Nonlinear Analysis: Modelling and Control, Vol. 21, No. 1, 121–134 ISSN 1392-5113
http://dx.doi.org/10.15388/NA.2016.1.9

On the optimality of some multi-point methods for finding multiple roots of nonlinear equation*

Nebojša M. Ralević, Dejan Ćebić

Faculty of Engineering, University of Novi Sad,
Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia
nralevic@uns.ac.rs; cebicd@gmail.com

Received: October 9, 2014 / **Revised:** July 5, 2015 / **Published online:** November 25, 2015

Abstract. This paper deals with the problem of determining the multiple roots of nonlinear equations, where the multiplicity of the roots is known. The paper contains some remarks on the optimality of the recently published methods [B. Liu, X. Zhou, A new family of fourth-order methods for multiple roots of nonlinear equations, *Nonlinear Anal. Model. Control*, 18(2):143–152, 2013] and [X. Zhou, X. Chen, Y. Song, Families of third- and fourth-order methods for multiple roots of nonlinear equations, *Appl. Math. Comput.*, 219(11):6030–6038, 2013]. Separate analysis of odd and even multiplicity, has shown the cases where those methods lose their optimal convergence properties. Numerical experiments are made and they support theoretical analysis.

Keywords: nonlinear equation, multiple roots, the modified Newton's method, optimal order of convergence.

1 Introduction

The problem of solving nonlinear equation $f(x) = 0$ is one of the most frequent problems in numerical mathematics and engineering. The classical Newton's method

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{1}$$

is the most commonly used iterative method for finding the solutions of $f(x) = 0$. When the initial iteration x_0 is close enough to the root α, the Newton's method has quadratic convergence if α is a simple root, or linear convergence if α is a multiple root.

According to the Kung–Traub conjecture, the iterative method is optimal if it reaches order of convergence 2^n using $n + 1$ function or derivative evaluations per iteration. Therefore, the classical Newton's method is optimal only for simple roots. When the

*This research was supported by grant No. 174009, Ministry of Education, Science and Technological Development of the Republic of Serbia.

multiplicity of α (denoted by m) is known, more suitable method is the modified Newton's method given by

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

which has optimal properties, i.e. it converges quadraticaly.

Based on the methods (1), (2) and on the work of Jarratt [4], many multipoint methods for finding multiple roots of $f(x) = 0$, where the multiplicity of roots is known in advance have been constructed in order to improve rate of convergence. In this paper, we are concerned only with the multipoint methods free from the second or higher order derivatives. In the literature, one can find the third-order methods, which require three function or derivative evaluations per iteration (see for example [1,2,3,5,9,10,13,15,17]), or the fourth-order methods with four function/derivative evaluations (see [10, 12]), but these methods are not optimal in the sense of Kung–Traub.

Beside them, there are other more efficient optimal methods for multiple roots. In the further text, we briefly recall only optimal fourth-order methods that are most commonly used and compare them with the recently developed optimal iteration schemes based on the modified Newton's method.

First, we present the method developed by Li et al. [7] denoted by LLC

$$\begin{aligned} y_n &= x_n - \frac{2m}{m+2}\frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= x_n - \frac{\frac{1}{2}m(m-2)(\frac{m}{m+2})^{-m}f'(y_n) - \frac{m^2}{2}f'(x_n)}{f'(x_n) - (\frac{m}{m+2})^{-m}f'(y_n)}\frac{f(x_n)}{f'(x_n)}. \end{aligned}$$

The following method, denoted by ShSh, has been constructed by Sharma and Sharma in [16]:

$$\begin{aligned} y_n &= x_n - \frac{2m}{m+2}\frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= x_n - a_1\frac{f(x_n)}{f'(x_n)} - a_2\frac{f(x_n)}{f'(y_n)} - a_3\frac{f(x_n)f'(x_n)}{f'(y_n)^2}, \end{aligned}$$

where $a_1 = (1/8)m(m^3 - 4m + 8)$, $a_2 = -(1/4)m(m-1)(m+2)^2(m/(m+2))^m$ and $a_3 = (1/8)m(m+2)^3(m/(m+2))^{2m}$.

In [6], Li et al. have introduced several fourth-order methods, but we restrict our attention on the following method (LCN) with optimal properties:

$$\begin{aligned} y_n &= x_n - \frac{2m}{m+2}\frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= x_n - a_1\frac{f(x_n)}{f'(x_n)} - \frac{f(x_n)}{a_2f'(x_n) + a_3f'(y_n)}, \end{aligned} \tag{3}$$

where

$$a_1 = -\frac{1}{2}m^2 + m, \quad a_2 = -\frac{1}{m} \quad \text{and} \quad a_3 = \left(\frac{m}{m+2}\right)^{-m}\frac{1}{m}.$$

Zhou et al. [18] have developed a more general version of (3), where iteration scheme is defined by

$$\begin{aligned} y_n &= x_n - \frac{2m}{m+2}\frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= x_n - Q\left(\frac{f'(y_n)}{f'(x_n)}\right)\frac{f(x_n)}{f'(x_n)}, \end{aligned} \tag{4}$$

where the function $Q(\cdot) \in C^2(\mathbb{R})$ satisfies the following conditions: $Q(u) = m$, $Q'(u) = -(1/4)m^{3-m}(m+2)^m$ and $Q''(u) = (1/4)m^4(m/(m+2))^{-2m}$ for $u = (m/(m+2))^{m-1}$. Beside the method (3), in further analysis we use another member of the family (4), where the function $Q(\cdot)$ is chosen such that

$$\begin{gathered} Q(t) = At^2 + Bt + C, \\ A = \frac{1}{8}m^4\left(\frac{m+2}{m}\right)^{2m}, \qquad B = -\frac{1}{4}m^3(m+3)\left(\frac{m+2}{m}\right)^{m}, \\ C = \frac{1}{8}m\big(m^3 + 6m^2 + 8m + 8\big). \end{gathered}$$

This method is denoted by ZCS.

