Algorithms **2015**, *8*, 656-668; doi:10.3390/a8030656

OPEN ACCESS

algorithms

ISSN 1999-4893

www.mdpi.com/journal/algorithms

Article

Fifth-Order Iterative Method for Solving Multiple Roots of the Highest Multiplicity of Nonlinear Equation

Juan Liang [1,†], Xiaowu Li [2,†,*], Zhinan Wu [3,†], Mingsheng Zhang [2,†,*], Lin Wang [2,†,*] and Feng Pan [2,†,*]

[1] Department of Science, Taiyuan Institute of Technology, Taiyuan 030008, China;
E-Mail: liangjuan76@126.com

[2] College of Information Engineering, Guizhou Minzu University, Guiyang 550025, China

[3] School of Mathematics and Computer Science, Yichun University, Yichun 336000, China;
E-Mail: zhi_nan_7@163.com

† These authors contributed equally to this work.

* Authors to whom correspondence should be addressed; E-Mails: lixiaowu002@126.com (X.L.); zhangmscomputer@163.com (M.Z.); wanglin@gzmu.edu.cn (L.W.); panf@vip.163.com (F.P.); Tel.: +86-18786132431 (X.L.); +86-138-8511-0635 (M.Z.); +86-139-8552-0012 (L.W.); +86-139-8400-1687 (F.P.).

Academic Editor: Alicia Cordero

Received: 8 June 2015 / Accepted: 14 August 2015 / Published: 20 August 2015

Abstract: A three-step iterative method with fifth-order convergence as a new modification of Newton's method was presented. This method is for finding multiple roots of nonlinear equation with unknown multiplicity m whose multiplicity m is the highest multiplicity. Its order of convergence is analyzed and proved. Results for some numerical examples show the efficiency of the new method.

Keywords: nonlinear equation; multiple roots; newton-like method; high-order convergence; iterative methods

1. Introduction

This paper addresses the problem of multiple roots x^* of nonlinear equation $f(x) = 0$ with unknown multiplicity m whose multiplicity m is the highest multiplicity, where $f : [a, b] \subset R \to R$ is a nonlinear differential function on $[a, b]$. In case the multiplicity m is given explicitly, there are many iterative methods established via various techniques (see [1–15] for more details). If the multiplicity m is not known explicitly, Traub [16] utilized a simple transformation $F(x) = f(x)/f'(x)$ instead of $f(x)$ for computing a multiple root of $f(x) = 0$. In this case, the aim of solving a multiple root is reduced to that of solving a simple root of the transformed equation $f(x) = 0$, and thus any iterative method can be used to preserve the original convergence order. However, Newton's method for this transformed equation requires evaluations of the derivatives $f'(x)$ and $f''(x)$. In order to avoid the evaluations of these derivatives with the multiplicity m unknown, for multiple roots, King [17] proposed the secant method which does not use the function $F = f/f'$, but rather use $F = \dfrac{f(x)}{\dfrac{f(x-f(x))-f(x)}{(x-f(x))-x}} = \dfrac{-f^2(x)}{f(x-f(x))-f(x)}$. Wu and Fu [18] further used $F(x) = \dfrac{f^2(x)}{f(x)-f(x-f(x))}$ and transformed the problem of solving multiple roots of $f(x) = 0$ into that of solving simple root of $f(x) = 0$. Actually, they established the following iteration formulae:

$$x_{n+1} = x_n - \frac{F^2(x_n)}{p \cdot F^2(x_n) + F(x_n) - F(x_n - F(x_n))}, \tag{1}$$

where $p \in R$, $|p| < \infty$. So, the sequence $\{x_n\}$ produced by the iteration Formulae (1) is at least quadratically convergent for multiple roots. Moreover, Wu *et al.* [19] defined a function

$$F(x) = \frac{sign(f(x))f(x)|f(x)|^{1/m}}{sign(f(x+sign(f(x))|f(x)|^{1/m})-f(x))f(x)|f(x)|^{1/m}+f(x+sign(f(x))|f(x)|^{1/m})-f(x)}, \tag{2}$$

where m is the multiplicity, and employed the modified Steffensen's method (see [20,21])

$$\begin{aligned} x_{n+1} =& x_n - h_n \frac{F^2(x_n)}{t \cdot F^2(x_n) + F(x_n + F(x_n)) - F(x_n)} \\ =& x_n - h_n \frac{F(x_n)}{t \cdot F(x_n) + (F(x_n + F(x_n)) - F(x_n))/(F(x_n))} \end{aligned} \tag{3}$$

to compute the approximate solution of the equation $f(x) = 0$, where $h_n (> 0)$ is the step size of iteration and $|t| < \infty$. Parida and Gupta [22] suggested another transformation

$$F(x) = \begin{cases} \dfrac{f^2(x)}{\delta + f(x+f(x)) - f(x)} & \text{if } f(x) \neq 0, \\ 0 & \text{if } f(x) = 0\ . \end{cases} \tag{4}$$

where $\delta = sign(f(x + f(x)) - f(x))f^2(x)$, and transform the task of solving multiple zeros of f into that of solving simple zero of F. In this case, they utilized a quadratically convergent derivative free Newton-like iterative method:

$$x_{n+1} = x_n - \frac{F^2(x_n)}{p \cdot F^2(x_n) + F(x_n) - F(x_n - F(x_n))} \tag{5}$$

where the parameter p should be chosen such that the denominator is the largest in magnitude. Yun [23] suggested a new transformation of $f(x)$ as

$$H_\varepsilon(x) = \frac{\varepsilon f^2(x)}{f(x+\varepsilon f(x)) - f(x)}, \tag{6}$$

took ε such that $\max\limits_{a\le x\le b}|\varepsilon f(x)|=\delta$, that is

$$\varepsilon=\frac{\delta}{\max_{a\le x\le b}|f(x)|}=\frac{\delta}{\max\{|f(a)|,|f(b)|\}},$$

and proposed an iterative method as follows:

$$x_{n+1}=x_n-\frac{2(x_n-x_{n-1})H_\varepsilon(x_n)}{H_\varepsilon(2x_n-x_{n-1})-H_\varepsilon(x_{n-1})}. \tag{7}$$

