ISSN - 2277 - 0593

Journal of Natural Sciences, Engineering and Technology

OPTIMIZATION ON $\Re^n$ BY COGGINS-FIBONACCI METHOD

***M.R. ODEKUNLE AND T.A. BADRU**

Department of Mathematics and Computer Science,
Federal University of Technology
PMB 2076, Yola, Adamawa State, Nigeria, 640001
***Corresponding author:** remiodekunle@yahoo.com

ABSTRACT

A computational procedure called Coggins-Fibonacci method for the optimization of unconstrained functions in $\Re^n$ is developed. The method is found to be more efficient and converges faster than either of the conventional Coggins or Fibonacci search methods.

Keywords: Search direction, Fibonacci search, Coggins method.

INTRODUCTION

Many search methods for unconstrained problems in optimization require searching for the minimal (maximal) point in a specified direction. These methods can be classified into two broad categories as direct search method and descent methods. The direct search methods require only objective function evaluation and do not use partial derivatives of the function in finding the minimum or maximum and are hence called the non-gradient methods. All the unconstrained minimization or maximization methods are iterative in nature and hence they start from initial trial solution x_0 and proceed towards the optimum point in a sequential manner. It is important to know that all the optimization methods differ from one another only in the method of generating new point x_{i+1} from x_i and in testing the point x_{i+1} for optimality.

The aim of all n-dimensional minimization techniques is to find a, the smallest non-negative value of a, for which the function

$$F(\alpha) = f(x + \alpha s) \tag{1}$$

attains a local minimum.

If the original function f(x) is expressible as an explicit function of x_i $(i = 1, 2, \ldots, n)$, we can readily write expression (1), for any specific vector s and then solve $\frac{df(\alpha)}{d\alpha} = 0$ to obtain a in terms of x and s. However, in many practical problems, the function $F(a)$ cannot be expressed explicitly in terms of a. In such cases the interpolation method can be used to find the value of a.

The Fibonacci and Coggins search methods (two-point search methods) can be used to solve unconstrained problems. The two

methods are efficient, requiring few functions evaluations and use unequally spaced points to bracket the minimum, followed by successive quadratic approximations, which result in rapid convergence to the optimum.

FIBONACCI SEARCH METHOD

This search technique is considered to be the best among the minimax methods (Foulds, 1981). It has the largest interval reduction of all of the procedures (Wallstreetcosmos.com, 2008). Unimodality is assumed and it requires that the number of experiments be specified in advance, which may be inconvenient. The deficiency has led to the development of other methods such as golden section search, lattice search, even block, odd block, golden block search and others. Walsh (1985) defined Fibonacci sequence $\{F_i\}$ as,

$$F_i = F_{i-1} + F_{i-2} \quad (i \geq 2), \quad F_0 = 0, \quad F_1 = F_2 = 1 \tag{2}$$

Fibonacci sequence was first discovered by Leonardo of Pisa (1175-1230) during an investigation of the rabbit population problem. It can be used to find optimal point of a function of one-variable even if the function is not continuous.

Fibonacci search direction

Suppose (x_1, x_2) brackets a required minimum of *f(x)*, the points x_3, x_4 are

$$x_3 = (1-\alpha_i)\, x_1 + \alpha_i x_2$$

$$x_4 = \alpha_i x_1 + (1-\alpha_i) x_2 \qquad 0 < a_i < \tfrac{1}{2} \tag{3}$$

symmetrically placed in this interval so that

By computing $f(x_i),\ i = 1, 2, 3, \ldots$,

either $f(x_4) > f(x_3)$ giving (x_3, x_2)

as the new bracket or $f(x_3) \geq f(x_4)$

giving (x_1, x_4) as the new bracket where

$$\alpha_i = \frac{F_{i-2}}{F_i}$$

If N is the total number of function evaluations to be performed, the test point for the ith iteration are