Recently, Rhee and Kim have presented a more general family of fourth-order methods (see [14] for details), and we list here just two special cases, which are tested and numerically verified by authors. They are defined by iteration schemes RK1 and RK2 as follows:

RK1:
$$\begin{aligned} y_n &= x_n - \left(\frac{2m}{m+2} + \frac{h^3}{h+1}\right)h, \\ x_{n+1} &= x_n - h\big(Av^2 + Bv + C\big), \end{aligned}$$

where

$$h = \frac{f(x_n)}{f'(x_n)}, \qquad v = \frac{f'(y_n)}{f'(x_n)}, \qquad \rho = \left(\frac{m}{m+2}\right)^{m-1},$$
$$A = \frac{(m(m+2))^2}{8\rho^2}, \quad B = \frac{m^2(m+2)(m+3)}{-4\rho} \quad \text{and} \quad C = \frac{m(m^3+6m^2+8m+8)}{8};$$

RK2:
$$\begin{aligned} y_n &= x_n - \left(\frac{2m}{m+2} + \frac{h^3}{h+1}\right)h, \\ x_{n+1} &= x_n - h\left(\frac{A + Bv^3}{C + v^3}\right), \end{aligned}$$

where

$$h = \frac{f(x_n)}{f'(x_n)}, \qquad v = \frac{f'(y_n)}{f'(x_n)}, \qquad \rho = \left(\frac{m}{m+2}\right)^{m-1},$$
$$A = \frac{\rho^3 m(m^2+4)}{2(m+4)}, \quad B = -\frac{m(m^2-8)}{2(m+4)} \quad \text{and} \quad C = -\frac{\rho^3(m-2)}{m+4}.$$

Since all the methods mentioned above are of fourth order and require one function and two first derivative evaluations per iteration, their efficiency index calculated by $p^{1/n}$ (p is convergence order and n is number of function/derivative evaluations per iteration) equals $4^{1/3} \approx 1.587$, while for the method (2) it is $2^{1/2} \approx 1.414$. Furthermore, all these methods are based on the Jarratt method.

On the other hand, recently developed method, obtained by Liu and Zhou [8], is based on the modified Newton's method (2) with iteration scheme

$$\begin{aligned} y_n &= x_n - m\frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= y_n - mQ(w_n)\frac{f(x_n)}{f'(x_n)}, \end{aligned} \tag{5}$$

where $w_n = \sqrt[m-1]{f'(y_n)/f'(x_n)}$, and the function $Q(\cdot)$ satisfies conditions

$$Q(0) = 0, \quad Q'(0) = 1, \quad Q''(0) = \frac{4m}{m-1} \quad \text{and} \quad Q'''(0) < \infty, \tag{6}$$

demanded for reaching the fourth order of convergence. Two special choices of $Q(w_n)$, slightly modified from those suggested in [8], will be considered to illustrate numerical behavior of the methods, denoted by LZ1 and LZ2 for

$$Q(w_n) = w_n + \frac{2m}{m-1}w_n^2 + kw_n^3 \quad \text{and} \quad Q(w_n) = \frac{(m-1)w_n}{m-1-2mw_n},$$

respectively, where k is some real coefficient.

Although this method requires computation of the $(m-1)$st root in every iteration, it does not affect the cost of the method. In [11], Neta et al. have numerically verified that evaluations of $(m-1)$st root do not significantly increase CPU time compared with other methods.

To construct a family of two-point methods using only one derivative and two function evaluations per iteration, Zhou et al. have investigated in [19] the following method:

$$\begin{aligned} y_n &= x_n - m\frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= y_n - mG(w_n)\frac{f(x_n)}{f'(x_n)}, \end{aligned} \tag{7}$$

where $w_n = \sqrt[m]{f(y_n)/f(x_n)}$, and the function $G(\cdot)$ satisfies the following conditions:

$$G(0) = 0, \quad G'(0) = 1, \quad G''(0) = 4 \quad \text{and} \quad G'''(0) < \infty. \tag{8}$$

With the purpose of comparing with other iterative methods, we have chosen two special members of this family using two variants of $G(w_n)$,

$$G(w_n) = kw_n^3 + 2w_n^2 + w_n \quad \text{and} \quad G(w_n) = \frac{w_n}{(1-w_n)^2},$$

denoted by ZCS1 and ZCS2, respectively, where k is a real coefficient. For $k = 0$, we get the methods considered in [19].

In the next section, we focus on method (5), especially on its convergence order, and consecutively on the optimal properties of the method. There we separately analyze cases when the multiplicity m is odd and when it is even. Similarly, in Section 3, we derive some conclusions about optimal properties of method (7). The numerical results and comparison of the methods with concluding remarks are given in Section 4.

2 On the optimality of method (5)

To obtain the optimal fourth order of convergence for iterative method (5), Theorem 1 in [8] gives the sufficient conditions for solving nonlinear equation $f(x) = 0$: if α is a multiple root with multiplicity m ($m > 1$) of sufficiently differentiable function $f : I \to \mathbb{R}$ for some open interval I, and if the initial approximation x_0 is sufficiently close to α, then conditions (6) provide at least fourth convergence order of method (5) with error equation

$$\begin{aligned} e_{n+1} = {} & \frac{1}{6(m-1)^2 m^3}\big(\big(3\big(m^3+8m^2+m+2\big)-(m-1)^2 Q'''(0)\big)c_1^3 \\ & - 6(m-1)m^2 c_1 c_2\big)e_n^4 + O\big(e_n^5\big), \end{aligned} \tag{9}$$

where e_n denotes the error of the nth iteration, i.e. $e_n = x_n - \alpha$, and $c_i = (m!/(m+i)!) \times (f^{(m+i)}(\alpha)/f^{(m)}(\alpha))$ for $i \geqslant 1$. Theorem 1 is proved and the error equation (9) is derived in [8].