Recently, for the transformed equation $K(x)=0$ with a simple root, Yun [24] proposed a Steffensen-type iterative formula

$$p_{k+1}=p_k-\frac{\mu K(p_k)^2}{K(p_k+\mu K(p_k))-K(p_k)},\quad k\ge 0, \tag{8}$$

where $K(x)=K(\varepsilon;x)=\begin{cases}\frac{\varepsilon f(x)^2}{f(x+\varepsilon f(x))-f(x)}, \text{if } f(x)\neq 0,\\ 0, \text{if } f(x)=0.\end{cases}$

In this paper we construct a new modified Newton's method. We will present the proof that the method is three-step iterative method with fifth-order convergence for nonlinear equations of multiple roots with unknown multiplicity m, whose multiplicity m is the highest multiplicity and without requiring the use of the second derivative.

2. Iterative Method with Fifth-Order Convergence for Solving Multiple Roots

We consider the simple transformation (see [16,25]):

$$F(x)=\begin{cases}\dfrac{f(x)}{f'(x)}, & \text{if } f(x)\neq 0,\\ 0, & \text{if } f(x)=0,\end{cases} \tag{9}$$

and use a Newton-like iterative method:

$$\begin{cases} y_n=x_n-\dfrac{F(x_n)}{F'(x_n)},\\ z_n=y_n-\dfrac{F(y_n)}{F'(y_n)},\\ x_{n+1}=z_n-\dfrac{F(z_n)}{F'(z_n)}.\end{cases} \tag{10}$$

In order to avoid computing the first derivatives of function $F(x_n)$, $F(y_n)$ and $F(z_n)$, we approximate them as follows:

$$F'(x_n)\approx\frac{F(x_n+F(x_n))-F(x_n)}{F(x_n)}=g_1(x_n), \tag{11}$$

$$F'(y_n)\approx\frac{2(F(y_n)-F(x_n))}{y_n-x_n}-g_1(x_n)=g_2(x_n) \tag{12}$$

$$\begin{aligned} F'(z_n)\approx& F[z_n,y_n]+F[z_n,x_n,x_n](z_n-y_n)\\ =&\frac{F(z_n)-F(y_n)}{z_n-y_n}+\frac{\dfrac{F(z_n)-F(x_n)}{z_n-x_n}-g_1(x_n)}{z_n-x_n}(z_n-y_n)\\ =&g_3(x_n)\end{aligned} \tag{13}$$

(See [25–28] for the detail discussions of Equations (11)–(13) respectively.) Substituting the approximations of $F'(x_n)$, $F'(y_n)$ and $F'(z_n)$ given by Equations (11)–(13) in Equation (10), we establish the following new iterative method:

$$\begin{cases} y_n = x_n - \dfrac{F(x_n)}{g_1(x_n)}, \\ z_n = y_n - \dfrac{F(y_n)}{g_2(x_n)}, \\ x_{n+1} = z_n - \dfrac{F(z_n)}{g_3(x_n)}. \end{cases} \tag{14}$$

We give the following convergence theorem for the proposed method Equation (14) as follows.

Theorem 1. *Suppose that $f \in C^1(D)(D \subseteq R \to R)$ has multiple roots $x^* \in D$, whose multiplicity m is the highest multiplicity of nonlinear equation, where D is an open interval. If the initial point x_0 is sufficiently close to x^*, the iterative method defined by (14) has fifth-order convergence.*

Proof. Without loss of generality, we assume that $f(x)$ has two multiple roots

$$f(x) = (x - x^*)^m (x - x_1)^n h(x) \tag{15}$$

where x^* is a multiple root of Equation (15) with multiplicity m and x_1 is a multiple root of Equation (15) with multiplicity n $(m > n)$, $h(x)$ is a continuous function with $h(x^*) \neq 0$ and $h(x_1) \neq 0$. According to Equation (15), we have

$$f'(x) = m(x - x^*)^{m-1}(x - x_1)^n h(x) + n(x - x^*)^m (x - x_1)^{n-1} h(x) + (x - x^*)^m (x - x_1)^n h'(x). \tag{16}$$

□

Dividing Equation (15) by Equation (16), we get

$$F(x) = \frac{f(x)}{f'(x)} = \frac{(x - x^*)(x - x_1)h(x)}{m(x - x_1)h(x) + n(x - x^*)h(x) + (x - x^*)(x - x_1)h'(x)} \tag{17}$$

From Equation (17), we can see that the problem of computing multiple roots of $f(x) = 0$ can be reduced to the equivalent problem of computing simple root x^* of $F(x) = 0$.

Using Taylor's expansion, we have

$$h(x_n) = h(x^*)[1 + b_1 e_n + b_2 e_n^2 + b_3 e_n^3 + b_4 e_n^4 + b_5 e_n^5 + b_6 e_n^6 + o(e_n^7)] \tag{18}$$

where $b_k = \dfrac{h^{(k)}(x^*)}{k! h(x^*)}, k = 1, 2, ...,$ and $e_n = x_n - x^*$.