$$x_3^{(i)} = \frac{F_{N-1-i}}{F_{N+1-i}}(x_2^{(i)} - x_1^{(i)})$$

and

$$x_4^{(i)} = \frac{F_{N-i}}{F_{N+1-i}}(x_2^{(i)} - x_1^{(i)}) + x_i^{(i)}$$

COGGINS SEARCH METHOD

The method is a combination of a single variable technique proposed by Davies et al. (1974) and Powell. The algorithm proceeds as follows [10 and 13]:

i. A starting point is chosen and the objective function evaluated.
ii. The independent variable is incremented a distance Δx and the objective function evaluated again. If a function improvement is obtained, the step size is doubled for the next function evaluation. If a function improvement is not obtained on the first step, the direction is reversed and the next point located a distance Δx from the starting point.

i. After the first step, the step size is doubled if a function improvement is obtained and halved if a worse function evaluation is obtained.
ii. When a local optimum is encountered, the procedure will yield three points (x_k, x_{k-1}, x_{k-2}) straddling the optimum. An additional point x_{k+1} is then located $x_{k+1} = x_{k-1} + \frac{\Delta x}{2}$ where Δx is the current step size. The best three points are then retained (say x_1, x_2, x_3)
iii. A quadratic function is then curve fitted to the three retained points. The optimum x* is then located by setting

$$\frac{\partial f}{\partial x_i} = 0, \quad i = (1,2,3)$$ so that

$$x^* = \frac{1}{2}\left[\frac{(x_2^2 - x_3^2)f(x_1) + (x_3^2 - x_1^2)f(x_2) + (x_1^2 - x_2^2)f(x_3)}{(x_2 - x_3)f(x_1) + (x_3 - x_1)f(x_2) + (x_1 - x_2)f(x_3)}\right] \quad (5)$$

The objective function at x* is then compared with the best previous point subject to a convergence limit $\left|x^* - x_{i(best)}\right| \leq \lim it$

vi. If the above criterion is satisfied, the procedure stops. If not, the worst point is replaced by x* and a new quadratic surface fitted and the local optimum obtained.
This process is repeated until the convergence criterion is satisfied.

Extended Coggins method

Even though Coggins method was developed for one variable objective function, (Reju, 2991) and Subramaniam *et al.* (2002) have generalized the algorithm to that of multi-variables objective function based on the formalism of the one variable method while Bamgbola (2004) gave the theoretical backup. On Bamgbola (2004), taking note of the importance of the search direction which was omitted in [18] a better result was obtained. The details of the generalization and the algorithm can be found in Subramaniam (1994).

COGGINS-FIBONACCI METHOD

The classical search method, which obtains the optimum of a given function, requires several functions evaluation. In order to have fewer functions evaluation in obtaining the optimum, the Coggins-Fibonacci method was suggested for single variable and multivariable optimization.

The original formulations of Fibonacci and Coggins methods take care of optimization in one dimension. For extension to higher dimensions, there are many different directions that can be explored leading possibly to different results Bunday, 1984.

The inclusion of Fibonacci search direction in place of Coggins search direction makes Coggins-Fibonacci to have a better and faster convergence than Coggins method as illustrated in the accompanying numerical examples.

Algorithm for single variable Coggins-Fibonacci method

i. A starting point is chosen and the objective function evaluated.
ii. A suitable trial step length a_i is found along the directions s_i to obtain a new point $x_{i+1} = x_i + \alpha_i s_i$ where

$\alpha_i = \frac{F_{i-2}}{F_i}$ and $F_i = \frac{(b-a)}{\varepsilon}$ (Kahya, 2006 and Subasi *et al.,* 2004)

iii. After the first step, the step size, ε, and the difference between the initial values are used to determine the Fibonacci sequence.