On the other hand, we will show that there is a class of nonlinear sufficiently differentiable functions with multiple root α with odd multiplicity m, for which method (5) satisfying the conditions of Theorem 1 does not reach fourth order. Consequently, it means that Theorem 1 does not provide optimality of the method.

Since our remarks are based on the proof of Theorem 1, we recall here some relevant steps of the proof and use similar notation as in [8]. The Taylor's expansion of $f(x_n)$ and $f'(x_n)$ about α yields

$$f(x_n) = \frac{f^{(m)}(\alpha)}{m!}e_n^m\big(1 + c_1 e_n + c_2 e_n^2 + c_3 e_n^3 + O\big(e_n^4\big)\big), \tag{10}$$

$$\begin{aligned} f'(x_n) = {} & \frac{f^{(m)}(\alpha)}{(m-1)!}e_n^{m-1} \\ & \times \left(1 + \frac{m+1}{m}c_1 e_n + \frac{m+2}{m}c_2 e_n^2 + \frac{m+3}{m}c_3 e_n^3 + O\big(e_n^4\big)\right), \end{aligned} \tag{11}$$

$$\begin{aligned} \frac{f(x_n)}{f'(x_n)} = {} & \frac{1}{m}e_n - \frac{c_1}{m^2}e_n^2 + \frac{c_1^2(m+1) - 2mc_2}{m^3}e_n^3 \\ & + \frac{c_1 c_2(3m+4) - 3m^2 c_3 - (m+1)^2 c_1^3}{m^4} + O\big(e_n^5\big), \end{aligned} \tag{12}$$

$$y_n - \alpha = \frac{c_1}{m}e_n^2 - \frac{(m+1)c_1^2 - 2mc_2}{m^2}e_n^3 + \frac{(m+1)^2c_1^3 - (3m+4)mc_1c_2 + 3m^2c_3}{m^3}e_n^4 + O\big(e_n^5\big). \tag{13}$$

From (11) and (13), using Mathematica symbolic computation, it is not difficult to get

$$f'(y_n) = \frac{f^{(m)}(\alpha)}{(m-1)!}e_n^{2(m-1)}\left(\frac{c_1}{m}\right)^{m-1}\Bigg[1 + \frac{m-1}{mc_1}\big(2mc_2 - (m+1)c_1^2\big)e_n + \frac{1}{2m^2c_1^2}\big(2(m+1)c_1^4 + m(m-1)\big((m+1)^2c_1^4 - 2m(2m+1)c_1^2c_2 + 4m(m-2)c_2^2 + 6mc_1c_3\big)e_n^2 + O\big(e_n^3\big)\Bigg],$$

and after dividing by $f'(x_n)$ and simplification, the result is

$$\frac{f'(y_n)}{f'(x_n)} = e_n^{m-1}\left(\frac{c_1}{m}\right)^{m-1}\big(1 + \psi_1 e_n + \psi_2 e_n^2 + O\big(e_n^3\big)\big), \tag{14}$$

where

$$\psi_1 = 2(m-1)\frac{c_2}{c_1} - (m+1)c_1,$$

$$\psi_2 = 3\frac{c_3}{c_1}(m-1) + 2\frac{c_2^2}{c_1^2}(m-2)(m-1) - m(2m+1)c_2 + \frac{(m+1)(m+2)(m^2+1)}{2m^2}c_1^2.$$

Now, since the argument of the function $Q(\cdot)$ is $w_n = \sqrt[m-1]{f'(y_n)/f'(x_n)}$, it is reasonable to calculate w_n using (14) and Taylor's expansion about α. This result is presented in [8] by

$$w_n = \frac{c_1}{m}\left(e_n + \frac{\psi_1}{m-1}e_n^2 - \frac{(m-2)\psi_1^2 - 2(m-1)\psi_2}{2(m-1)^2}e_n^3 + \frac{(m-2)\psi_1((2m-3)\psi_1^2 - 6(m-1)\psi_2}{6(m-1)^3}e_n^4 + O\big(e_n^5\big)\right). \tag{15}$$

Our main remark on the proof of Theorem 1 is related to (15) because it does not hold for every sufficiently differentiable function $f(x)$, even when the initial point x_0 is very close to α. Namely, if the multiplicity m is odd ($m-1$ is even), and if $e_nc_1 < 0$, then w_n must be positive, so then w_n has the following form:

$$w_n = -\frac{c_1}{m}e_n\left(1 + \frac{\psi_1}{m-1}e_n - \frac{(m-2)\psi_1^2 - 2(m-1)\psi_2}{2(m-1)^2}e_n^2 + \frac{(m-2)\psi_1((2m-3)\psi_1^2 - 6(m-1)\psi_2}{6(m-1)^3}e_n^3 + O\big(e_n^4\big)\right). \tag{16}$$

Furthermore, if $Q(w_n)$ is chosen such that it satisfies conditions (6), then from (12), (13) and (16), we have

$$e_{n+1} = \frac{2c_1}{m}e_n^2 + \frac{4m(m-1)c_2 - 2(m^2+2m-1)c_1^2}{m^2(m-1)}e_n^3 + O\big(e_n^4\big).$$

Thus, it is intuitively clear that such iteration step provides just second order of convergence instead fourth.

For further analysis, we rewrite the error equation (9) by

$$e_{n+1} = \frac{1}{6(m-1)^2m^3}c_1A \cdot e_n^4 + O\big(e_n^5\big), \tag{17}$$

where $A = (3(m^3+8m^2+m+2) - (m-1)^2Q'''(0))c_1^2 - 6(m-1)m^2c_2$. When the multiplicity m is odd, the coefficient A plays a crucial role in the convergence behavior of method (5).