By Equation (18), we obtain

$$h'(x_n) = h(x^*)[b_1 + 2b_2 e_n + 3b_3 e_n^2 + 4b_4 e_n^3 + 5b_5 e_n^4 + 6b_6 e_n^5 + o(e_n^6)]. \tag{19}$$

Substituting Equations (18) and (19) into Equation (17), we get

$$F(x_n) = \frac{e_n h(x_n)}{m h(x_n) + e_n h'(x_n)} = c_1 e_n + c_2 e_n^2 + c_3 e_n^3 + c_4 e_n^4 + o(e_n^5) \tag{20}$$

where

$$c_1 = \frac{1}{m}, c_2 = -\frac{(a-b)b_1 + n}{m^2(a-b)} \tag{21}$$

$$\begin{aligned} c_3 =& \frac{1}{(a-b)^2 m^3}((a-b)^2 b_1^2 m + a^2 b_1^2 - 2a^2 b_2 m - 2abb_1^2 + 4abb_2 m \\ &+ b^2 b_1^2 - 2b^2 b_2 m + 2ab_1 n - 2bb_1 n + mn + n^2) \end{aligned} \tag{22}$$

$$\begin{aligned} c_4 =& -\frac{1}{m^4(a-b)^3}(a^3 b_1^3 m^2 - 3a^2 bb_1^3 m^2 + 3ab^2 b_1^3 m^2 - b^3 b_1^3 m^2 + 2a^3 b_1^3 m \\ &- 3a^3 b_1 b_2 m^2 - 6a^2 bb_1^3 m + 9a^2 bb_1 b_2 m^2 + 6ab^2 b_1^3 m - 9ab^2 b_1 b_2 m^2 \\ &- 2b^3 b_1^3 m + 3b^3 b_1 b_2 m^2 + a^3 b_1^3 - 4a^3 b_1 b_2 m + 3a^3 b_3 m^2 - 3a^2 bb_1^3 \\ &+ 12a^2 bb_1 b_2 m - 9a^2 bb_3 m^2 + 2a^2 b_1^2 mn + 3ab^2 b_1^3 - 12ab^2 b_1 b_2 m \\ &+ 9ab^2 b_3 m^2 - 4abb_1^2 mn - b^3 b_1^3 + 4b^3 b_1 b_2 m - 3b^3 b_3 m^2 + 2b^2 b_1^2 mn \\ &+ 3a^2 b_1^2 n - 4a^2 b_2 mn - 6abb_1^2 n + 8abb_2 mn + 3b^2 b_1^2 n - 4b^2 b_2 mn \\ &+ 2ab_1 mn + 3ab_1 n^2 - 2bb_1 mn - 3bb_1 n^2 + m^2 n + 2mn^2 + n^3) \end{aligned} \tag{23}$$

Substituting Equation (20) into Equation (11), we obtain

$$\begin{aligned} g_1(x_n) =& c_1 + c_2(2 + c_1)e_n + (3c_3 + 3c_1c_3 + c_1^2 c_3 + c_2^2)e_n^2 + (4c_2c_3 + 2c_1c_2c_3 + 4c_4 \\ &+ 6c_1c_4 + 4c_1^2 c_4 + c_1^3 c_4)e_n^3 + (5c_5 + 10c_1c_5 + 5c_1^3 c_5 + 10c_1^2 c_5 + 2c_3^2 c_1 + c_2^2 c_3 \\ &+ 7c_2c_4 + c_1^4 c_5 + 3c_3^2 + 3c_1^2 c_2 c_4 + 8c_1c_2c_4)e_n^4 + o(e_n^5) \end{aligned} \tag{24}$$

Substituting Equation (20) and Equation (24) into the first formula of Equation (14), we have

$$\begin{aligned} y_n =& x^* + \frac{c_2(1 + c_1)}{c_1}e_n^2 + \frac{1}{c_1^2}(-2c_2^2 + 2c_1c_3 + 3c_1^2 c_3 + c_1^3 c_3 - 2c_1c_2^2 - c_1^2 c_2^2)e_n^3 \\ &+ \frac{1}{c_1^3}(3c_1^2 c_4 + 6c_1^3 c_4 + 4c_1^4 c_4 + c_1^5 c_4 + 5c_1c_2^3 + 3c_1^2 c_2^3 + c_1^3 c_2^3 + 4c_2^3 - 10c_1^2 c_2 c_3 \\ &- 7c_1c_2c_3 - 7c_1^3 c_2 c_3 - 2c_1^4 c_2 c_3)e_n^4 + o(e_n^5) \end{aligned} \tag{25}$$

With Equation (25), we get

$$\begin{aligned} F(y_n) =& c_2(1 + c_1)e_n^2 + \frac{1}{c_1}(-2c_2^2 + 2c_1c_3 + 3c_1^2 c_3 + c_1^3 c_3 - 2c_1c_2^2 - c_1^2 c_2^2)e_n^3 \\ &+ \frac{1}{c_1^2}(3c_1^2 c_4 + 6c_1^3 c_4 + 4c_1^4 c_4 + c_1^5 c_4 + 7c_1c_2^3 + 4c_1^2 c_2^3 + c_1^3 c_2^3 + 5c_2^3 - 10c_1^2 c_2 c_3 \\ &- 7c_1c_2c_3 - 7c_1^3 c_2 c_3 - 2c_1^4 c_2 c_3)e_n^4 + o(e_n^5) \end{aligned} \tag{26}$$

