an improvement over the point x if $\frac{\partial f}{\partial x}(x) \neq 0$ and λ is positive and in the neighborhood of zero. By repeated construction of new points in this manner we may hope to approach x^*, That is, local minimum of f in $\mathbb{D}$. The gradient consists in the construction of a sequence $\{x\}$ of points in $\mathbb{R}^n$ by the recursion equation

$$x_{i+1} = x_i - \lambda \frac{\partial f}{\partial x}(x_i^{'}), i = 0, 1, 2, \ldots$$

where x_o is the initial guess value [11].

2 Main result

The focal point in CGM algorithm is the control operator A, while our proposed method is focused on the perturbation term λ, in this regards, we observe that there is no need to construct any operator. It is proposed that $\lambda = \frac{1}{F_n}$, where F_n is the usual Fibonacci number. That is, at each iteration n, $\lambda = \frac{1}{F_n}$. With this, one of the effectiveness of the method is that a concatenated error detecting code is used to correct located errors. The classical decoding of a measured code block is performed recursively, starting with the correct initial guess; a divergence is flagged if an error is detected in a subblock. Therefore, optimum solution is attained with Fibonacci scheme algorithm faster than the conjugate gradient method algorithm in solving quadratic functional problems. However, the results of the two methods are favorable and comparable.

In this section, we shall solve many problems with the aid of Fortran compiler [1, 4, 5, 9] and then compare the convergence rate of our method with the result of CGM algorithm.

2.1 Problem 1

Using Conjugate gradient method to Minimize;

$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$, Taking the initial guess $x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$

2.2 Solution 1

Let $f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$, by letting the initial guess $x_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$ $A = \begin{pmatrix} 4 & 2 \\ 2 & 2 \end{pmatrix}$ and $a = \begin{pmatrix} 1 & -1 \end{pmatrix}^T$ Then, the solution obtained is tabulated as:

2.2.1 Table 1: Conjugate Gradient Method for solution 1

No of itera.	x_1	x_2	p_1	p_2	g_1	g_2	alpha	beta	xnum
1	-1.0000	1.0000	-1.0000	-1.0000	1.0000	-1.0000	1.0000	1.0000	2.0000
2	-1.0000	1.5000	0.0000	2.0000	-1.0000	-1.0000	0.25000	0.0000	0.0000

2.3 Problem 2

Using Conjugate gradient method to Minimize;

$f(x_1, x_2, x_3) = \frac{3}{2}x_1^2 + 2x_2^2 + \frac{3}{2}x_3^2 + x_1x_3 + 2x_2x_3 - 3x_1 - x_3$, Taking the initial guess $x_0 = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^T$

$$A = \begin{pmatrix} 3 & 0 & 1 \\ 0 & 4 & 2 \\ 1 & 2 & 3 \end{pmatrix}$$

$$a = \begin{pmatrix} -3 & 0 & -1 \end{pmatrix}^T$$

2.4 Solution 2

Let $f(x_1, x_2, x_3) = \frac{3}{2}x_1^2 + 2x_2^2 + \frac{3}{2}x_3^2 + x_1x_3 + 2x_2x_3 - 3x_1 - x_3$, by letting the initial guess $x_0 = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^T$ $A = \begin{pmatrix} 3 & 0 & 1 \\ 0 & 4 & 2 \\ 1 & 2 & 3 \end{pmatrix}$ and $a = \begin{pmatrix} -3 & 0 & -1 \end{pmatrix}^T$

2.4.1 Table 2: Conjugate Gradient Method for solution 2

No of itera.	x_1	x_2	x_3	p_1	p_2	p_3	g_1	g_2	g_3
1	0.8333	0.0000	0.2778	3.0000	0.0000	1.0000	-3.0000	0.0000	-1.0000
2	0.9346	-0.1215	0.1495	0.4630	-0.5556	-0.5864	-0.2222	0.5556	0.6667
3	1.0000	0.0000	0.0000	0.0795	0.1476	-0.1817	-0.0467	-0.1869	0.1402

No of iteration	alpha	beta	xnum
1	0.277778		
2	0.218692	8.024692E-02	1.44444
3	0.823077	7.074854E-02	0.373832
		1.344473E-13	1.229386E-07