For instance, let A be negative and x_0 is chosen very close to α such that $e_0c_1 > 0$ (e_0 is error of the initial approximation). Then, after calculating $(m-1)$st root of (14), we have w_n defined by (15), and the error of the first iteration x_1 can be obtained from (17) as follows:

$$e_1 \approx \frac{1}{6(m-1)^2m^3}c_1A \cdot e_0^4.$$

Since $A < 0$, e_1 has opposite sign to c_1, so in the next iteration step, we get $e_1c_1 < 0$. Thus, $(m-1)$st root of (14) is now defined by (16) instead by (15), and we have

$$e_2 \approx \frac{2c_1}{m}e_1^2 + O\big(e_1^3\big). \tag{18}$$

Clearly, $e_2c_1 > 0$ from (18), and the next iteration calculates w_n by (15), which leads to (9) and, informally speaking, to the step of fourth order.

The alternation of the second- and the fourth-order iteration steps continues until some stopping criterion is satisfied. Therefore, a sequence $\{x_n\}$ produced by method (5) does not converge to α with fourth order of convergence. Consequently, method (5) is not optimal for odd multiplicity m of root α when $A < 0$. This theoretical approach is numerically verified in Section 4 (see Tables 1 and 2).

Nevertheless, Theorem 1 is valid for every even m. Furthermore, method (5) is of fourth order for every odd $m > 1$ if the functions $f(x)$ and $Q(\cdot)$ satisfy $A > 0$.

3 On the optimality of method (7)

The iterative method (7) has been developed in [19], where Theorem 3.1 gives the following conclusion: if α is a multiple root of multiplicity m of sufficiently differentiable function $f : I \to \mathbb{R}$ (I is some open interval), and if the starting point x_0 is close enough to α, then for the function $G(\cdot) \in C^2(\mathbb{R})$ satisfying (8), the convergence order of method defined by (7) is at least four with the error equation

$$e_{n+1} = \frac{1}{6m^3}\big((3m+27-G'''(0))c_1^3 - 6mc_1c_2\big)e_n^4 + O\big(e_n^5\big). \tag{19}$$

According to the Kung–Traub conjecture and Theorem 3.1, method (7) is optimal since it requires one derivative and two function evaluations per iteration. But, similarly to method (5), in the further text, we will show that Theorem 3.1. does not hold for some nonlinear functions with multiple root α with even multiplicity m.

The first step in the iteration scheme (7) is the modified Newton's step, which is the same as in (5). Hence, results (10), (11), (12) and (13) can be used for further analysis. From (10) and (13), we have

$$f(y_n) = \frac{f^{(m)}(\alpha)}{m!} e_n^{2m} \left(\frac{c_1}{m}\right)^m \Bigg[1 + \frac{2mc_2 - (m+1)c_1^2}{c_1} e_n + \frac{1}{2mc_1^2}\Big(\big(m^3 + 3m^2 + 3m + 3\big)c_1^4 - 2m\big(2m^2 + 3m + 2\big)c_1^2 c_2 + 4m^2(m-1)c_2^2 + 6m^2 c_1 c_3 \Big) e_n^2 + O\big(e_n^3\big)\Bigg]. \tag{20}$$

Dividing (20) by (10), we get

$$\frac{f(y_n)}{f(x_n)} = e_n^m \left(\frac{c_1}{m}\right)^m \Bigg[1 + \frac{2mc_2 - (m+2)c_1^2}{c_1} e_n + \frac{1}{2mc_1^2}\Big((m+1)^2(m+3)c_1^4 - 2m\big(2m^2 + 5m + 3\big)c_1^2 c_2 + 4m^2(m-1)c_2^2 + 6m^2 c_1 c_3 \Big) e_n^2 + O\big(e_n^3\big)\Bigg]. \tag{21}$$

For every odd m (including $m = 1$) and for every even m when $c_1 e_n > 0$, the mth root of (21) is obtained by

$$w_n = e_n \frac{c_1}{m}\left[1 + \left(\frac{2c_2}{c_1} - \frac{(m+2)c_1}{m}\right) e_n + \frac{(2m^2 + 7m + 7)c_1^3 - 2m(3m+7)c_1 c_2 + 6m^2 c_3}{2m^2 c_1} e_n^2 + O\big(e_n^3\big)\right]. \tag{22}$$

Thus, using (12), (22) and Taylor's expansion of $G(w_n)$ about 0, the second equation of (7) yields the error equation (19) derived in [19], which can be rearranged as

$$e_{n+1} = \frac{1}{6m^3} c_1 \hat{A} e_n^4 + O\big(e_n^5\big), \tag{23}$$

where $\hat{A} = -6mc_2 + c_1^2(3(m+9) - G'''(0))$.

On the other hand, when the multiplicity m is even and $c_1 e_n < 0$, because w_n has to be positive, we get the mth root of (21) defined by

$$w_n = -e_n \frac{c_1}{m}\left[1 + \left(\frac{2c_2}{c_1} - \frac{(m+2)c_1}{m}\right) e_n + \frac{(2m^2 + 7m + 7)c_1^3 - 2m(3m+7)c_1 c_2 + 6m^2 c_3}{2m^2 c_1} e_n^2 + O\big(e_n^3\big)\right]. \tag{24}$$

Therefore, for $G(w_n)$ satisfying (8), by substituting (13), (21) and (24) into the second step of (7), we get

$$e_{n+1} = \frac{2c_1}{m}e_n^2 + \frac{8mc_2 - 4(m+3)c_1^2}{2m^2}e_n^3 + O\big(e_n^4\big), \tag{25}$$

which clearly represents the iteration step of the second order.

When the multiplicity of root α is even, the convergence behavior of method (7) is strongly dependent on the corresponding coefficient $\hat{A}$ from (23). If $\hat{A} > 0$, the fourth convergence order remains. If $\hat{A} < 0$, it is easy to show in the same fashion as in previous section, that method (7) implies the alternation of the second- and the fourth-order iteration steps with the errors given by (25) and (23), respectively, and the method loses optimal properties. The numerical examples, which confirm this theoretical analysis, are given in the next section.