Thereby, with Equations (20) and (24)–(26), we obtain

$$\begin{aligned} g_2(x_n) =& c_1 - c_1c_2e_n - \frac{1}{c_1}(c_1c_3 + 3c_1^2 c_3 + c_1^3 c_3 - c_1c_2^2 - 2c_2^2)e_n^2 \\ &- \frac{1}{c_1^2}(2c_1^2 c_4 + 6c_1^3 c_4 + 4c_1^4 c_4 + c_1^5 c_4 + 4c_1c_2^3 + 2c_1^2 c_2^3 + 4c_2^3 - 4c_1^2 c_2 c_3 - 6c_1c_2c_3)e_n^3 \\ &- \frac{1}{c_1^3}(10c_1^4 c_5 - 3c_1^3 c_3^2 + 3c_1^3 c_5 - 4c_1^2 c_3^2 - 8c_2^4 + c_2c_4c_1^5 + 20c_2^2 c_1^2 c_3 + 15c_3c_1^3 c_2^2 \\ &+ 16c_1c_2^2 c_3 + 4c_1^4 c_2^2 c_3 + 10c_5c_1^5 + 5c_5c_1^6 + c_5c_1^7 - 10c_1c_2^4 - 6c_1^2 c_2^4 - 2c_2^4 c_1^3 \\ &- 7c_2c_4c_1^3 - 8c_2c_4c_1^2)e_n^4 + o(e_n^5) \end{aligned} \tag{27}$$

From Equations (25)–(27), it follows that

$$\begin{aligned} z_n =& x^* - \frac{c_2^2(1+c_1)}{c_1}e_n^3 - \frac{c_2}{c_1^3}(-3c_2^2c_1 - 2c_2^2c_1^2 - c_2^2 + 6c_3c_1^2 + c_3c_1 + 7c_3c_1^3 + 2c_1^4c_3)e_n^4 \\ & - \frac{1}{c_1^4}(9c_3^2c_1^3 + 2c_3^2c_1^2 + 4c_2^2 + 16c_1^4c_2c_4 + 2c_1^6c_2c_4 + 9c_2c_4c_1^5 - 21c_1^2c_2^2c_3 - 10c_1^3c_2^2c_3 - 8c_1c_2^2c_3 \\ & + 7c_1c_2^4 + 3c_1^2c_2^4 + 2c_1^3c_2^4 + 11c_2c_4c_1^3 + 2c_2c_4c_1^2 + 12c_1^4c_3^2 + 6c_1^5c_3^2 + c_1^6c_3^2 + c_1^4c_2^4)e_n^5 + o(e_n^6) \end{aligned} \tag{28}$$

It is similar to Equation (26), we have

$$\begin{aligned} F(z_n) =& - c_2^2(1+c_1)e_n^3 - \frac{c_2}{c_1^2}(-3c_2^2c_1 - c_2^2c_1^2 - c_2^2 + 6c_1^2c_3 + c_3c_1 + 7c_3c_1^3 + 2c_3c_1^4)e_n^4 \\ & + o(e_n^5) \end{aligned} \tag{29}$$

Moreover, substituting Equations (20), (24)–(26), (28) and (29) into Equation (13), we get

$$g_3(x_n) = c_1 - c_2^2(1+c_1)e_n^2 - \frac{c_2}{c_1}(2c_3c_1^3 + 7c_3c_1^2 + 7c_3c_1 + 2c_2^2c_1 + c_2^2 + 2c_3)e_n^e + o(e_n^4) \tag{30}$$

Substituting Equations (28)–(30) into the third formula of Equation (14), we get

$$x_{n+1} = z_n - \frac{F(z_n)}{g_3(x_n)} = x^* + \frac{(1+2c_1+c_1^2)c_2^4}{c_1^2}e_n^5 + o(e_n^6) \tag{31}$$

which just means that the iterative method defined by Equation (14) has fifth-order convergence. The proof is completed.

We further consider how to find the highest multiplicity of the root x^* in the iterative method. If x_n is the n-th iteration computed by an iterative method applied to f, then from Equation (9), we have

$$\begin{aligned} f_n &\approx \frac{(x_n - x^*)(x_n - x_1)h(x_n)}{m(x_n - x_1)h(x_n) + n(x_n - x^*)h(x_n) + (x_n - x^*)(x_n - x_1)h'(x_n)} \\ &= \frac{\varepsilon_n(x_n - x_1)h(x_n)}{m(x_n - x_1)h(x_n) + n\varepsilon_n h(x_n) + \varepsilon_n(x_n - x_1)h'(x_n)}, \end{aligned} \tag{32}$$

where $f_n = f(x_n)$. Because ε_n is small, we get $f_n \approx \dfrac{\varepsilon_n}{m}$. Similarly, we can compute that $f_{n+1} \approx \dfrac{\varepsilon_{n+1}}{m}$. Furthermore $\varepsilon_{n+1} - \varepsilon_n = x_{n+1} - x_n$. Consequently, when the iteration becomes closer to the root x^*, we can estimate its multiplicity by computing

$$m \approx \frac{x_{n+1} - x_n}{f_{n+1} - f_n}. \tag{33}$$

In the practical computing root x^* process, some iteration number is no more than two by using Equation (14). According to this case, we compute the root x^* by using Equation (14), then we select the initial value near the root x^*, and we use Newton iterative method to compute the highest multiplicity. Therefore, m is approximately the reciprocal of the divided difference of f for successive iteration x_n and x_{n+1}.