iv. When a local optimum is encountered, the procedure will yield three points (x_k, x_{k-1}, x_{k-2}) straddling the optimum. An additional point x_{k+1} is then located by $x_{k+1}^i = (1-\alpha_i)x_{k-1}^i + \alpha_i x_k^i$ where α_i is the current step size. The best three points are then retained (say x_1, x_2, x_3).

v. A quadratic equation is then curve fitted to the three retained points. The optimum location x* is obtained by setting $\frac{\partial f}{\partial x_i} = 0, \quad (i = 1,2,3)$ so that

$$x^* = \frac{1}{2}\left[\frac{(x_1^2 - x_2^2)f(x_0) + (x_2^2 - x_0^2)f(x_1) + (x_0^2 - x_1^2)f(x_2)}{(x_1 - x_2)f(x_0) + (x_2 - x_0)f(x_1) + (x_0 - x_1)f(x_2)}\right]$$

The objective function at x* is then compared with the best previous point subject to a convergence limit

$$\left|x^* - x_{i(best)}\right| \le \lim it$$

vi. If the above criterion is satisfied, the procedure stops. If not, the worst point is replaced by x* and a new quadratic surface fitted and the local optimum obtained.

This process is repeated until the convergence criterion is satisfied.

Algorithm for multi-variables Coggins-Fibonacci method

The algorithm to find the optimum value of a function with more than one variable is listed in the steps here. As an illustration, consider an n-dimensional case.

i. The objective function is evaluated using the initial value $X^{(0)} (= x_0^{(1)}, x_0^{(2)}, ..., x_0^{(n)})$ and the trial value $X^{(1)} (= x_1^{(1)}, x_1^{(2)}, ..., x_1^{(n)})$.

ii. The third point $X^{(2)} (= x_2^{(1)}, x_2^{(2)}, ..., x_2^{(n)})$ is obtained as

$$x_2^{(1)} = (1-\alpha_i)x_0^{(1)} + (1-\alpha_i)x_1^{(1)} + ... + (1-\alpha_i)x_{n-1}^{(1)} + \alpha_i x_n^{(1)}$$
$$x_2^{(2)} = (1-\alpha_i)x_0^{(2)} + (1-\alpha_i)x_1^{(2)} + ... + (1-\alpha_i)x_{n-1}^{(2)} + \alpha_i x_n^{(2)}$$
$$\vdots$$
$$x_2^{(n)} = (1-\alpha_i)x_0^{(n)} + (1-\alpha_i)x_1^{(n)} + ... + (1-\alpha_i)x_{n-1}^{(n)} + \alpha_i x_n^{(n)} \quad (6)$$

$$\alpha_i = \frac{F_{i-2}}{F_i}$$

where, is the step length, F_i is found by specifying the initial trial step length ε and the boundary values (a, b) to ascertain the starting value of the Fibonacci sequence $\{F_i\}$.

iii. The new value $X^{(2)}$ and the initial values $X^{(0)}, X^{(1)}$ are used to evaluate the function as the first iteration.

iv When a local optimum is obtained, the procedure will yield three points say $X^{(k)}(= x_k^{\backslash(1)}, x_k^{(2)}, ..., x_k^n), X^{(k-1)}(= x_{k-1}^{\backslash(1)}, x_{k-1}^{(2)}, ..., x_{k-1}^{(n)})$ *and* $X^{(k-2)}(= x_{k-2}^{\backslash(1)}, x_{k-2}^{(2)}, ..., x_{k-2}^{(n)})$

straddling the optimum. Then an additional point $X^{(k+1)}(= x_{k+1}^{(1)}, x_{k+1}^{(2)},..., x_{k+1}^{n})$ is located using (Foulds, 1981). The best three points say $X^{(0)}, X^{(1)}\ and\ X^{(2)}$ are retained.

v. A quadratic equation is then curve-fitted to the three retained points and the optimum point X^* is located by setting

$$df = \frac{\partial f}{\partial X^{(1)}} dX^{(1)} + \frac{\partial f}{\partial X^{(2)}} dX^{(2)} + ... + \frac{\partial f}{\partial X^{(n)}} dX^{(n)} = 0$$