4 Numerical results

All numerical computations presented in this section have been carried out by Mathematica with the aid of $\mathrm{SetPrecision}$ function with 10000 significant digits, on a personal computer with 32-bit Windows Vista operating system and 1.73 GHz processor speed. The stopping criterion for all test examples has been $|f(x_n)| < 10^{-200}$, where the number of iterations does not exceed 100.

With the aim to consider the convergence properties of the method LZ1 (5) for the roots with odd multiplicity, in Table 1 and Table 2, we present the numerical results for various choices of coefficient k. Although conditions (6) for the function $Q(w_n)$ are satisfied for every real k, it is not enough to provide the fourth convergence order of LZ1. On the left side of the tables, we have shown the results for the choice of k, where $A < 0$, while the results for k where $A > 0$ are on the right side. It is easy to obtain that the variants of LZ1, where $A < 0$ require more iterations to satisfy the stopping criterion, and that they have slower convergence compared with the right side competitors. More importantly, the convergence behavior can be numerically checked by considering exponents of 10 in $x_i - \alpha$ or $|f(x_i)|$ columns, especially for iterations $\{x_i\}$ very close to solution α. It is well-known that if p is the order of convergence of some method, then the method approximately multiplies by p the number of exact decimals after every iteration in some very close neighborhood of α, i.e. the number of zeros after decimal point multiplies by p after each iteration. Obviously, by dividing the exponents of $x_i - \alpha$ or $|f(x_i)|$, the right-hand side methods have results approximately equal four. On the other hand, by dividing the exponents of the left-side methods, we get alternation of the two results, two and four, which ruin optimal properties of the method (5) for odd m when $A < 0$. In these cases, the sequence $\{x_i\}$ oscillates about α until the stopping criterion is satisfied. These test examples clearly confirm the theoretical analysis from Section 2. Tables 3 and 4 show similar behavior of the method ZCS1 (7) when the multiplicity of the root α is even for different sign of the coefficient $\hat{A}$.

Table 1. Numerical results for $f(x) = x^3(x-1)^2$, $\alpha = 0$, $m = 3$ and $x_0 = -0.5$.

i	$x_i - \alpha$	$\lvert f(x_i)\rvert$	$x_i - \alpha$	$\lvert f(x_i)\rvert$
	Method LZ1 for $k = 15$ ($A = -300$)		Method LZ1 for $k = 1$ ($A = 1044$)	
1	-0.010718	$1.2578 \cdot 10^{-6}$	-0.021346	$1.1015 \cdot 10^{-5}$
2	$9.6869 \cdot 10^{-9}$	$9.0898 \cdot 10^{-25}$	$-5.8496 \cdot 10^{-7}$	$2.0016 \cdot 10^{-19}$
3	$-1.2511 \cdot 10^{-16}$	$1.9585 \cdot 10^{-48}$	$-3.7728 \cdot 10^{-25}$	$5.3702 \cdot 10^{-74}$
4	$2.2689 \cdot 10^{-64}$	$1.1679 \cdot 10^{-191}$	$-6.5284 \cdot 10^{-98}$	$2.7824 \cdot 10^{-292}$
5	$-6.8636 \cdot 10^{-128}$	$3.2334 \cdot 10^{-382}$		

Table 2. Numerical results for $f(x) = (x^2 - \mathrm{e}^x - 3x + 2)^5$, $\alpha = 0.25753\ldots$, $m = 5$ and $x_0 = 1.8$.

i	$x_i - \alpha$	$\lvert f(x_i)\rvert$	$x_i - \alpha$	$\lvert f(x_i)\rvert$
	Method LZ1 for $k = 0$ ($A \approx -6.12$)		Method LZ1 for $k = -1$ ($A \approx 14.84$)	
1	0.025738	$8.5990 \cdot 10^{-6}$	0.037103	$5.3260 \cdot 10^{-5}$
2	$-1.2117 \cdot 10^{-4}$	$2.0121 \cdot 10^{-17}$	$-2.4923 \cdot 10^{-4}$	$7.4089 \cdot 10^{-16}$
3	$5.0974 \cdot 10^{-20}$	$2.6512 \cdot 10^{-94}$	$-2.2463 \cdot 10^{-18}$	$4.4060 \cdot 10^{-86}$
4	$-4.8566 \cdot 10^{-40}$	$2.0814 \cdot 10^{-194}$	$-1.4714 \cdot 10^{-74}$	$5.3129 \cdot 10^{-367}$
5	$1.3258 \cdot 10^{-161}$	$3.1559 \cdot 10^{-802}$		

Table 3. Numerical results for $f(x) = x^3(x-1)^2$, $\alpha = 1$, $m = 2$ and $x_0 = 1.75$.

i	$x_i - \alpha$	$\lvert f(x_i)\rvert$	$x_i - \alpha$	$\lvert f(x_i)\rvert$
	Method ZCS1 for $k = 10$ ($\hat{A} = -297$)		Method ZCS1 for $k = 2$ ($\hat{A} = 153$)	
1	0.05332	0.0033229	0.1095	0.016367
2	$-5.789 \cdot 10^{-5}$	$3.3607 \cdot 10^{-9}$	$6.764 \cdot 10^{-4}$	$4.5847 \cdot 10^{-7}$
3	$1.009 \cdot 10^{-8}$	$1.0175 \cdot 10^{-16}$	$1.993 \cdot 10^{-12}$	$3.9713 \cdot 10^{-24}$
4	$-1.805 \cdot 10^{-31}$	$3.2590 \cdot 10^{-62}$	$1.508 \cdot 10^{-46}$	$2.2744 \cdot 10^{-92}$
5	$9.777 \cdot 10^{-62}$	$9.5590 \cdot 10^{-123}$	$4.947 \cdot 10^{-183}$	$2.4469 \cdot 10^{-365}$
6	$-1.593 \cdot 10^{-243}$	$2.5387 \cdot 10^{-486}$		