3. Numerical Results

We employ the proposed modification of Newton's method with three-step Equation (14) (MNM) to solve some nonlinear equations. All the computations were done by using Visual C++ 6.0 and were satisfied the condition such that $|f(x_n)| < 1.E-17$, $|x_n - x^*| < 1.E-17$. In order to show the effectiveness of our iterative method, we provide at least three different initial iterative values. From different initial iterative values, they can be convergent to the same iterative solution whose multiplicity is the highest multiplicity of nonlinear equation. We used the following test functions and obtained the approximate zeros x^* round up to the 17-th decimal place:

$$g_1(x) = \frac{(x-\sqrt{5})^7(x-\sqrt{3})^4}{(x-1)^2+1}, x_0 = -12.0, 23.0, 1.9, m = 7, n = 4, x^* = 2.236067977499790$$

$$g_2(x) = (\sin(x)^2 - 2x + 1)^5(x-2)^3, x_0 = -23, 23, 1.5, 1.7, m = 5, n = 3, x^* = 0.71483582544138924$$

$$g_3(x) = (8x\exp(-x^2) - 2x - 3)^8(x-2)^5, x_0 = -20, 20, 0.0, 1.5, m = 8, n = 5, x^* = -1.7903531791589544$$

$$g_4(x) = \frac{(2x\cos(x) + x^2 - 3)^{10}(x-3)^8}{(x^2+1)}, x_0 = -23, 23, 3.1, 2.99, m = 10, n = 8, x^* = 2.9806452794385368$$

$$g_5(x) = (\exp(-x^2 + x + 3) - x + 2)^9(x - \frac{13}{5})^6, x_0 = -18, 18, 2.5, 2.55, m = 9, n = 6, x^* = 2.4905398276083051$$

$$g_6(x) = (\exp(-x) + 2\sin(x))^4(x-2)^3, x_0 = 4.0, 2.5, m = 4, n = 3, x^* = 3.1627488709263654$$

$$g_7(x) = (\ln(x^2 + 3x + 5) - 2x + 7)^{15}(x - \sqrt{37})^{10}, x_0 = 34, 5, 5.8, m = 15, n = 10, x^* = 5.4690123359101421$$

$$g_8(x) = (\sqrt{x^2 + 2x + 5} - 2\sin(x) - x^2 + 3)^{20}(x - \sqrt{7})^{11}, x_0 = 9, 2, 3, m = 20, n = 11, x^* = 2.3319676558839640$$

$$g_9(x) = \frac{(x-2)^7(x-\sqrt{5})^8}{(x-1)^2+1}, x_0 = -12, 12, 2.1, 2.5, m = 8, n = 7, x^* = 2.2360679774997897$$

$$g_{10}(x) = (x - \frac{5}{2})^{\frac{15}{4}}\exp(x)(x-2)^2, x_0 = -13, 2.4, 2.1, m = \frac{15}{4}, n = 2, x^* = 2.50000000000$$

$$g_{11}(x) = (\sqrt{x} - x - 1)^{11}(x - \sqrt{3})^5, x_0 = 11, 2.0, 1.8, m = 11, n = 5, x^* = 2.147899035704787$$

$$g_{12}(x) = (\ln(x) + \sqrt{x} - 5)^{15}(x-8)^{10}, x_0 = 23, 9, 7, 8.1, m = 15, n = 10, x^* = 8.309432694231572$$

$$g_{13}(x) = (\sin(x)\cos(x) - x^3 + 1)^9(x - \frac{3}{2})^3, x_0 = 5, 0, 1.4, 1.8, m = 9, n = 3, x^* = 1.117078770687451$$

$$g_{14}(x) = (x - \sqrt{7})^5 e^x (x - \sqrt{2})^2, x_0 = 7, 3, 1.8, m = 5, n = 2, x^* = 2.6457513110645906$$

$$g_{15}(x) = (\ln(x) + \sqrt{x^4+1} - 2)^7(x - \sqrt{\frac{3}{2}})^6, x_0 = 5, 1.5, 1, 0.5, m = 7, n = 6, x^* = 1.222813963628973$$

$$g_{16}(x) = (\ln(x) + \sqrt{x^4+1} - 2)^8(x - \sqrt{\frac{3}{2}})^4(x - \sqrt{\frac{4}{3}})^2, x_0 = 12, 4, 1.3, 1.2, m = 8, n = 4, x^* = 1.222813963628973$$

$$g_{17}(x) = (x - \sqrt{7})^5(x - \sqrt{2})^2(x - \sqrt{\frac{3}{2}}), x_0 = 11, -11, 1.8, 0.5, m = 5, n = 2, x^* = 2.6457513110645906$$

$$g_{18}(x) = (\ln(x) + \sqrt{x^4+1} - 2)^{10}(x - \sqrt{\frac{3}{2}})^{10}(x-1), x_0 = 12, 2, 1.5, 1.3, m = 10, n = 10, x^* = 1.222813963628973$$

The computational results indicate that our proposed iterative method can converge to multiple roots whose multiplicity is the highest multiplicity of nonlinear equation. In the next section, for the special case which the multiplicity of the roots of nonlinear equation is a single multiplicity, we present the analysis result for comparison with previous methods.

Remark 1. *The method of the Formula (14) can also solve the problem of Euler equation for higher-order linear ordinary differential equation with variable coefficients. We consider a Euler equation*

$$x^8y^{(8)} + x^7y^{(7)} - 2x^5y^{(5)} - x^4y^{(4)} + 8x^2y^{(2)} + y = 0, \quad (34)$$

where $a_1 = 1, a_2 = 0, a_3 = -2, a_4 = -1, a_5 = 0, a_6 = 8, a_7 = 0, a_8 = 1$. *It is not difficult to know that the corresponding characteristic equation of Equation (34) is the following,*

$$\begin{aligned} p(K) =& K(K-1)(K-2)(K-3)(K-4)(K-5)(K-6)(K-7) \\ &+ K(K-1)(K-2)(K-3)(K-4)(K-5)(K-6) - 2K(K-1)(K-2)(K-3)(K-4) \\ &- K(K-1)(K-2)(K-3) + 8K(K-1) + 1 \\ &= K^8 - 27K^7 + 301K^6 - 178K^5 + 6053K^4 - 11572K^3 + 11401K^2 - 4370K + 1 \\ &= 0 \end{aligned} \quad (35)$$

Using iterative Formula (14), the real roots of characteristic Equation (35) are K_1 = *0.0002289696985655638,* K_2 = *0.9983856337914987,* K_3 = *2.111134535386187,* K_4 = *2.659996979882180,* K_5 = *4.396978604791156,* K_6 = *7.038659857754116, respectively. So the general solution of the Euler Equation (34) is* $y(x) = C_1x^{K_1} + C_2x^{K_2} + C_3x^{K_3} + C_4x^{K_4} + C_5x^{K_5} + C_6x^{K_6}$,*where* $C_1, C_2, C_3, C_4, C_5, C_6$ *are different constants.*

4. Comparison with Previous Methods

In this section, we use the proposed modification of Newton's method with three-step (14) (MNM) (the Formula (14) in our paper) to solve some nonlinear equations which the multiplicity of the roots is a single multiplicity, and compare them with King's method [17] (KM, (4), (13)) ($G = \frac{f(x)}{\frac{f(x-f(x))-f(x)}{(x-f(x))-x}} = \frac{-f^2(x)}{f(x-f(x))-f(x)}$ (4), $x_2 = x_1 - (x_0 - x_1)\frac{G_1}{G_0-G_1}$ (13)), the high-order convergence iteration methods without employing derivatives given in [18] (WFM, (6)) ($x_{n+1} = x_n - \frac{F^2(x_n)}{p \cdot F^2(x_n)+F(x_n)-F(x_n-F(x_n))}$ (6)), the improved method for finding multiple roots and itąŕs multiplicity of nonlinear equations given in [22] (PGM, ((6), (11))) ($G(x) = \begin{cases} \frac{f^2(x)}{\delta+f(x+f(x))-f(x)}, \, if\ f(x) \neq 0 \\ 0, \ \ if\ f(x) = 0 \end{cases}$, where $\delta = sign(f(x+f(x)) - f(x))f^2(x)$ (6), $x_{n+1} = x_n - \frac{G^2(x_n)}{pG^2(x_n)+G(x_n)-G(x_n-G(x_n))}$ (11)), the derivative free iterative method for finding multiple roots of nonlinear equations given in [23] (YM, ((7), (10))) ($H_\varepsilon(x) = \frac{\varepsilon f(x)^2}{f(x+\varepsilon f(x))-f(x)}$ (7), $x_{k+1} = x_k - \frac{2(x_k-x_{k-1})H_\varepsilon(x_k)}{H_\varepsilon(2x_k-x_{k-1})-H_\varepsilon(x_{k-1})}$ (10)), transformation methods for finding multiple roots of nonlinear equations [24] (YM, ((10), (12))) ($K(x) = \begin{cases} \frac{\varepsilon f(x)^2}{f_\varepsilon(x)}, if\, f(x) \neq 0 \\ 0, \ if\ f(x) = 0. \end{cases}$, where $f_\varepsilon(x) \approx \varepsilon f(x)f'(x)$ (10), $p_{k+1} = p_k - \frac{\mu K(p_k)^2}{K(p_k+\mu K(p_k))-K(p_k)}$ (12)), as well as Newton's first order method(NM) $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$). Displayed in Table 1 are the number of iterations satisfied such that$|f(x_n)| < 1.E-17$, $|x_n - x^*| < 1.E-17$. All the computations were done by using Visual C++ 6.0. Since King's method and Yun's method require two starting values, we have used $x_1 = x_0 - 0.1$. We used the following test functions and obtained the approximate zeros x^* round up to the 17th decimal place:

$$f_1(x) = \frac{(x-\sqrt{5})^4}{(x-1)^2+1}, \quad m = 4, \quad x^* = 2.236067977499790$$

$$f_2(x) = (\sin^2(x) - 2x + 1)^5, \quad m = 5, \quad x^* = 0.71483582544138924$$

$$f_3(x) = (8xe^{-x^2} - 2x - 3)^8, \quad m = 8, \quad x^* = -1.7903531791589544$$

$$f_4(x) = \frac{(2x\cos(x) + x^2 - 3)^{10}}{(x^2+1)}, \quad m = 10, \quad x^* = 2.9806452794385368$$

$$f_5(x) = (e^{-x^2+x+3} - x + 2)^9, \quad m = 9, \quad x^* = 2.4905398276083051$$

$$f_6(x) = (e^{-x} + 2\sin(x))^4, \quad m = 4, \quad x^* = 3.1627488709263654$$

$$f_7(x) = (\ln(x^2+3x+5) - 2x + 7)^8, \quad m = 8, \quad x^* = 5.4690123359101421$$

$$f_8(x) = (\sqrt{x^2+2x+5} - 2\sin(x) - x^2 + 3)^5, \quad m = 5, \quad x^* = 2.3319676558839640$$

$$f_9(x) = (x-2)^4/((x-1)^2+1), \quad m = 4, \quad x^* = 2.0000000000000000$$

$$f_{10}(x) = (x-2.5)^{\frac{15}{4}}e^x, \quad m = \frac{15}{4}, \quad x^* = 2.500000000000000$$

$$f_{11}(x) = (\sqrt{x} - \frac{1}{x} - 1)^7, \quad m = 7, \quad x^* = 2.147899035704787$$

$$f_{12}(x) = (\ln(x) + \sqrt{x} - 5)^3, \quad m = 3, \quad x^* = 8.309432694231572$$

$$f_{13}(x) = (\sin(x)\cdot\cos(x) - x^3 + 1)^9, \quad m = 9, \quad x^* = 1.117078770687451$$

$$f_{14}(x) = ((x-3)\exp(x))^5, \quad m = 5, \quad x^* = 3.0000000000000000$$

$$f_{15}(x) = (\ln(x) + \sqrt{x^4+1} - 2)^7, \quad m = 7, \quad x^* = 1.222813963628973$$

Note that we used NC in Table 1 to mean that the method does not converge to the root. And these methods can converge to root by using closer initial values. The computational results in Table 1 demonstrate that our proposed iterative method (MNM) requires less number of iterations than those of KM, WFM, PGM, YM, and NM. Therefore, it is significant and applicable and can compete with other existing methods.