(Bamigbola, 2004)

vi. The values of the objective function at $X^{(1)} = X^{*(1)},..., X^{(n)} = X^{*(n)}$ are compared with the best previous point subject to the convergence criteria $\left|X^{*(1)} - X^{(1)}{}_{(best)}\right| \le \lim it,..., \left|X^{*(n)} - X^{(n)}{}_{(best)}\right| \le \lim it$ where, $X^{(1)}{}_{(best)}$ and $X^{(n)}{}_{(best)}$ are the best previous points. If the inequality is satisfied, the procedure stops otherwise, the worst points are replaced by $X^{*(1)},..., X^{*(n)}$ and a new quadratic surface is fitted and local optimum obtained.

Computational Examples

To illustrate the computational details as well as checking the workability and efficiency of our proposed method, the method was tested on various problems of various dimensions and the performance compared with some other known unconstrained minimization methods. The numerical examples used are the classical test cases used by earlier authors.

Minimize

1 $f(x_1, x_2) = 10(x_1 + x_2 - 5)^2 + (x_1 - x_2)^2$ Starting values are (0, 0)

Problems 5.2 – 5.5 are from [4].

2 $f(x_1, x_2) = 3803.84 - 138.08x_1 - 232.92\,x_2 + 123.08x_1^2 + 203.64x_2^2 + 182.25\,x_1x_2$

Starting values: (1, 0.5)

3 Gottfried function: Starting values are (0.5, 0.5)

$$f(x_1, x_2) = [x_1 - 0.1136(x_1 + 3x_2)(1 - x_1)]^2 + [x_2 + 7.5(2x_1 - x_2)(1 - x_2)]^2$$

4 Sisser's function: $f(x_1, x_2) = 3x_1^2 - 2x_1x_2 + 3x_2^2$ Starting values are (1, 0.1)

5 Rosenbrock's function: $f(x_1, x_2) = (1 - x_2)^2 + 100(x_2 - x_1^2)^2$
Starting values are (-1.2, 1)

6 Variably dimensioned function (Ibiejugba *et al.,* 1991)

$$f(x) = \sum_{i=1}^{n+2} f_i^2(x)$$

where n is the number of variables in the problem and

$$f_i(x) = x_i - 1, \quad i = 1,...,n$$

$$f_{n+1}(x) = \sum_{j=1}^{n} j(x_j - 1),$$

$$f_{n+2}(x) = \left(\sum_{j=1}^{n} j(x_j - 1) \right)^2$$

The initial values are $x^0 = (1 - (j/n))$. The minimizer x* = (1, 1, …, 1) with f(x*) = 0. The iterations results are shown in table 5 for n = 4 and n = 12.

7. Numeric font learning problem (Magoulas *et al.,* 1997); (Sperduti, 1993).
It is a well-known fact in the neural network field (NNF) (Haykij, 1994) that the rapid computation of the resulting global minimum problem is a difficult task. This is because the number of network variables is large and the corresponding nonconvex multi modal objective function possesses multitudes of local minima and has flat regions that are broad and adjoined with narrow steep ones. Due to the special characteristics of these problems, globally convergent schemes are required. Another problem associated with this class of problems is that of the choice of starting values. As very small initial values lead to very small corrections of the variables which may results in undesired local minimum so also can large initial values speed up the learning process and may again lead neurons to saturation and thus generate undesired results too. A way out is to choose the starting values between (x_{min}, x_{max}) (More *et al.*, 1981) where $x_{min} = -x_{max}$. In this example, we shall use the interval (-1, +1) choosing 1000 starting points randomly from this interval to rest our scheme.