Table 4. Numerical results for $f(x) = (x^2 - \mathrm{e}^x - 3x + 2)^4$, $\alpha = 0.25753\ldots$, $m = 4$ and $x_0 = 2$.

i	$x_i - \alpha$	$\lvert f(x_i)\rvert$	$x_i - \alpha$	$\lvert f(x_i)\rvert$
	Method ZCS1 for $k = 0$ ($\hat{A} \approx -1.29$)		Method ZCS1 for $k = -2$ ($\hat{A} \approx 0.39$)	
1	0.055831	0.0019412	0.11229	0.031168
2	$-5.5293 \cdot 10^{-4}$	$1.9060 \cdot 10^{-11}$	$-2.0972 \cdot 10^{-3}$	$3.9467 \cdot 10^{-9}$
3	$1.1652 \cdot 10^{-16}$	$3.7575 \cdot 10^{-62}$	$-7.9249 \cdot 10^{-15}$	$8.0414 \cdot 10^{-55}$
4	$-2.5375 \cdot 10^{-33}$	$8.4521 \cdot 10^{-129}$	$-1.5016 \cdot 10^{-60}$	$1.0365 \cdot 10^{-237}$
5	$5.1894 \cdot 10^{-134}$	$1.4785 \cdot 10^{-531}$		

The test examples from [18] and [14] given in Table 4 have been used for further analysis. For the functions where the exact root α is not available, we have used the approximation calculated with 10000 digits (only 7 digits are displayed).

Numerical results for all relevant fourth-order previously presented methods, together with the second-order modified Newton's method (2) denoted by MNM, are given in Tables 6–11, with the purpose of comparing with the results of the methods LZ1 (for $k = 0$), LZ2, ZCS1 (for $k = 0$) and ZCS2. Each table lists the number of iterations for

Table 5. Test functions.

$f(x)$	α	m	x_0
$f_1(x) = (x^2 - \mathrm{e}^x - 3x + 2)^5$	$0.2575302\ldots$	5	1.8
$f_2(x) = (\cos x - x)^3$	$0.7390851\ldots$	3	2.5
$f_3(x) = (\log x + \sqrt{x}/x^2 - 1)^3$	$1.7910672\ldots$	3	1.95
$f_4(x) = (2x + \mathrm{e}^{-x} + \sin(x^2) - 3)^5$	$0.9244631\ldots$	5	0.75
$f_5(x) = (\mathrm{e}^x + x - 20)^4$	$2.8424389\ldots$	4	3
$f_6(x) = (x^{10} - \sqrt{3}x^3\cos(x\pi/6) + 1/(x^2+1))(x-1)^5$	1.0000000	6	1.08

Table 6. Numerical results for $f_1(x)$.

Method	it	$\lvert x_3 - \alpha\rvert$	$\lvert f(x_3)\rvert$	CPU	COC
MNM	6	$4.2743 \cdot 10^{-6}$	$1.0991 \cdot 10^{-24}$	0.890	2.0000
LLC	4	$3.3967 \cdot 10^{-22}$	$3.4830 \cdot 10^{-105}$	0.943	4.0000
ShSh	4	$1.4137 \cdot 10^{-22}$	$4.3495 \cdot 10^{-107}$	0.953	4.0000
LCN	4	$3.3967 \cdot 10^{-22}$	$3.4830 \cdot 10^{-105}$	0.922	4.0000
ZCS	4	$6.8563 \cdot 10^{-23}$	$1.1672 \cdot 10^{-108}$	0.950	4.0000
RK1	4	$4.5104 \cdot 10^{-16}$	$1.4380 \cdot 10^{-74}$	0.930	4.0000
RK2	4	$1.8611 \cdot 10^{-15}$	$1.7200 \cdot 10^{-71}$	0.967	4.0000
LZ1	5	$5.0974 \cdot 10^{-20}$	$2.6512 \cdot 10^{-94}$	1.38	6.0718
LZ2	4	$3.5201 \cdot 10^{-13}$	$4.1636 \cdot 10^{-60}$	1.06	6.0133
ZCS1	3	$4.2578 \cdot 10^{-60}$	$1.0780 \cdot 10^{-294}$	0.507	4.0004
ZCS2	4	$6.8013 \cdot 10^{-32}$	$1.1211 \cdot 10^{-153}$	0.867	4.0000

Table 7. Numerical results for $f_2(x)$.

Method	it	$\lvert x_3 - \alpha\rvert$	$\lvert f(x_3)\rvert$	CPU	COC
MNM	7	0.00016723	$2.1924 \cdot 10^{-11}$	2.90	2.0000
LLC	5	$4.2258 \cdot 10^{-15}$	$3.5375 \cdot 10^{-43}$	3.59	4.0000
ShSh	5	$4.9948 \cdot 10^{-15}$	$5.8414 \cdot 10^{-43}$	3.60	4.0000
LCN	5	$4.2258 \cdot 10^{-15}$	$3.5375 \cdot 10^{-43}$	3.54	4.0000
ZCS	5	$6.7304 \cdot 10^{-15}$	$1.4292 \cdot 10^{-42}$	3.60	4.0000
RK1	–	–	–	–	–
RK2	–	–	–	–	–
LZ1	7	0.18489	0.025967	5.31	4.0000
LZ2	6	0.17199	0.026513	4.49	4.0000
ZCS1	4	$1.4633 \cdot 10^{-22}$	$1.4689 \cdot 10^{-65}$	2.06	4.0000
ZCS2	4	$2.2723 \cdot 10^{-25}$	$5.5003 \cdot 10^{-74}$	2.26	4.0000

corresponding method required to satisfy the stopping criterion. The tables also display errors $|x_i - \alpha|$ and residual errors $|f(x_i)|$ at the same level of iterative process for all methods (in most cases there are errors and residual errors after third iteration). For all test examples we have run each method 25 times and the average time is given in fifth column (CPU). To numerically check the convergence order, the computational order of convergence (COC) has been calculated by

$$\mathrm{COC} = \frac{\log|(x_n - \alpha)/(x_{n-1} - \alpha)|}{\log|(x_{n-1} - \alpha)/(x_{n-2} - \alpha)|}. \tag{26}$$

If the method exceeds 100 iterations or diverges, it is denoted by "–".

In Tables 6–9, we have tested the examples where the multiplicity m is odd. From those tables, by observing the errors and residual errors, it is easy to see that both variants

Table 8. Numerical results for $f_3(x)$.

Method	it	$\lvert x_3-\alpha\rvert$	$\lvert f(x_3)\rvert$	CPU	COC
MNM	6	$1.4277\cdot 10^{-10}$	$2.6545\cdot 10^{-32}$	0.653	2.0000
LLC	4	$2.2008\cdot 10^{-60}$	$9.7229\cdot 10^{-182}$	0.677	4.0000
ShSh	4	$2.7396\cdot 10^{-60}$	$1.8755\cdot 10^{-181}$	0.688	4.0000
LCN	4	$2.2008\cdot 10^{-60}$	$9.7229\cdot 10^{-182}$	0.669	4.0000
ZCS	4	$3.9855\cdot 10^{-60}$	$5.7740\cdot 10^{-181}$	0.685	4.0000
RK1	4	$4.6859\cdot 10^{-64}$	$9.3847\cdot 10^{-193}$	0.693	4.0000
RK2	4	$2.0615\cdot 10^{-64}$	$7.9904\cdot 10^{-194}$	0.716	4.0000
LZ1	4	$3.7256\cdot 10^{-59}$	$4.7166\cdot 10^{-178}$	0.696	4.0000
LZ2	4	$1.3035\cdot 10^{-64}$	$2.0199\cdot 10^{-194}$	0.704	4.0000
ZCS1	4	$9.4430\cdot 10^{-65}$	$7.6802\cdot 10^{-195}$	0.678	4.0000
ZCS2	3	$6.6386\cdot 10^{-68}$	$2.6685\cdot 10^{-204}$	0.499	4.0000

Table 9. Numerical results for $f_4(x)$.

Method	it	$\lvert x_3-\alpha\rvert$	$\lvert f(x_3)\rvert$	CPU	COC
MNM	5	$1.0818\cdot 10^{-13}$	$2.6289\cdot 10^{-63}$	2.84	2.0000
LLC	3	$4.9466\cdot 10^{-72}$	$5.2541\cdot 10^{-355}$	2.62	4.0000
ShSh	3	$4.8957\cdot 10^{-72}$	$4.9896\cdot 10^{-355}$	2.64	4.0000
LCN	3	$4.9466\cdot 10^{-72}$	$5.2541\cdot 10^{-355}$	2.60	4.0000
ZCS	3	$4.8470\cdot 10^{-72}$	$4.7461\cdot 10^{-355}$	2.63	4.0000
RK1	3	$2.3530\cdot 10^{-64}$	$1.2797\cdot 10^{-316}$	2.65	4.0000
RK2	3	$2.3646\cdot 10^{-64}$	$1.3114\cdot 10^{-316}$	2.66	4.0000
LZ1	4	$4.6441\cdot 10^{-22}$	$3.8326\cdot 10^{-105}$	4.03	4.0000
LZ2	4	$4.0581\cdot 10^{-22}$	$1.9526\cdot 10^{-105}$	4.04	4.0000
ZCS1	3	$1.2254\cdot 10^{-74}$	$4.9011\cdot 10^{-368}$	2.15	4.0000
ZCS2	3	$5.2477\cdot 10^{-75}$	$7.0607\cdot 10^{-370}$	2.24	4.0000

Table 10. Numerical results for $f_5(x)$.

Method	it	$\lvert x_3-\alpha\rvert$	$\lvert f(x_3)\rvert$	CPU	COC
MNM	6	$1.6650\cdot 10^{-9}$	$8.3529\cdot 10^{-31}$	0.996	2.0000
LLC	3	$1.2065\cdot 10^{-71}$	$2.3030\cdot 10^{-279}$	0.809	4.0000
ShSh	3	$8.2939\cdot 10^{-71}$	$5.1435\cdot 10^{-276}$	0.819	4.0000
LCN	3	$1.2065\cdot 10^{-71}$	$2.3030\cdot 10^{-279}$	0.802	4.0000
ZCS	3	$6.8692\cdot 10^{-70}$	$2.4203\cdot 10^{-272}$	0.818	4.0000
RK1	3	$3.6965\cdot 10^{-78}$	$2.0296\cdot 10^{-305}$	0.824	4.0000
RK2	3	$6.1394\cdot 10^{-83}$	$1.5443\cdot 10^{-324}$	0.837	4.0000
LZ1	3	$7.6842\cdot 10^{-56}$	$3.7899\cdot 10^{-216}$	0.829	4.0000
LZ2	3	$1.9590\cdot 10^{-73}$	$1.6008\cdot 10^{-286}$	0.836	4.0000
ZCS1	3	$4.4084\cdot 10^{-61}$	$4.1055\cdot 10^{-237}$	0.660	4.0000
ZCS2	3	$5.1501\cdot 10^{-71}$	$7.6473\cdot 10^{-277}$	0.655	4.0000

of the method (7) have faster convergence to the exact root α. On the other hand, for some test examples such as $f_1(x)$ and $f_2(x)$ (Tables 6 and 7), method (7) requires less CPU time for root-finding. In Tables 10 and 11, we present the numerical results for the functions where multiplicities of the roots are even. Again, the new methods ZCS1 and ZCS2 produce relatively good numerical results, especially for CPU time.

In every test example from Table 5, the methods ZCS1 and ZCS2 reach optimal fourth order of convergence, even when $m = 4$ or $m = 6$. For the later cases when the multiplicity is even, the optimal order is achieved because $\hat{A}$ is positive. Note that

Table 11. Numerical results for $f_6(x)$.