Table 1. Comparison of various aiterative methods for the case of single multiplicity.

f(x)	**MNM**	**KM [17]**	**WFM [18]**	**PGM [22]**	**YM [23]**	**YM [24]**
Parameters		$\boldsymbol{x_1 = x_0 - 0.1}$	$\boldsymbol{p = 1}$ $\boldsymbol{x_1 = x_0 - 0.1}$	$\boldsymbol{p = 1}$	$\boldsymbol{\varepsilon = 10^{-8}}$ $\boldsymbol{\mu = 1}$	$\boldsymbol{\varepsilon = 10^{-8}}$
$f_1, x_0 = 3.0$	2	4	6	4	3	2
$f_2, x_0 = 1.5$	2	NC	NC	4	NC	5
$f_3, x_0 = -1.1$	2	NC	NC	NC	3	3
$f_4, x_0 = 3.2$	2	3	3	28	NC	NC
$f_5, x_0 = 3.0$	2	NC	66	4	3	NC

Table 1. *Cont.*

f(x)	MNM	KM [17]	WFM [18]	PGM [22]	YM [23]	YM [24]
Parameters		$x_1 = x_0 - 0.1$	$p = 1$ $x_1 = x_0 - 0.1$	$p = 1$	$\varepsilon = 10^{-8}$ $\mu = 1$	$\varepsilon = 10^{-8}$
$f_6, x_0 = 3.5$	2	5	4	NC	3	2
$f_7, x_0 = 6.5$	2	NC	3	5	2	2
$f_8, x_0 = 2.7$	2	NC	NC	9	3	3
$f_9, x_0 = 2.5$	1	4	4	6	4	3
$f_{10}, x_0 = 2.8$	2	6	3	20	3	2
$f_{11}, x_0 = 2.5$	2	3	3	3	NC	2
$f_{12}, x_0 = 9.0$	1	3	4	4	4	3
$f_{13}, x_0 = 1.4$	2	NC	NC	NC	NC	4
$f_{14}, x_0 = 3.4$	2	59	NC	NC	4	6
$f_{15}, x_0 = 1.7$	2	NC	NC	32	4	2

5. Efficiency of Iterative Methods

In the following we compare the efficiency of methods mentioned in Section 1 whose the multiplicity of the roots of nonlinear equation is a single multiplicity. We consider the definition of efficiency index [29–31] as EFF = $r^{\frac{1}{\theta}}$, where r is the order of the method and θ is number of function (and derivatives) evaluations per iteration required by the method. These results are presented in Table 2. In Table 2, we listed the methods according to decreasing order of efficiency index, and multiplicity m of roots with all methods under the row of King's method (including King's method) are unknown. For unknown multiplicity m, our new method comes second, next to King's method.

Table 2. Comparison the efficiency of various iterative methods for the case of single multiplicity.

Method	Reference	r	θ	EFF
$Neta$	[8] (49) $m \neq 3$	2.732	2	1.653
$Neta$	[8] (51)	2.732	2	1.653
$Neta\ and\ Johnson$	[10] $m = 2$	4	3	1.587
$Neta$	[8] (39)	3	3	1.442
$Neta$	[8] (29) $m \neq 3$	3	3	1.442
$Chun\ and\ Neta$	[7] (22)	3	3	1.442
$Chun{,}Bae,\ and\ Neta$	[9] (14)	3	3	1.442
$Victory\ and\ Neta$	[4] (3)	3	3	1.442
$Hansen\ and\ Patrick$	[2] (8.1)	3	3	1.442
$Halley$	[12]	3	3	1.442
$Laguerre$	[13]	3	3	1.442

Table 2. *Cont.*

Method	Reference	r	θ	EFF
$Dong$	[14] (7), (8)	3	3	1.442
$Dong$	[5] (9), (10)	3	3	1.442
$Osada$	[6]	3	3	1.442
$E.Schrder$	[1]	2	2	1.414
$Neta\ and\ Johnson$	[10] $m \neq 2$	4	4	1.414
$Neta$	[11]	4	4	1.414
$Neta$	[8] (32) $m = 3$	2	3	1.259
$Werner$	[15] (16) $m = 2$	1.5	3	1.145
$King$	[17]	1.618	2	1.272
$Our\ method$	This paper	5	8	1.223
$Xinyuan\ Wu$	[18]	2	4	1.190
$Xinyuan\ Wu$	[19]	2	4	1.190
$P.K.Parida$	[22]	2	4	1.190
$Beong\ In\ Yun$	[23]	2	4	1.190
$Beong\ In\ Yun$	[24]	2	4	1.190

6. Conclusions

A new iterative method with fifth-order convergence has been developed as a modification of Newton's method for finding multiple roots with unknown multiplicity m whose multiplicity m is the highest multiplicity of nonlinear equation. Several numerical examples demonstrate that the proposed iterative method is efficient. For the special case which the multiplicity of the roots of nonlinear equation is a single multiplicity, our method is more efficient and performs better than classical Newton's method and many other existing methods.

Acknowledgments

We take the opportunity to thank the reviewers for their thoughtful and meaningful comments. This work was supported by National Natural Science Foundation of China (61263034), Scientific and Technology Foundation Funded Project of Guizhou Province ([2014] 2092, [2014] 2093), Scientific and Technology Joint Foundation Funded Project of Guizhou Province ([2011] 18, [2013] 20), National Bureau of statistics Foundation Funded Project (2014LY011), Key Laboratory of Pattern Recognition and Intelligent System of Construction Project of Guizhou Province ([2009] 4002), Information Processing and Pattern Recognition for Graduate Education Innovation Base of Guizhou Province.

Author Contributions

The idea for this research work is proposed by Xiaowu Li and Mingsheng Zhang, numerical solution of nonlinear dynamic system is done by Juan Liang, the code procedure realization is achieved by

Zhinan Wu and Feng Pan, theorem proof is done by Lin Wang, and the paper writing is completed by Juan Liang and Xiaowu Li.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Schröder, E. Über unendlich viele Algorithmen zur Auflösung der Gleichungen. *Math. Ann.* **1870**, *2*, 317–365.
2. Hansen, E.; Patrick, M. A family of root finding methods. *Numer. Math.* **1977**, *27*, 257–269.
3. Li, S.G.; Cheng, L.Z.; Neta, B. Some fourth-order nonlinear solvers with closed formulae for Multiple roots. *Comput. Math. Appl.* **2010**, *59*, 126–135.
4. Victory, H.D.; Neta, B. A higher order method for multiple zeros of nonlinear functions. *Int. J. Comput. Math.* **1983**, *12*, 329–335.
5. Dong, C. A family of multipoint iterative functions for finding multiple roots of equations. *Int. J. Comput. Math.* **1987**, *21*, 363–367.
6. Osada, N. An optimal multiple root-finding method of order three. *J. Comput. Appl. Math.* **1994**, *51*, 131–133.
7. Chun, C.; Neta, B. A third-order modification of Newtonąŕs method for multiple roots. *Appl. Math. Comput.* **2009**, *211*, 474–479.
8. Neta, B. New third order nonlinear solvers for multiple roots. *Appl. Math. Comput.* **2008**, *202*, 162–170.
9. Chun, C.; Bae, H.J.; Neta, B. New families of nonlinear third-order solvers for finding multiple roots. *Comput. Math. Appl.* **2009**, *57*, 1574–1582.
10. Neta, B.; Johnson, A.N. High-order nonlinear solver for multiple roots. *Comput. Math. Appl.* **2008**, *55*, 2012–2017.
11. Neta, B. Extension of Murakami's High order nonlinear solver to multiple roots. *Int. J. Comput. Math.* **2010**, *87*, 1023–1031.
12. Halley, E. A new, exact and easy method of finding the roots of equations generally and that without any previous reduction. *Philos. Trans. R. Soc. Lond.* **1694**, *18*, 136–148.
13. Laguerre, E.N. Sur une méthode pour obtener par approximation les racines d'une équation algébrique qui a toutes ses racines réelles. 2e séries. *Nouv. Ann. Math.* **1880**, *19*, 88–103.
14. Dong, C. A basic theorem of constructing an iterative formula of the higher order for computing multiple roots of an equation. *Math. Numer. Sin.* **1982**, *11*, 445–450.
15. Werner, W. Iterationsverfahren höherer Ordnung zur Lösung nicht linearer Gleichungen. *Z. Angew. Math. Mech.* **1981**, *61*, T322–T324.
16. Traub, J.F. *Iterative Methods for the Solution of Equations*; Prentice Hall: Englewood, IL, USA, 1964.
17. King, R.F. A secant method for multiple roots. *BIT* **1977**, *17*, 321–328.

18. Wu, X.Y.; Fu, D.S. New higher-order convergence iteration methods without employing derivatives for solving nonlinear equations. *Comput. Math. Appl.* **2001**, *41*, 489–495.
19. Wu, X.Y.; Xia, J.L.; Shao, R. Quadratically convergent multiple roots finding method without derivatives. *Comput. Math. Appl.* **2001**, *42*, 115–119.
20. Steffensen, I.F. Remark on Iteration. *Skand. Aktuarietidskr* **1933**, *16*, 64–72.
21. Wu, X.Y. A new continuation Newton-like method and its deformation. *Appl. Math. Comput.* **2000**, *112*, 75–78.
22. Parida, P.K.; Gupta, D.K. An improved method for finding multiple roots and it's multiplicity of nonlinear equations in R. *Appl. Math. Comput.* **2008**, *202*, 498–503.
23. Yun, B.I. A derivative free iterative method for finding multiple roots of nonlinear equations. *Appl. Math. Lett.* **2009**, *22*, 1859–1863.
24. Yun, B.I. Transformation methods for finding multiple roots of nonlinear equations. *Appl. Math. Comput.* **2010**, *217*, 599–606.
25. Kioustelidis, J.B. A derivative-free transformation preserving the order of convergence of iteration methods in case of multiple zeros. *Numer. Math.* **1979**, *33*, 385–389.
26. Wang X.; Liu, L. Modified Ostrowski's method with eighth-order convergence and high efficiency index. *Appl. Math. Lett.* **2010**, *23*, 549–554.
27. Bi, W.; Ren, H.; Wu, Q. New family of seventh-order methods for nonlinear equations. *Appl. Math. Comput.* **2008**, *203*, 408–412.
28. Bi, W.; Ren, H.; Wu, Q. Three-step iterative methods with eighth-order convergence for solving nonlinear equations. *J. Comput. Appl. Math.* **2009**, *225*, 105–112.
29. Gautschi, W. *Numerical Analysis: An Introduction*; Birkhäuser: Boston, MA, USA, 1997.
30. Neta, B. On a family of multipoint methods for non-linear equations. *Int. J. Comput. Math.* **1981**, *9*, 353–361.
31. Li, X.; Mu, C.; Ma, J.; Wang, C. Sixteenth-order method for nonlinear equations. *Appl. Math. Comput.* **2010**, *215*, 3754–3758.