Applied Mathematics and Computation 217 (2010) 2608–2618

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

A novel cubically convergent iterative method for computing complex roots of nonlinear equations

R. Oftadeh *, M. Nikkhah-Bahrami, A. Najafi

School of Mechanical Engineering, College of Engineering, University of Tehran, Tehran, Iran

ARTICLE INFO

Keywords:
Root of continuous functions
Taylor expansion
Real and complex root
Number of iterations

ABSTRACT

A fast and simple iterative method with cubic convergent is proposed for the determination of the real and complex roots of any function $F(x) = 0$. The idea is based upon passing a defined function $G(x)$ tangent to $F(x)$ at an arbitrary starting point. Choosing $G(x)$ in the form of x^k or k^x, where k is obtained for the best correlation with the function $F(x)$, gives an added freedom, which in contrast to all existing methods, accelerates the convergence. Also, this new method can find complex roots just by a real initial guess. This is in contrast to many other methods like the famous Newton method that needs complex initial guesses for finding complex roots. The proposed method is compared to some new and famous methods like Newton method and a modern solver that is *fsolve* command in MATLAB. The results show the effectiveness and robustness of this new method as compared to other methods.

1. Introduction

Solving for the roots of equations such as $F(x) = 0$ is an old and known problem. The most famous and commonly used method is Newton method defined by:

$$x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)} \quad n \geqslant 0. \tag{1}$$

Other familiar methods are Bisection, Secant, False position, Brent, Halley, Schroder, Householder, Ridders, Muller, and Laguerr etc. which can be found in the literature. Also, in the recent years, many methods have been developed for solving nonlinear equations. These methods were developed using Taylor interpolating polynomials [1,2], quadrature formulas [3,4], decomposition [5,6], homotopy perturbation method [7,8], and other techniques [9,10]. Also, many Newton-type iterative methods have been developed for finding roots of nonlinear equations. From one point of view, these methods can be categorized as one-step [11,12], two-step [13,14] and three-step [15] iterative methods. Each of these methods has a different rate of convergence; second order [16,17], third order [11,18] and more than third order [15,19].

Most of these methods need a proper first guess of the root. Some of them calculate only the real roots and complex mode of computation is not possible, or if it is possible the initial guess must be complex (such as Newton's method). The proposed method introduced here does not have those weaknesses and can find both the real and the complex roots of any nonlinear function even if the initial guess was a real number.

Recently, the authors have developed a similar method for computing complex roots of systems of nonlinear equations [20]. We show here that the modified version of that method can effectively be used for finding the roots of a single nonlinear equation.

* Corresponding author.
E-mail addresses: roftadeh@me.ut.ac.ir, roftadeh@ut.ac.ir (R. Oftadeh).

0096-3003/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2010.07.075

2. The proposed method

Let the nonlinear function be represented by $F(x)$. Therefore, the nonlinear equation can be written as:

$$F(x) = 0 \tag{2}$$

Using the Taylor series expansion, we express the function in terms of an arbitrary function $G(x)$ which will be defined later.

$$F(x) = F(x_n) + \frac{a_1(x_n)}{1!}(G(x) - G(x_n)) + \frac{a_2(x_n)}{2!}(G(x) - G(x_n))^2 + \dots \tag{3}$$

Where

$$a_1(x_n) = \frac{F'(x_n)}{G'(x_n)} \tag{4}$$

$$a_{i+1}(x_n) = \frac{a_i'(x_n)}{G'(x_n)} \quad i = 2, 3, 4, \dots \tag{5}$$

Therefore

$$F(x) = F(x_n) + \frac{F'(x_n)}{G'(x_n)}(G(x) - G(x_n)) + \left(\frac{F''(x_n)}{G'^2(x_n)} - \frac{F'(x_n)G''(x)}{G'^3(x_n)}\right)(G(x) - G(x_n))^2 + \dots \tag{6}$$

By considering the above equation we can approximate $F(x)$ with:

$$F(x) \approx F(x_n) + \frac{F'(x_n)}{G'(x_n)}(G(x) - G(x_n)) \tag{7}$$

Let the right hand of Eq. (7) be represented by $H(x)$:

$$H(x) = F(x_n) + \frac{F'(x_n)}{G'(x_n)}(G(x) - G(x_n)) \tag{8}$$

In order for $H(x)$ to be compatible with $F(x)$ around $x = x_n$ we must have:

$$H(x_n) = F(x_n) \tag{9}$$

$$H'(x_n) = F'(x_n) \tag{10}$$

$$H''(x_n) = F''(x_n) \tag{11}$$

Conditions (9) and (10) are automatically satisfied. For condition (11) to be satisfied we must have:

$$\frac{F''(x_n)}{F'(x_n)} = \frac{G''(x_n)}{G'(x_n)} \tag{12}$$

Now by letting, $F(x) = 0$ from Eq. (7) we obtain:

$$x_{n+1} = G^{-1}\left(G(x_n) - G'(x_n)\frac{F(x_n)}{F'(x_n)}\right) \tag{13}$$

Note that Eq. (13) will be equal to the Newton formula, if we let $G(x) = x$.

3. Selection of $G(x)$

$G(x)$ must be selected in a way that can approximate any function. In addition, according to Eq. (13) the inverse of function $G(x)$ must be obtainable. Polynomials and exponential functions are usually appropriate for these purposes. So $G(x)$ can be expressed in one of the forms k^x, x^k or $exp(kx)$.

If $G(x) = x^k$ from Eqs. (12) and (13) we have:

$$\begin{aligned} &(12) \to \frac{F''(x_n)}{F'(x_n)} = \frac{(k-1)(k)x_n^{k-2}}{(k)x_n^{k-1}} \Rightarrow k = 1 + \frac{x_nF''(x_n)}{F'(x_n)} \\ &(13) \to x_{n+1} = \left(x_n^k - k\frac{x_n^{k-1}F(x_n)}{F'(x_n)}\right)^{1/k} \Rightarrow x_{n+1} = x_n \times \left(1 - k\frac{F(x_n)}{x_nF'(x_n)}\right)^{1/k} \quad n = 0, 1, 2, \dots \end{aligned} \tag{14}$$

If $G(x) = k^x$, then: