2009

þÿA third-order modification of I
method for multiple roots

Chun, Changbum

Applied Mathematics and Computation 211 (2009) 474–479

Contents lists available at ScienceDirect

Applied Mathematics and Computation

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

A third-order modification of Newton's method for multiple roots

Changbum Chun [a], Beny Neta [b,*]

[a] Department of Mathematics, Sungkyunkwan University, Suwon 440-746, Republic of Korea
[b] Department of Applied Mathematics, Naval Postgraduate School, Monterey, CA 93943, United States

ARTICLE INFO

Keywords:
Newton's method
Multiple roots
Iterative methods
Nonlinear equations
Order of convergence
Root-finding

ABSTRACT

In this paper, we present a new third-order modification of Newton's method for multiple roots, which is based on existing third-order multiple root-finding methods. Numerical examples show that the new method is competitive to other methods for multiple roots.

1. Introduction

Solving nonlinear equations is one of the most important problems in numerical analysis. In this paper, we consider iterative methods to find a multiple root α of multiplicity m, i.e., $f^{(j)}(\alpha) = 0,\ j = 0, 1, \ldots, m-1$ and $f^{(m)}(\alpha) \neq 0$, of a nonlinear equation $f(x) = 0$.

Newton's method is only of first order unless it is modified to gain the second order of convergence, see Schröder [1]. This modification requires a knowledge of the multiplicity. Traub [2] has suggested to use any method for $f^{(m)}(x)$ or $g(x) = \frac{f(x)}{f'(x)}$. Any such method will require higher derivatives than the corresponding one for simple zeros. Also the first one of those methods require the knowledge of the multiplicity m. In such a case, there are several other methods developed by Hansen and Patrick [3], Victory and Neta [4], Dong [5,6], Neta and Johnson [7], Neta [8] and Werner [9]. See also Neta [10]. Since in general one does not know the multiplicity, Traub [2] suggested a way to approximate it during the iteration. The way it is done is by evaluating the quotient

$$\frac{x_{n-2} - x_n}{x_{n-2} - x_{n-1}}$$

and rounding the number up.

For example, the quadratically convergent modified Newton's method is (see [1])

$$x_{n+1} = x_n - m\frac{f_n}{f'_n} \tag{1}$$

and the cubically convergent Halley's method [11] is a special case of the Hansen and Patrick's method [3]

$$x_{n+1} = x_n - \frac{f_n}{\frac{m+1}{2m} f'_n - \frac{f_n f''_n}{2 f'_n}}, \tag{2}$$

where $f_n^{(i)}$ is short for $f^{(i)}(x_n)$. Another third-order method was developed by Victory and Neta [4] and based on King's fifth order method (for simple roots) [12]

* Corresponding author.
E-mail addresses: cbchun@skku.edu (C. Chun), bneta@nps.edu (B. Neta).

0096-3003/$ - see front matter Published by Elsevier Inc.
doi:10.1016/j.amc.2009.01.087

$$\begin{aligned} y_n &= x_n - u_n, \\ x_{n+1} &= y_n - \frac{f(y_n)}{f_n'}\frac{f_n + Af(y_n)}{f_n + Bf(y_n)}, \end{aligned} \tag{3}$$

where

$$\begin{aligned} A &= \mu^{2m} - \mu^{m+1}, \\ B &= -\frac{\mu^m(m-2)(m-1)+1}{(m-1)^2}, \end{aligned} \tag{4}$$

$$\mu = \frac{m}{m-1} \tag{5}$$

and

$$u_n = \frac{f_n}{f_n'}. \tag{6}$$

Dong [5] has developed two third-order methods requiring two evaluations of f and one evaluation of f'

$$\begin{cases} y_n = x_n - \sqrt{m}\,u_n, \\ x_{n+1} = y_n - m\left(1 - \frac{1}{\sqrt{m}}\right)^{1-m}\frac{f(y_n)}{f_n'}, \end{cases} \tag{7}$$

$$\begin{cases} y_n = x_n - u_n, \\ x_{n+1} = y_n + \frac{u_n f(y_n)}{f(y_n) - \left(1-\frac{1}{m}\right)^{m-1} f_n}, \end{cases} \tag{8}$$

where u_n is given by (6).

Yet two other third-order methods developed by Dong [6], both require the same information and both based on a family of fourth order methods (for simple roots) due to Jarratt [13]:

$$\begin{cases} y_n = x_n - u_n, \\ x_{n+1} = y_n - \frac{f_n}{\left(\frac{m}{m-1}\right)^{m+1} f'(y_n) + \frac{m-m^2-1}{(m-1)^2} f_n'}, \end{cases} \tag{9}$$

$$\begin{cases} y_n = x_n - \frac{m}{m+1} u_n, \\ x_{n+1} = y_n - \frac{\frac{m}{m+1} f_n}{\left(1+\frac{1}{m}\right)^m f'(y_n) - f_n'}, \end{cases} \tag{10}$$

where u_n is given by (6).

Osada [14] has developed a third-order method using the second derivative,

$$x_{n+1} = x_n - \frac{1}{2}m(m+1)u_n + \frac{1}{2}(m-1)^2\frac{f_n'}{f_n''}, \tag{11}$$

where u_n is given by (6).

Neta and Johnson [7] have developed a fourth order method requiring one function- and three derivative-evaluation per step. The method is based on Jarratt's method [15] given by the iteration

$$x_{n+1} = x_n - \frac{f_n}{a_1 f_n' + a_2 f'(y_n) + a_3 f'(\eta_n)}, \tag{12}$$

where u_n is given by (6) and

$$\begin{aligned} y_n &= x_n - a u_n, \\ v_n &= \frac{f_n}{f'(y_n)}, \\ \eta_n &= x_n - b u_n - c v_n. \end{aligned} \tag{13}$$

Neta and Johnson [7] give a table of values for the parameters a, b, c, a_1, a_2, a_3 for several values of m. In the case $m = 2$ they found a method that will require only two derivative-evaluations ($a_3 = 0$). This was not possible for higher m.

Neta [8] has developed a fourth order method requiring one function- and three derivative-evaluation per step. The method is based on Murakami's method [16] given by the iteration

$$x_{n+1} = x_n - a_1 u_n - a_2 v_n - a_3 w_3(x_n) - \psi(x_n), \tag{14}$$

where u_n is given by (6), v_n, y_n, and η_n are given by (13) and

$$\begin{aligned} w_3(x_n) &= \frac{f_n}{f'(\eta_n)}, \\ \psi(x_n) &= \frac{f_n}{b_1 f_n' + b_2 f'(y_n)}. \end{aligned} \tag{15}$$

Neta [8] gives a table of values for the parameters $a, b, c, a_1, a_2, a_3, b_1, b_2$ for several values of m.

A method of order 1.5 requiring two function- and one derivative-evaluation is given by Werner [9]. It is only for double roots

$$\begin{aligned} y_n &= x_n - u_n, \\ x_{n+1} &= x_n - s_n u_n, \end{aligned} \tag{16}$$

where

$$s_n = \begin{cases} \frac{2}{1+\sqrt{1-4f(y_n)/f_n}}, & \text{if } f(y_n)/f_n \leqslant \frac{1}{4}, \\ \frac{1}{2} f_n / f(y_n), & \text{otherwise}. \end{cases}$$

Later we give a table comparing the efficiency of all known methods for multiple roots including our new ones we develop here. The informational efficiency, E, of a method of order p using d function/derivative evaluations is defined [2] as

$$E = \frac{p}{d}. \tag{17}$$

The efficiency index, I, is defined as

$$I = p^{1/d}. \tag{18}$$

It can be seen in Table 1 that our new method is competitive with the previously developed schemes.

There is an approach in [17] that uses any pair of existing methods in constructing new iterative methods or families of the same order in the case of simple roots. Now, a simple question arises; does the approach also work for multiple roots case? The answer indeed is affirmative as will be seen in the following section. To this end and for the sake of illustration, two methods for multiple roots are considered. One is Osada's third-order method (11) and the other is the Euler–Chebyshev method of order three [2]

$$x_{n+1} = x_n - \frac{m(3-m)}{2}\frac{f(x_n)}{f'(x_n)} - \frac{m^2}{2}\frac{f(x_n)^2 f''(x_n)}{f'(x_n)^3}. \tag{19}$$

The methods (11) and (19) are used to obtain a new modification of Newton's method for multiple roots. The new method is shown to be cubically convergent by analysis of convergence, its performance and practical utility are demonstrated by numerical examples.

2. Development of method and convergence analysis

Now, we approximately equate the correcting terms of both methods (11) and (19) to obtain the following approximate expression:

$$-\frac{1}{2}m(m+1)\frac{f(x_n)}{f'(x_n)} + \frac{1}{2}(m-1)^2\frac{f'(x_n)}{f''(x_n)} \approx -\frac{m(3-m)}{2}\frac{f(x_n)}{f'(x_n)} - \frac{m^2}{2}\frac{f(x_n)^2 f''(x_n)}{f'(x_n)^3}, \tag{20}$$

Table 1
Comparison of methods for multiple roots.

Algorithm	p	d	E	I	f'	f''	f'''
Werner [9] (16) $m=2$	1.5	3	0.5	1.145	1		
Schröder [1] (1)	2	2	1	1.414	1		
Hansen and Patrick [3]	3	3	1	1.442	1	1	
Halley (2)	3	3	1	1.442	1	1	
Laguerre	3	3	1	1.442	1	1	
Hansen and Patrick [3]	3	4	.75	1.316	1	1	1
Victory and Neta [4] (3)	3	3	1	1.442	1		
Dong [5] (7), (8)	3	3	1	1.442	1		
Dong [6] (9), (10)	3	3	1	1.442	2		
Osada [14]	3	3	1	1.442	1	1	
Neta and Johnson [7] (12) $m \neq 2$	4	4	1	1.414	3		
Neta and Johnson [7] (12) $m = 2$	4	3	1.333	1.587	2		
Neta [8]	4	4	1	1.414	1	3	
Neta [19] $m \neq 3$	3	3	1	1.442	1	1	
Neta [19] $m = 3$	2	3	.667	1.259	1	1	
Neta [19]	3	3	1	1.442	1		
Neta [19]	2.732	2	1.366	1.653	1		
Neta [19]	2.732	2	1.366	1.653	1		
Chun and Neta (22)	3	3	1	1.442	1	1	

this gives a new approximation

$$\frac{f(x_n)f''(x_n)}{2f'(x_n)} \approx \frac{m-1}{m}f'(x_n) - \frac{1}{2}\frac{(m-1)^2}{m^2}\frac{f'(x_n)^3}{f(x_n)f''(x_n)}. \tag{21}$$

If we apply the approximation (21) to Halley's method (2), then we obtain a new method

$$x_{n+1} = x_n - \frac{2m^2 f(x_n)^2 f''(x_n)}{m(3-m)f(x_n)f'(x_n)f''(x_n) + (m-1)^2 f'(x_n)^3}. \tag{22}$$

For (22), we have

Theorem 1. *Let $\alpha \in I$ be a multiple root of multiplicity m of a sufficiently differentiable function $f : I \to \mathbf{R}$ for an open interval I. If x_0 is sufficiently close to α, then the method defined by (22) has third-order convergence, and satisfies the error equation*

$$e_{n+1} = \frac{(m^2+3)C_1^2 - 2m(m-1)C_2}{2m^2(m-1)}\, e_n^3 + O(e_n^4), \tag{23}$$

where $e_n = x_n - \alpha$ and $C_j = \frac{m!}{(m+j)!}\frac{f^{(m+j)}(\alpha)}{f^{(m)}(\alpha)}$.

Proof. Using Taylor expansion of $f(x_n)$ about α, we have

$$f(x_n) = \frac{f^{(m)}(\alpha)}{m!}e_n^m\left[1 + C_1 e_n + C_2 e_n^2 + O(e_n^3)\right], \tag{24}$$

$$f'(x_n) = \frac{f^{(m)}(\alpha)}{(m-1)!}e_n^{m-1}\left[1 + \frac{m+1}{m}C_1 e_n + \frac{m+2}{m}C_2 e_n^2 + O(e_n^3)\right], \tag{25}$$

$$f'(x_n)^2 = \frac{f^{(m)}(\alpha)^2}{[(m-1)!]^2}e_n^{2m-2}\left\{1 + 2\frac{m+1}{m}C_1 e_n + \left[\left(\frac{m+1}{m}C_1\right)^2 + 2\frac{m+2}{m}C_2\right]e_n^2 + O(e_n^3)\right\}, \tag{26}$$

$$f''(x_n) = \frac{f^{(m)}(\alpha)}{(m-2)!}e_n^{m-2}\left[1 + \frac{m+1}{m-1}C_1 e_n + \frac{(m+1)(m+2)}{m(m-1)}C_2 e_n^2 + O(e_n^3)\right], \tag{27}$$

where $C_j = \frac{m!}{(m+j)!}\frac{f^{(m+j)}(\alpha)}{f^{(m)}(\alpha)}$ and $e_n = x_n - \alpha$.

Dividing (24) by (25) gives us

$$\frac{f(x_n)}{f'(x_n)} = \frac{e_n}{m}\left[1 - \frac{1}{m}C_1 e_n + \frac{(m+1)C_1^2 - 2mC_2}{m^2}e_n^2 + O(e_n^3)\right]. \tag{28}$$

We now use Maple [18] to collect all these expansions into (22) to have

$$\begin{aligned} m(3-m)f(x_n)f''(x_n) + (m-1)^2(f'(x_n))^2 = \frac{e_n^{2m-2}}{((m-1)!)^2}\Big\{&2(m-1) + \frac{2}{m}(2m^2 - m + 1)C_1 e_n \\ &+ \left[\frac{2m^3+m^2+1}{m^2}C_1^2 + \frac{4m^2-2m+10}{m}C_2\right]e_n^2 \\ &+4\left[\frac{m^3+m^2+m+1}{m^2}C_1C_2 + \frac{m^2-m+6}{m}C_3\right]e_n^3\Big\}, \end{aligned} \tag{29}$$

$$\begin{aligned} 2m^2 f(x_n)f''(x_n) = \frac{e_n^{2m-2}}{((m-1)!)^2}\Big\{&2m(m-1) + 4m^2C_1e_n + \left[2m(m+1)C_1^2 + 4(m^2+m+1)C_2\right]e_n^2 \\ &+4\left[(m^2+2m+3)C_3 + (m+1)^2C_1C_2\right]e_n^3\Big\}. \end{aligned} \tag{30}$$

Now divide (30) by (29) and multiply by (28) and collect terms we have

$$e_{n+1} = \frac{(m^2+3)C_1^2 - 2m(m-1)C_2}{2m^2(m-1)}\, e_n^3 + O(e_n^4), \tag{31}$$

which indicates that the order of convergence of the methods defined by (22) is at least three. This completes the proof. □

Note that the method requires one evaluation of the function, the first and the second derivative at each step, therefore the informational efficiency is $E = 1$ and the efficiency index is $I = 1.442$.

In a similar fashion, the proposed approach can be continuously applied to produce various types of new approximations from the other iterative methods for multiple roots, which can in turn be freely employed to find many new iterative methods or families for multiple roots.

3. Numerical examples

We present some numerical test results for various third-order multiple root-finding methods as well as our new methods and the Newton method in Table 3. Compared were the Newton method (1) (NM), Halley-like method (2) (HM), Osada's method (11) (OM), the Euler–Chebyshev method (19) (ECM), and the method (22) (CM) introduced in this contribution. All computations were done using MAPLE with 128 digit floating point arithmetics (Digits := 128). Displayed in Table 3 are the number of iterations (IT) required such that $|f(x_n)| < 10^{-32}$, and the value of $|f(x_n)|$ after the required iterations.

The following functions are used for the comparison and we display the approximate zeros x_* found up to the 28th decimal places

$$f_1(x) = (x^3 + 4x^2 - 10)^3, \quad x_* = 1.3652300134140968457608068290,$$
$$f_2(x) = (\sin^2 x - x^2 + 1)^2, \quad x_* = 1.4044916482153412260350868178,$$
$$f_3(x) = (x^2 - e^x - 3x + 2)^5, \quad x_* = 0.25753028543986076045536730494,$$
$$f_4(x) = (\cos x - x)^3, \quad x_* = 0.73908513321516064165531208767,$$
$$f_5(x) = ((x-1)^3 - 1)^6, \quad x_* = 2,$$
$$f_6(x) = (xe^{x^2} - \sin^2 x + 3\cos x + 5)^4, \quad x_* = -1.2076478271309189270094167584,$$
$$f_7(x) = (\sin x - x/2)^2, \quad x_* = 1.8954942670339809471440357381.$$

We also ran the second case with various initial points and average the number of iterations required for convergence. These results are given in Table 2.

The results presented in Tables 2 and 3 show that for the functions we tested, the new method introduced here have at least equal performance as compared to the other multiple root-finding methods of the same order, and also converge more rapidly than Newton's method for multiple roots.

Table 2
Comparison of various third-order multiple root-finding methods and Newton's method. The last row gives the average number of iterations required for convergence when starting from various initial guesses.

x_0	NM	HM	OM	ECM	CM
0.0001	46	23	37	68	35
1	7	5	6	6	6
2	7	5	5	5	5
3	7	5	6	6	6
4	7	6	5	5	5
5	8	5	6	6	6
6	8	6	7	7	7
7	8	6	7	6	6
8	9	6	7	6	7
9	9	9	7	6	6
	11.6	7.6	9.3	12.1	8.9

Table 3
Comparison of various third-order multiple root-finding methods and Newton's method.