Now, we shall solve the above problems using our Fibonacci scheme $\big(\lambda = \dfrac{1}{F_n}\big)$ to observe the convergence. The gradient method for minimizing $f(x, y)$ is

$$x_{i+1} = x_i - (\frac{1}{F_n})\frac{\partial}{\partial x}(x_i, y_i),$$

$$y_{i+1} = y_i - (\frac{1}{F_n})\frac{\partial f}{\partial y}(x_i, y_i).$$

We shall use the notations $x_1 = x, x_2 = y, x_3 = z$ for our subsequent development

Fibonacci scheme for problem 1.

$$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$$

We compute the partial derivatives of f:

$$\frac{\partial f}{\partial x} = 1 + 4x2y,$$

$$\frac{\partial f}{\partial y} = 2x + 2y - 1.$$

Table 3: Fibonacci Method for $x_0 = 0.3, \quad y_o = 0.2$ **and** $\left(\lambda = \frac{1}{F_n}\right)$

Itera.	x_i	y_i
0	0.30000000000000	0.2000000000000
1	-1.00000001490	1.50000001490
2	-1.00000000497	1.50000000828
3	-1.00000000430	1.50000000669
4	-1.00000000382	1.50000000597
5	-1.00000000357	1.50000000560
6	-1.00000000342	1.50000000539
7	-1.00000000334	1.50000000527
8	-1.00000000328	1.50000000520
9	-1.00000000325	1.50000000516
10	-1.00000000324	1.50000000513
11	-1.00000000322	1.50000000511
12	-1.00000000322	1.50000000510
13	-1.00000000321	1.50000000510
14	-1.00000000321	1.50000000509
15	-1.00000000321	1.50000000509
16	-1.00000000321	1.50000000509
17	-1.00000000321	1.50000000509
⋮	⋮	⋮

Table 4: Fibonacci Method for $x_0 = 0, \quad y_o = 0$ **and** $\big(\lambda = \frac{1}{F_n}\big)$

Itera.	x_i	y_i
0	0.0000000000	0.00000000000
1	-0.500000000000	1.00000000000
2	-0.833333333333	1.22222222222
3	-0.855555555556	1.27555555556
4	-0.871666666667	1.29958333333
5	-0.880320512821	1.31200443787
6	-0.885212266366	1.31897661201
7	-0.888068271030	1.32304082725
8	-0.889773881400	1.32546748376
9	-0.890805785242	1.32693576020
10	-0.891435287877	1.32783158697
20	-0.892432572949	1.32925105864
30	-0.892440616669	1.32926250911
40	-0.892440682065	1.32926260221
50	-0.892440682597	1.32926260296
60	-0.892440682601	1.32926260297
65	-0.892440682601	1.32926260297
70	-0.892440682601	1.32926260297
⋮	⋮	⋮

Fibonacci scheme for Problem 2.

$$f(x_1, x_2, x_3) = \frac{3}{2}x_1^2 + 2x_2^2 + \frac{3}{2}x_3^2 + x_1x_3 + 2x_2x_3 - 3x_1 - x_3.$$

We compute the partial derivatives of f:

$$\frac{\partial f}{\partial x} = 3x + z - 3,$$

$$\frac{\partial f}{\partial y} = 4y + 2z,$$

$$\frac{\partial f}{\partial y} = 3z + x + 2y - 1.$$

Table 5: Fibonacci Method for $x_0 = 0$, $y_o = 0$, $z_0 = 0$ and $\left(\lambda = \frac{1}{F_n}\right)$