Method	it	$\lvert x_3-\alpha\rvert$	$\lvert f(x_3)\rvert$	CPU	COC
MNM	6	$4.0588\cdot 10^{-9}$	$2.4380\cdot 10^{-50}$	3.64	2.0000
LLC	3	$2.5592\cdot 10^{-55}$	$1.5320\cdot 10^{-327}$	3.09	3.9999
ShSh	3	$3.5306\cdot 10^{-55}$	$1.0563\cdot 10^{-326}$	3.08	3.9999
LCN	3	$2.5592\cdot 10^{-55}$	$1.5320\cdot 10^{-327}$	3.06	3.9999
ZCS	3	$4.4903\cdot 10^{-55}$	$4.4701\cdot 10^{-326}$	3.08	3.9999
RK1	3	$3.7429\cdot 10^{-55}$	$1.4993\cdot 10^{-326}$	3.10	3.9999
RK2	3	$2.1744\cdot 10^{-55}$	$5.7643\cdot 10^{-328}$	3.11	3.9999
LZ1	3	$3.6869\cdot 10^{-51}$	$1.3697\cdot 10^{-302}$	3.14	3.9998
LZ2	3	$1.4482\cdot 10^{-62}$	$5.0302\cdot 10^{-371}$	3.14	4.0000
ZCS1	3	$4.3113\cdot 10^{-54}$	$3.5019\cdot 10^{-320}$	2.09	3.9999
ZCS2	3	$9.6919\cdot 10^{-61}$	$4.5197\cdot 10^{-360}$	2.12	4.0000

for $f_1(x)$ methods LZ1 and LZ2 have COC values approximately 6. It is not difficult to show that for stopping criterion $|f(x_n)| < 10^{-300}$, method LZ2 requires one more iteration, and COC value is then approximately 1.3. This happens because m is odd and the corresponding coefficients A are negative. Similarly, evaluating COC from (26) for method (7) when m is even and corresponding $\hat{A} < 0$, leads to the same COC values.

This is the main drawback of methods (5) and (7). One possible way to overcome this difficulty is to choose sufficiently good functions $Q(w_n)$ and $G(w_n)$ (for example, for each negative k with sufficiently large absolute value, ZCS1 method gives positive coefficient $\hat{A}$ due to $G'''(0) = 6k$). Nevertheless, the numerical results show that these methods are very competitive. According to the CPU time, evaluation of the mth root in every iteration does not significantly increase computational cost, which agrees with the conclusions of the paper [11]. Developing higher order schemes based on these methods could be worth of investigation.

References

1. C. Dong, A basic theorem of constructing an iterative formula of the higher order for computing multiple roots of an equation, *Math. Numer. Sin.*, **11**:445–450, 1982.
2. C. Dong, A family of multipoint iterative functions for finding multiple roots of equations, *Int. J. Comput. Math.*, **21**:363–367, 1987.
3. Y.H. Geum, Y.I. Kim, Cubic convergence of parameter-controlled newton-secant method for multiple zeros, *Journal of Computational and Applied Mathematics*, **233**:931–937, 2009.
4. P. Jarratt, Mulitipoint iterative methods for solving certain equations, *Comput. J.*, **8**:398–400, 1966.
5. Y.I. Kim, Y.H. Geum, A cubic-order variant of newton's method for finding multiple roots of nonlinear equations, *Computers and Mathematics with Applications*, **62**:1634–1640, 2011.
6. S. Li, L. Cheng, B. Neta, Some fourth-order nonlinear solvers with closed formulae for multiple roots, *Comput. Math. Appl.*, **59**:126–135, 2010.
7. S. Li, X. Liao, L. Cheng, A new fourth-order iterative method for finding multiple roots of nonlinear equations, *Appl. Math. Comput.*, **215**:1288–1292, 2009.

8. B. Liu, X. Zhou, A new family of fourth-order methods for multiple roots of nonlinear equations, *Nonlinear Anal. Model. Control*, **18**(2):143–152, 2013.

9. B. Neta, New third order nonlinear solvers for multiple roots, *Appl. Math. Comput.*, **202**:162–170, 2008.

10. B. Neta, Extension of murakami's high order nonlinear solver to multiple roots, *Int. J. Comput. Math.*, **87**:1023–1031, 2010.

11. B. Neta, C. Chun, M. Scott, On the development of iterative methods for multiple roots, *Appl. Math. Comput.*, **224**:358–361, 2013.

12. B. Neta, A.N. Johnson, High-order nonlinear solver for multiple roots, *Comput. Math. Appl.*, **55**:2012–2017, 2008.

13. N. Osada, An optimal multiple root-finding method of order three, *J. Comput. Appl. Math.*, **51**:131–133, 1994.

14. M.S. Rhee, Y.I. Kim, A general class of optimal fourth-order multiple-root finders without memory for nonlinear equations, *Appl. Math. Sci.*, **111**:5537–5551, 2013.

15. D. Sbibih, A. Serghini, A. Tijini, A. Zidna, A general family of third order method for finding multiple roots, *Appl. Math. Comput.*, **233**:338–350, 2014.

16. J.R. Sharma, R. Sharma, Modified jarratt method for computing multiple roots, *Appl. Math. Comput.*, **217**:878–881, 2010.

17. H.D. Victory, B. Neta, A higher order method for multiple zeros of nonlinear functions, *Int. J. Comput. Math.*, **12**:329–335, 1983.

18. X. Zhou, X. Chen, Y. Song, Constructing higher-order methods for obtaining the multiple roots of nonlinear equations, *J. Comput. Appl. Math.*, **235**:4199–4206, 2011.

19. X. Zhou, X. Chen, Y. Song, Families of third and fourth order methods for multiple roots of nonlinear equations, *Appl. Math. Comput.*, **219**:6030–6038, 2013.