This example involves the training of a multilayer feedforward neural network (FNN) with 460 variables for recognizing 8´8 pixel machine that prints numerals from 0 to 9. There are 64 input neurons and 10 output neurons representing 0 - 9. The numerals are in the form of a finite sequence $S = (s_1, s_2, ..., s_\ell)$ of input-output pairs $s_\ell = (r_\ell, t_\ell)$ where r_ℓ are the binary input vectors in $\Re^{64}$ determining the 8´8 binary pixel while t_ℓ are the binary output vectors in $\Re^{10}$ for $\ell = 1,..,10$. The corresponding objective function is,

$$f(x)=\sum_{\ell=1}^{10}\sum_{j=1}^{10}\left\{\left[1+\exp\left(\sum_{i=1}^{6}\alpha_{ij}y_{i,\ell}+\beta_j\right)\right]^{-1}-t_{j\ell}\right\}^2$$

where

$$y_{i,\ell}=\left[1+\exp\left(\sum_{k=1}^{6}\varphi_{ki}u_{k,\ell}+\rho_i\right)\right]^{-1}$$

and

$$x=(\alpha_{11},...,\alpha_{ij},...,\alpha_{6,10},\beta_1,...,\beta_j,...,\beta_{10},\varphi_{11},...,\varphi_{ki},...,\varphi_{64,6},\rho_1,...,\rho_i,...,\rho_6)$$

The iteration is terminated when $|f(x)|\le 10^{-4}$.

The numerical solutions obtained by implementing the Coggins-Fibonacci algorithm on these problems compared with other popular methods using Matlab 6.5 are shown in Tables 1 – 7.

Table 1: Computational results for problem 1

Method	x1*	x2*	f(x1*, x2*)	No. of iterations
Exact	2.5	2.5	0	-
Powell	2.5	2.5	0	5
Coggins	2.5	2.5	0	3
Proposed method	2.499999993	2.499999993	0	1

Table 2: Computational results for problem 2

Method	x1*	x2*	f(x1*, x2*)	No. of iterations
Exact	0.205658567	0.479863303	3733.756452	-
Subramaniam	0.205623709	0.479862599	3733.756452	492
Nelder	0.205188700	0.479829030	3733.756452	16
Hooke	0.205761720	0.479785160	3733.756452	110
Rosenbrock	0.206094820	0.479612030	3733.756452	62
Powell	0.205658570	0.479863300	3733.756452	3
Coggins	0.205651313	0.479863220	3733.756452	3
Proposed method	0.231895187	0.463790374	3733.756452	1

Table 3: Computational results for problem 3 (Gottfried function)

Method	x1*	x2*	f(x1*, x2*)	No. of iterations
Coggins	0.603063881	0.040355205	1.169803020	6
Proposed method	0.603588590	1.040340430	0.169803060	2

Table 4: Computational result for problem 4 (Sisser's function)

Method	x1*	x2*	f(x1*, x2*)	No. of iterations
Exact	0.000000000	0.000000000	0.000000000	-
Coggins	9.8697906E-05	7.8315839E-05	2.7803140E-16	19
Proposed method	0.000000000	0.000000000	0.000000000	1

Table 5: Computational result for problem 5 (Rosenbrock's function)

Method	x1*	x2*	f(x1*, x2*)	No. of iterations
Exact	1.000000000	1.000000000	0.000000000	-
Coggins	0.999937092	1.005153050	2.813191104E-03	4
Proposed method	0.999999999	0.999999999	0.000000000	1

Table 6: Computational results for the problem 6

Method	No. of iterations when n = 4	No. of iterations when n = 12
Fletcher-Reeves	11	15
Coggins	9	13
Proposed method	2	3

Table 7: Computational results for the problem 7 when n = 460

Method	Average no. of iterations	Success ratio
Fletcher-Reeves	620	420/1000
Coggins	310	820/1000
Proposed method	26	990/1000

DISCUSSION OF RESULTS

The stopping criteria for problems 1 to 6 is $\left|f(x^*)^{i+1} - f(x^*)^i\right| \le 10^{-9}$. For the Fletcher-Reeves method, Armijo line search was used. From tables 1-7, the proposed method shows better performance with 99% success (by success we mean the number of successful runs out of 1000 runs) in problem 7 and with clearly minimum number of function evaluations resulting in a faster running. The problems were run using Matlab 6.5 on HP Compact nx7300 laptop. The execution time for the new method is in general about 78% faster than the execution time for Coggins method of 4]. From these results, the new Coggins – Fibonacci method is shown to be consistently very accurate and comparable with other methods. In particular, the present method converges faster than Coggins or Fibonacci method.