Itera.	x_i	y_i	z_i
0	0.0000000000000	0.000000000000	0.000000000000000
1	1.50000000000	0.00000000000	-0.250000000000
2	1.08333333333	0.166666666667	-0.138888888889
3	1.06111111111	8.888888888889E-02	-0.103333333333
4	1.05111111111	7.027777777778E-02	-8.854166666667E-02
5	1.04612713675	6.227564102564E-02	-8.123808349770E-02
6	1.04340602595	5.815057449673E-02	-7.723774656696E-02
7	1.04184778092	5.585272729517E-02	-7.493892291418E-02
8	1.04092770056	5.451576250695E-02	-7.357787667466E-02
9	1.04037483286	5.371905130831E-02	-7.275854302805E-02
10	1.04003895817	5.323739075847E-02	-7.226019657389E-02
20	1.03950901285	5.248093618402E-02	-7.147294538041E-02
30	1.03950474928	5.247486739992E-02	-7.146660679565E-02
40	1.03950471462	5.247481806227E-02	-7.146655526311E-02
50	1.03950471434	5.247481766113E-02	-7.146655484412E-02
60	1.03950471433	5.247481765787E-02	-7.146655484071E-02
65	1.03950471433	5.247481765784E-02	-7.146655484069E-02
70	1.03950471433	5.247481765784E-02	-7.146655484068E-02
75	1.03950471433	5.247481765784E-02	-7.146655484068E-02
80	1.03950471433	5.247481765784E-02	-7.146655484068E-02
⋮	⋮	⋮	⋮

3 Conclusion

Conjugate Gradient Method was used to solve the problems earlier identified and convergence were obtained. For problem 1, the values of x and y are -1.0000 and 1.5000 respectively and for problem 2, the values of x, y and z are 1.0000, 0.0000 and 0.0000 respectively.

We now observed that, using Fibonacci scheme $\lambda = \frac{1}{F_n}, n = 2, 3, \ldots$ for problem 1 with initial guess x = 0.3 and y = 0.2, we obtained -1.00000000321 and 1.50000000509 as the optimum values. For the same problem taking x = 0.0000 and y = 0.0000, we obtained -0.892440682601 and 1.32926260297 which are closely related to the conjugate gradient method result. For problem 2 with initial guess x = 0, y = 0 and z = 0, we obtained 1.03950471433, 0.05247481765784 and -0.07146655484068.

Finally, we are able to show that optimum can be obtained using Fibonacci scheme if the initial guess is chosen correctly. Also, any chosen initial points can still give values very close to optimum. This can be observed from the table of result (Table 4).

References

[1] J. O. Omolehin, I. Abdullahi, K. Rauf, *New Computational Results on some Methods with Fortran in Optimization,* this journal, **8,** no. 2, (2013), 3753

[2] M. A. Ibiejugba, *Computational methods in Optimization*. Ph.D Thesis, University of Leeds, Leeds, U.K., 1985.

[3] S. S. Rao, *Theory and Practice of Optimization, third edition*: A Wiley Inter science Publication John Wiley and Sons, Inc., New York / Chester / Brisbane / Toronto / Singapore, 1996.

[4] T. V. Mourik, *Fortran 90/95 Programming Manual, fifth revision,* Chemistry Department, University College, London, 2005.

[5] W. E. Mayo, M. Cwiakala, *Theory and problems of Programming with Fortran 77*: McGraw-Hill Companies, Inc. U.S.A, 1995.

[6] E. K. Chong, H. Z. Stainslaw, *An Introduction to Optimization,* 2nd Edition, A Wiley Inter science publication. John Wiley and sons, Inc., New York/Chester/Weinhein/Brisbabe/Singapore/Toronto, 2001.

[7] J. W. Chinneck, *Practical Optimization*: A Gentle Introduction. http://www.sce.carelton.ca/ffaculty/chinneck/po.html 2008.

[8] J. O. Omolehin, *Further Results on CGM Algorithm,* ABACUS, Mathematics Association of Nigeria, **33**, (2006), 374-388.

[9] S. Soft, *Fortran 90 Users Guide*: A Sun Microsystems, Inc., California , U.S.A., 1995.

[10] J. O. Omolehin, K. Rauf, *Computational Capability of CGM Algorithm,* Creative Math., **15**, (2006), 41-46 Department of Mathematics and Computer Science, North University of Baimare, Romania, http://creative-mathematics.ubm.rol

[11] David L. Russel, *Optimization Theory,* W. A. Benjamin, Inc., New York, 1970

[12] E. Knill, *Nature (London)*: **434**,39 ; e-print arXiv:quantph/0410199, 2005.