$f(x)$	IT ($\lvert f(x_n)\rvert$)				
	NM	HM	OM	ECM	CM
$f_1, x_0 = 2$	6(8.49e−54)	4(7.06e−49)	4(6.47e−33)	4(4.01e−38)	4(4.01e−38)
$f_1, x_0 = 1$	6(4.91e−62)	4(3.38e−57)	5(5.40e−84)	4(1.94e−38)	4(1.94e−38)
$f_2, x_0 = 2.3$	7(7.31e−52)	5(4.84e−57)	5(2.07e−38)	5(1.73e−47)	5(4.55e−42)
$f_2, x_0 = 2$	7(5.11e−64)	5(7.43e−77)	5(3.53e−51)	5(1.53e−63)	5(4.09e−56)
$f_3, x_0 = 0$	4(1.03e−55)	3(1.68e−53)	3(5.83e−62)	3(4.31e−58)	3(1.71e−55)
$f_3, x_0 = 1$	4(3.46e−52)	4(1.39e−85)	4(2.01e−91)	4(2.24e−89)	4(1.93e−87)
$f_4, x_0 = 1.7$	5(6.04e−47)	4(9.12e−43)	4(1.17e−39)	4(5.25e−41)	4(5.25e−41)
$f_4, x_0 = 1$	5(1.22e−60)	4(1.78e−85)	4(1.42e−78)	4(1.43e−81)	4(-1.43e−81)
$f_5, x_0 = 3$	6(2.70e−45)	4(7.44e−45)	5(3.12e−85)	5(1.89e−94)	4(3.55e−37)
$f_5, x_0 = -1$	10(5.23e−49)	11(2.22e−65)	24(7.70e−44)	23(1.87e−52)	5(2.67e−77)
$f_6, x_0 = -2$	8(5.60e−37)	5(1.60e−61)	6(5.09e−45)	6(3.21e−64)	6(2.83e−82)
$f_6, x_0 = -1$	6(5.61e−60)	3(4.75e−35)	5(1.56e−103)	4(1.47e−47)	4(9.70e−58)
$f_7, x_0 = 1.7$	6(3.80e−57)	4(7.40e−47)	5(1.81e−76)	4(1.01e−37)	5(1.03e−92)
$f_7, x_0 = 2$	5(2.09e−40)	4(1.55e−65)	4(3.45e−53)	4(1.67e−59)	4(8.23e−56)

References

[1] E. Schröder, Über unendlich viele Algorithmen zur Auflösung der Gleichungen, Math. Ann. 2 (1870) 317–365.
[2] J.F. Traub, Iterative Methods for the Solution of Equations, Prentice Hall, New Jersey, 1964.
[3] E. Hansen, M. Patrick, A family of root finding methods, Numer. Math. 27 (1977) 257–269.
[4] H.D. Victory, B. Neta, A higher order method for multiple zeros of nonlinear functions, Int. J. Comput. Math. 12 (1983) 329–335.
[5] C. Dong, A basic theorem of constructing an iterative formula of the higher order for computing multiple roots of an equation, Math. Numer. Sinica 11 (1982) 445–450.
[6] C. Dong, A family of multipoint iterative functions for finding multiple roots of equations, Int. J. Comput. Math. 21 (1987) 363–367.
[7] B. Neta, A.N. Johnson, High order nonlinear solver for multiple roots, Comput. Math. Appl. 55 (2008) 2012–2017, doi:10.1016/j.camwa.2007.09.001.
[8] B. Neta, Extension of Murakami's High order nonlinear solver to multiple roots, Int. J. Comput. Math., in press, doi:10.1080/00207160802272263.
[9] W. Werner, Iterationsverfahren höherer Ordnung zur Lösung nicht linearer Gleichungen, Z. Angew. Math. Mech. 61 (1981) T322–T324.
[10] B. Neta, Numerical Methods for the Solution of Equations, Net-A-Sof, California, 1983.
[11] E. Halley, A new, exact and easy method of finding the roots of equations generally and that without any previous reduction, Phil. Trans. Roy. Soc. London 18 (1694) 136–148.
[12] R.F. King, A family of fourth order methods for nonlinear equations, SIAM J. Numer. Anal. 10 (1973) 876–879.
[13] P. Jarratt, Some fourth order multipoint methods for solving equations, Math. Comput. 20 (1966) 434–437.
[14] N. Osada, An optimal multiple root-finding method of order three, J. Comput. Appl. Math. 51 (1994) 131–133.
[15] P. Jarratt, Multipoint iterative methods for solving certain equations, Comput. J. 8 (1966) 398–400.
[16] T. Murakami, Some fifth order multipoint iterative formulae for solving equations, J. Inform. Process. 1 (1978) 138–139.
[17] C. Chun, A simply constructed third-order modifications of Newton's method, J. Comput. Appl. Math. (2007), doi:10.1016/j.cam.2007.07.004.
[18] D. Redfern, The Maple Handbook, Springer-Verlag, New York, 1994.
[19] B. Neta, New third order nonlinear solvers for multiple roots, Appl. Math. Comput. 202 (2008) 162–170, doi:10.1016/j.amc.2008.01.031.