Our computational experience in this work, as highlighted in section 1, supports the assertion that the efficiency of an optimization method depends on the search direction explored [20].

CONCLUSION

The Coggins – Fibonacci algorithm for solving n – dimensional ($n \ge 1$) unconstrained optimization problems is here proposed. The method is amenable to the usual mathematical analysis. By means of some test problems, the Coggins –Fibonacci method is shown to be no less inferior to other known methods for the rate of convergence is faster than most other methods.

REFERENCES

Adby, P.R., Dempster, M.A.H. 1974. *Introduction to optimization method.* Chapman & Hall, London.

Agusto, F.B. 2002. *Multivariable Coggins method*. M.Sc. Thesis, University of Ilorin, Ilorin, Nigeria.

Armijo, L. 1966. Minimization of function having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16: 1-3.

Bamigbola, O.M., Agusto, F.B. 2004. Optimization in Ân by Coggins method. *International Journal of Computer Mathematics*, 81(9): 1145-1152.

Bunday, B.D. 1984. *Basic optimization methods*. Edward Arnold, London.

Foulds, L.R. 1981. *Optimization techniques*. Springer-Verlag, New York.

Haykij, S. 1994. *Neural networks: A comprehensive foundation*. Macmillan college publishing company, New York (NY).

Ibiejugba, M.A., Adewale, T.A., Bamigbola, O.M. 1991. A Quasi-Newton method for minimizing factorable functions. *Afrika mathematika*, 4(2): 19–36.

Kahya, E. 2006. Comment on titled "An improvement on Fibonacci search method in optimization theory. *Applied mathematics and computation,* 176: 753-756.

Kuester, J.L., Mize, J.H. 1973. *Optimization techniques with Fortran.* McGraw-Hill, New York.

Magoulas, G.D., Vrahatis, M.N., Androulakis, G.S. 1997. Effective backpropagation training with variable stepsize. *Neural networks*, 10: 69-82.

More, B.J., Garbow, B.S., Hillstrom, K.E. 1981. Testing unconstrained optimization. *ACM Transactions on Mathematical Software*, 7: 17-41.

Rao, S.S. 1991. *Optimization theory and application*, Eastern John Wiley, New Delhi.

Reju, S.A. 2001. *Roc-CMAI lecture notes*. National Mathematical Centre, Abuja, Nigeria.

Sperduti, A., Starita, A. 1993. Speed up learning and network optimization with extended back-propagation. *Neural networks*, 6: 365-383.

Subasi, M., Yildirim, B., Yildiz, B. 2004. An improvement on Fibonacci search method in optimization theory. *Applied mathematics and computation*, 147(3): 893-901.

Subramaniam, S. 1994. *Extended Coggins optimization techniques*. B. Tech. Thesis, Department of Mathematics, Computer Science and Statistics, Federal University of Technology, Minna, Nigeria.

Subramaniam, S., Reju, S.A., Ibiejugba, M.A. 2002. An extended Coggins optimization method. *Nigeria Journal of Mathematics and Applications*, 8: 85-91.

Wallstreetcosmos.com, 2008. *Fibonacci numbers and stock market analysis*. Downloaded on March 14th, 2009.

Walsh, G.R. 1985. *Methods of optimization*. John Wiley & Sons, London.

(Manuscript received: 24th April, 2008; accepted: 31st August, 2009).