A Modified Ren’s Method with Memory Using a Simple Self-Accelerating Parameter

Xiaofeng Wang * and Qiannan Fan

School of Mathematics and Physics, Bohai University, Jinzhou 121000, China; qnfbh@163.com
* Correspondence: w200888w@163.com; Tel.: +86-15041699258

Received: 08 March 2020; Accepted: 1 April 2020; Published: 7 April 2020

Abstract: In this paper, a self-accelerating type method is proposed for solving nonlinear equations, which is a modified Ren’s method. A simple way is applied to construct a variable self-accelerating parameter of the new method, which does not increase any computational costs. The highest convergence order of new method is \(2 + \sqrt{6} \approx 4.4495\). Numerical experiments are made to show the performance of the new method, which supports the theoretical results.

Keywords: self-accelerating type method; iterative method; variable parameter; root-finding

MSC: 65H05; 65B99

1. Introduction

Self-accelerating type method is a kind of efficient iterative method with memory for solving nonlinear equations, which chooses some varying parameters as self-accelerating parameters in the iteration processes. The self-accelerating parameter is calculated by using information from previous and current iterations, which does not increase the computational cost of iterative method. Thus, self-accelerating type methods possess a very high computational efficiency. Self-accelerating parameter is very important to self-accelerating type method, which can make a big difference to the efficiency of iterative method. There are two ways to construct the self-accelerating parameter, which are secant method and interpolation method. Traub’s method [1] is one of the most representative methods for self-accelerating type method, which can be written as

\[
\begin{align*}
\omega_n &= x_n + T_n f(x_n), \quad T_n = -\frac{f[x_n, x_{n-1}]}{f(x_n, w_n)}, \\
x_{n+1} &= x_n - \frac{f(x_n)}{f[x_n, w_n]},
\end{align*}
\]

where \(f[x_n, w_n] = \{f(x_n) - f(w_n)\}/(x_n - w_n)\) is a divided difference and the self-accelerating parameter \(T_n\) is constructed by using secant method. The convergence order of method (1) is \(1 + \sqrt{2}\).

Similar, Petković et al. [2] obtained the following self-accelerating type method with order \(2 + \sqrt{5}\)

\[
\begin{align*}
\omega_n &= x_n - T_n f(x_n), \quad T_n = -\frac{f[x_n, x_{n-1}]}{f(x_n, w_n)}, \\
y_n &= x_n - \frac{f(x_n)}{f[x_n, w_n]}, \\
x_{n+1} &= y_n - \frac{f(y_n)}{f[x_n, w_n]} \left(1 + \frac{f(y_n)}{f(x_n)} + \frac{f(y_n)}{f(w_n)}\right),
\end{align*}
\]
Zheng et al. [3] also proposed a self-accelerating type method with order \((3 + \sqrt{3})/2 \approx 3.3028\), which can be given by

\[
\begin{align*}
   w_n &= x_n + T_n f(x_n),
   T_n = f[x_n, x_{n-1}]^{-1}, \\
   y_n &= x_n - \frac{f(x_n)}{f[x_n, w_n]}, \\
   x_{n+1} &= x_n - \frac{f(x_n)^2}{f[x_n, w_n](f(x_n) - f(y_n))}.
\end{align*}
\]

(3)

Using interpolation method to construct self-accelerating parameter, we [4–6] obtained some self-accelerating type methods, one of them is given by

\[
\begin{align*}
   y_n &= x_n - \frac{f(x_n)}{f'(x_n) + f'(x_n)},
   T_n = -\frac{H_2(x_n)}{f'(x_n)}, \\
   x_{n+1} &= y_n - \frac{f(y_n)}{2f'(x_n)} \left(1 + 2 \frac{f(y_n)}{f'(x_n)} \right),
\end{align*}
\]

(4)

where \(H_2(x) = f(x_n) + f'(x_n)(x - x_n) + f[x_n, x_n, y_{n-1}](x - x_n)^2\). \(H_2'(x_n) = 2f[x_n, x_n, y_{n-1}]\) and the self-accelerating parameter \(T_n\) is constructed by Hermite interpolation polynomial. Method (4) has convergence order \((5 + \sqrt{17})/2\). Using Newton interpolation to construct self-accelerating parameter, Džunić et al. [7,8] gave some efficient self-accelerating type methods, one of them can be given by

\[
\begin{align*}
   w_n &= x_n + T_n f(x_n),
   T_n = \frac{1}{N_2(x_n)}, \\
   x_{n+1} &= x_n - \frac{f(x_n)}{f'(x_n)},
\end{align*}
\]

(5)

where \(N_2 = N_2(t; x_n, x_{n-1}, w_{n-1})\) is Newton’s interpolation polynomial of second degree and \(N_2'(x_n) = f[x_n, x_{n-1}] + f[x_n, x_{n-1}, w_{n-1}](x_n - x_{n-1})\). The convergence order of method (5) is 3. By increasing the number of self-accelerating parameters or using interpolation polynomial of high degree to construct self-accelerating parameter, the convergence order and the computational efficiency of self-accelerating type methods can be improved greatly. Recently, more and more higher order self-accelerating type methods have been presented for solving nonlinear equations. Zaka et al. [9] gave an efficient tri-parametric iterative method by using Newton interpolation polynomial to construct self-accelerating parameter. Using four self-accelerating parameters, Lotfi and Assari [10] obtained a derivative-free iterative method with efficiency index near 2. We also proposed an efficient iterative method with \(n\) self-accelerating parameters, some known methods [1–3,7–14] can be seen as the special cases of our method [15]. Cordero et al. [16] and Campos et al. [17] designed some new self-accelerating type methods and studied the stability of their methods. A more extensive list of references as well as a survey on progress made on the self-accelerating type methods may be found in the recent books by Petković et al. [18] and Iliev et al. [19]. It is obvious that the secant method and interpolation method are very effective ways to construct self-accelerating parameter. However, all of these researches focused their works on the design of the methods trying to improve their convergence order and computational efficiency. Only some recent works [20,21] try to use new techniques to construct the self-accelerating parameter.

The main purpose of this paper is that a new way is proposed to construct a self-accelerating parameter, which is different from the secant method and interpolation method. In Section 2, firstly, based on Ren’s method [22], a modified optimal fourth-order method is proposed for solving nonlinear equations. Then, the modified iterative method is extended into a new self-accelerating type method by using a self-accelerating parameter. Using the interpolation method to construct the self-accelerating parameter, the convergence order of the new method reaches 4.2361. In Section 3, a new way is applied to construct the self-accelerating parameter. The maximal convergence order of new method is 4.4495. Numerical examples are given in Section 4 to confirm theoretical results. Section 5 is a short conclusion.
2. A New Self-Accelerating Type Method

In order to obtain a new self-accelerating type method, we first propose a modified fourth-order method without memory. Based on the Ren’s method [22],

\[
\begin{align*}
    w_n &= x_n + f(x_n), \\
    y_n &= x_n - \frac{f(x_n)}{f'(x_n)}, \\
    x_{n+1} &= y_n - \frac{f(y_n)}{f'(y_n)},
\end{align*}
\]

(6)

we construct the following modified iterative method

\[
\begin{align*}
    w_n &= x_n + f(x_n), \\
    z_n &= x_n - \frac{f(x_n)}{f'(x_n)}, \\
    y_n &= z_n - T(z_n - x_n)^2, \\
    x_{n+1} &= y_n - \frac{f(y_n)}{f'(y_n)},
\end{align*}
\]

(7)

where \(T \in R\) is a parameter. For method (7), we have the following convergence analysis.

**Theorem 1.** If function \(f : I \subset R \rightarrow R\) is sufficiently differentiable and has a simple zero \(\xi\) on an open interval \(I\), then iterative method (7) is of fourth-order convergence and its error equation is as follows

\[
e_{n+1} = (c_2 + c_2f'(\xi) - T)c_2^2(1 + f'(\xi)) - c_3(1 + f'(\xi)) - c_2 T e_n^4 + O(e_n^5),
\]

(8)

where \(e_n = x_n - \xi, T \in R\) and \(c_n = (1/n!) f^{(n)}(\xi)/f'(\xi), n = 2, 3, \ldots\)

**Proof.** Using Taylor expansion of \(f(x)\), we have

\[
f(x_n) = f'(\xi)[e_n + c_2 e_n^2 + c_3 e_n^3 + c_4 e_n^4 + c_5 e_n^5 + O(e_n^6)],
\]

(9)

\[
e_{n, w} = w_n - \xi = e_n + f(x_n) = (1 + f'(\xi))e_n + c_2 f'(\xi)e_n^2 + c_3 f'(\xi)e_n^3 + O(e_n^4),
\]

(10)

\[
f(w_n) = f'(\xi)[1 + f'(\xi)]e_n + c_2(1 + 3f'(\xi) + f'(\xi)^2)e_n^2 + (2c_2^2 f'(\xi)(1 + f'(\xi))
\]

\[
+ c_3(1 + 4f'(\xi) + 3f'(\xi)^2 + f'(\xi)^3)e_n^3 + O(e_n^4)],
\]

(11)

\[
f[x_n, y_n] = f'(\xi)[1 + c_2(2 + f'(\xi))e_n + (c_2^2 f'(\xi) + c_3(3 + 3f'(\xi) + f'(\xi)^2))e_n^2
\]

\[
+ (2 + f'(\xi))(2c_2 c_3 f'(\xi) + c_4(2 + 2f'(\xi) + f'(\xi)^2))e_n^3 + O(e_n^4)].
\]

(12)

According to (7), (9) and (12), we get

\[
e_{n, z} = z_n - \xi = c_2(1 + f'(\xi))e_n^2 + (-c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 3f'(\xi) + f'(\xi)^2))e_n^3 + O(e_n^4).\]

(13)

From (7) and (13), we obtain

\[
e_{n, y} = y_n - \xi = (c_2 + c_2 f'(\xi) - T)e_n^2 + (-c_2^2(2 + 2f'(\xi) + f'(\xi)^2)
\]

\[
+ c_3(2 + 3f'(\xi) + f'(\xi)^2) + 2c_2(1 + f'(\xi))T)e_n^3 + O(e_n^4).
\]

(14)

By a similar argument to that of (9), we get

\[
f(y_n) = f'(\xi)(c_2 + c_2 f'(\xi) - T)e_n^2 + f'(\xi)(-c_2^2(2 + 2f'(\xi) + f'(\xi)^2)
\]

\[
+ c_3(2 + 3f'(\xi) + f'(\xi)^2) + 2c_2(1 + f'(\xi))T)e_n^3 + O(e_n^4).
\]

(15)

Using (9), (11), (14) and (15), we have

\[
f[x_n, y_n] = f'(\alpha) + c_2 f'(\xi)e_n + f'(\xi)(c_3 + c_2^2(1 + f'(\xi)) - c_2 T)e_n^2 + O(e_n^3),
\]

(16)
\[ f[y_n, w_n] = f'(\xi) + c_2 f'(\xi)(1 + f'\xi) e_n + f'(\xi)(c_3(1 + f'\xi))^2 + c_3(1 + 2f'\xi) - c_2 T e_n^2 + O(e_n^3). \]  

(17)

Together with (7) and (14)–(17), we obtain the error equation

\[ e_{n+1} = x_{n+1} - \xi = (c_2 + c_2 f'\xi - T)[c_2^2(1 + f'\xi) - c_3(1 + f'\xi)] - c_2 T e_n^4 + O(e_n^5). \]  

(18)

The proof is completed. \[ \square \]

**Remark 1.** From (8), we can see that the convergence order of method (7) is at least five provided that \( T = c_2 (1 + f'\xi) \). Hence, in order to obtain a new self-accelerating type method, we will use a self-accelerating parameter \( T_n \) to replace the parameter \( T \) if the parameter \( T_n \) satisfies the relation \( \lim_{n \to \infty} T_n = T = c_2 (1 + f'\xi) \). Similar to methods (4) and (5), we can use interpolation method to construct self-accelerating parameter. For example, the self-accelerating parameter \( T_n \) can be given by

\[ T_n = \frac{N''_2(x_n)}{2N'_2(x_n)} (1 + N'_2(x_n)), \]  

(19)

where \( N_2(t) = N_2(t; x_n, x_{n-1}, w_{n-1}) \) is Newton’s interpolatory polynomial of second degree, \( N'_2(x_n) = f[x_n, x_{n-1}] + f[x_n, x_{n-1}, w_{n-1}](x_n - x_{n-1}) \) and \( N''_2(x_n) = 2f[x_n, x_{n-1}, w_{n-1}] \). Now, we obtain a new self-accelerating type method as follows:

\[
\begin{aligned}
w_n &= x_n + f(x_n), \\
z_n &= x_n - \frac{f(x_n)}{f(x_n) - f(y_n)}, \\
y_n &= z_n - T_n(z_n - x_n)^2, \\
x_{n+1} &= y_n - \frac{f(y_n)}{f(x_n, y_n) + f(y_n, w_n) - f(x_n, w_n)}. \\
\end{aligned}
\]  

(20)

**Theorem 2.** Let the varying parameter \( T_n \) be calculated by (19) in method (20). If an initial value \( x_0 \) is sufficiently close to a simple zero \( \xi \) of function \( f(x) \), then the \( R \)-order of convergence of self-accelerating type method (20) is at least \( 2 + \sqrt{5} \approx 4.2361 \).

**Proof.** If an iterative method (IM) generates sequence \( \{x_n\} \) that converges to the zero \( \xi \) of \( f(x) \) with the \( R \)-order \( O_R(IM, a) \geq r \), then we can write

\[ e_{n+1} \sim D_{n,r} e_n^r, \]  

(21)

where \( e_n = x_n - \xi \) and the limit of \( D_{n,r} \) is the asymptotic error constant of iterative method, as \( n \to \infty \). So,

\[ e_{n+1} \sim D_{n,r}(D_{n-1,r} e_{n-1}^r)^r = D_{n,r} D_{n-1,r}^r e_{n-1}^{2r}. \]  

(22)

Similar to (22), if the \( R \)-order of iterative sequence \( \{y_n\} \) is \( p \), then

\[ e_{n,y} \sim D_{n,p} e_n^p \sim D_{n,p}(D_{n-1,r} e_{n-1}^r)^p = D_{n,p} D_{n-1,p}^p e_{n-1}^{2p}. \]  

(23)

According to (14) and (18), we obtain

\[ e_{n,y} = y_n - \xi \sim (c_2 + c_2 f'\xi - T_n)e_n^2, \]  

(24)

\[ e_{n+1} = x_{n+1} - \xi \sim (c_2 + c_2 f'\xi - T_n)[c_2^2(1 + f'\xi) - c_3(1 + f'\xi)] - c_2 T_n e_n^4. \]  

(25)
Here, we omit the higher-order terms in (24)–(25). Let \( N_2(t) \) be the Newton interpolating polynomial of degree two that interpolates the function \( f \) at nodes \( x_n, w_{n-1}, x_{n-1} \) contained in interval. Then, the error of the Newton interpolation can be expressed as follows:

\[
f(t) - N_2(t) = \frac{f^{(3)}(\zeta)}{3!}(x-x_n)(x-w_{n-1})(x-x_{n-1}), \zeta \in I. \tag{26}
\]

Differentiating (26) at the point \( t = x_n \), we get

\[
N'_2(x_n) \sim f'(\xi)(1 - c_3(1 + f'(\xi))e^2_{n-1} + O(e^3_{n-1})), \tag{27}
\]

\[
N''_2(x_n) \sim 2f''(\xi)c_2 + c_3(2 + f'(\xi))e_{n-1} + O(e^2_{n-1})), \tag{28}
\]

\[
T_n = \frac{N''_2(x_n)}{2N'_2(x_n)} (1 + N'_2(x_n)) \tag{29}
\]

Using (24), (25) and (29), we get

\[
e_{n,y} \sim (c_2 + c_2f'(\xi) - T_n)e^2_{n} \tag{30}
\]

\[
\sim -c_3(1 + f'(\xi))(2 + f'(\xi))e_{n-1}(D_{n-1,r}e^2_{n-1})^2 \tag{31}
\]

Solving system (32), we obtain \( r = 2 + \sqrt{5} \approx 4.2361 \) and \( p = \sqrt{5} \approx 2.2361 \). Therefore, the \( R \)-order of method (20) is at least 4.2361, when \( T_n \) is calculated by (19). The proof is completed. \( \Box \)

3. A New Technique to Construct the Self-Accelerating Parameter of New Method

In this Section, we will give a new way to construct the self-accelerating parameter. It is known that Steffensen method (SM) without memory [23] converges quadratically, which can be written as

\[
\begin{aligned}
w_n &= x_n + f(x_n), \\
x_{n+1} &= x_n - \frac{f(x_n)}{f'(x_n)},
\end{aligned} \tag{33}
\]

which satisfies the following expression

\[
\lim_{n \to \infty} \frac{x_{n+1} - \xi}{(x_n - \xi)^2} = \lim_{n \to \infty} \frac{e_{n+1}}{e^2_{n}} = c_2(1 + f'(\xi)). \tag{34}
\]

From (34), we know that the asymptotic error constant of SM is \( c_2(1 + f'(\xi)) \). Coincidentally, the first two steps of our method (20) is Steffensen method, so Equation (34) can be the self-accelerating parameter, which satisfies \( \lim_{n \to \infty} T_n = T = c_2(1 + f'(\xi)) \). Since the root \( \xi \) of function is unknown, we could use sequence information from the current and previous iterations to approximate the root \( \xi \).
Theorem 3. From (13), (14) and (18), we get

$$T_n = \frac{z_{n-1} - x_n}{(x_n - x_{n-1})^2},$$ \hspace{1cm} (35)

**Formula 2:**

$$T_n = \frac{(z_{n-1} - x_n)(y_{n-1} - x_{n-1})}{(x_n - x_{n-1})^3}.$$ \hspace{1cm} (36)

**Formula 3:**

$$T_n = \frac{2z_{n-1} - x_n}{(x_n - x_{n-1})^2} \left\{ \frac{x_n + y_{n-1} - x_{n-1} - 2z_{n-1}}{z_{n-1} - x_{n-1}} \right\} - \frac{2z_{n-1} - x_n}{(x_n - y_{n-1})(z_{n-1} - x_{n-1})}$$

$$\left( \frac{(x_n - z_{n-1})^2}{(z_{n-1} - x_{n-1})} \right)^2,$$ \hspace{1cm} (37)

**Formula 4:**

$$T_n = \frac{(z_{n-1} - x_n)(y_{n-1} - x_{n-1})}{(x_n - x_{n-1})^3} \left\{ \frac{x_n + y_{n-1} - x_{n-1} - 2z_{n-1}}{z_{n-1} - x_{n-1}} \right\} - \frac{2z_{n-1} - x_n}{(x_n - y_{n-1})(z_{n-1} - x_{n-1})}$$

$$\left( \frac{(x_n - z_{n-1})^2}{(z_{n-1} - x_{n-1})} \right)^2,$$ \hspace{1cm} (38)

The self-accelerating parameter $T_n$ is calculated by using one of the formulas (35)–(38).

**Theorem 3.** Let varying parameters $T_n$ be calculated by (35) or (36) in method (20), respectively. If an initial value $x_0$ is sufficiently close to a simple zero $\xi$ of $f(x)$, then the R-order of convergence of iterative method (20) is at least $2 + \sqrt{5} \approx 4.2361$.

**Proof.** From (13), (14) and (18), we get

$$z_{n-1} - x_n = c_2(1 + f'(\xi))c_{n-1}^2 + (-c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 3f'(\xi) + f'(\xi)^2)c_{n-1}^3 + (c_2^2(3 + 3f'(\xi) + f'(\xi)^2))$$

$$+ c_3(c_3(3 + 4f'(\xi) + f'(\xi)^2) + c_4(3 + 3f'(\xi) + f'(\xi)^2)$$

$$- c_3T_{n-1}) - c_2(2c_3(3 + 4f'(\xi) + f'(\xi)^2) + f'(\xi)^3) + T_{n-1}^3) + O(c_{n-1})$$ \hspace{1cm} (39)

$$x_n - x_{n-1} = -c_{n-1} + (c_2 + c_2f'(\xi) - T_{n-1})(c_2(1 + f'(\xi))) - c_3(1 + f'(\xi)) - c_2T_{n-1})c_{n-1}^4 + O(c_{n-1})$$ \hspace{1cm} (40)

$$y_{n-1} - x_{n-1} = -c_{n-1} + (c_2 + c_2f'(\xi) - T_{n-1})c_{n-1}^2 + (-c_2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 3f'(\xi) + f'(\xi)^2)$$

$$+ c_2(1 + f'(\xi))T_{n-1}c_{n-1}^2 + c_2c_3(4 + 5f'(\xi) + 3f'(\xi)^2 + f'(\xi)^3) - c_2c_3(7 + 10f'(\xi)$$

$$+ 7f'(\xi)^2 + 2f'(\xi)^3) - c_2(c_3(5 + 6f'(\xi) + 3f'(\xi)^2)T_{n-1} + (1 + f'(\xi))(c_2(c_3 + 3f'(\xi)$$

$$+ f'(\xi)^2) + 2c_3(2 + f'(\xi))T_{n-1}) + O(c_{n-1}^5).$$ \hspace{1cm} (41)

From (39)–(41), we get

$$T_n = \frac{z_{n-1} - x_n}{(x_n - x_{n-1})^2}$$

$$= c_2(1 + f'(\xi)) + (-c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 3f'(\xi) + f'(\xi)^2)c_{n-1} + O(c_{n-1}^2),$$ \hspace{1cm} (42)

$$c_2(1 + f'(\xi)) - T_n = (c_2^2(2 + 2f'(\xi) + f'(\xi)^2) - c_3(2 + 3f'(\xi) + f'(\xi)^2)c_{n-1} + O(c_{n-1}^2),$$ \hspace{1cm} (43)

$$T_n = \frac{(z_{n-1} - x_n)(y_{n-1} - x_{n-1})}{(x_n - x_{n-1})^3}$$

$$= c_2(1 + f'(\xi)) + (c_3(2 + 3f'(\xi) + f'(\xi)^2) - c_2^2(3 + 4f'(\xi) + 2f'(\xi)^2)$$

$$+ c_2(1 + f'(\xi))T_{n-1})c_{n-1} + O(c_{n-1}^2),$$ \hspace{1cm} (44)
According to (10), (13), (14) and (18), we have
\[ c_2(1 + f'(\xi)) - T_n = -(c_3(2 + 3f'(\xi) + f'(\xi)^2) - c_2^2(3 + 4f'(\xi) + 2f'(\xi)^2) \\
+ c_2(1 + f'(\xi))T_{n-1})c_{n-1} + O(c_{n-1}^2). \]  \hfill (45)

Using (24), (25) and (43), we get
\[ e_{n,y} \sim (c_2 + c_2f'(\xi) - T_n)c_{n}^2 \]
\[ \sim -(c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 3f'(\xi) + f'(\xi)^2))c_{n-1}(D_{n-1,r}c_{n-1})^2 \]
\[ \sim -(c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 3f'(\xi) + f'(\xi)^2))D_{n-1,r}^2c_{n-1}^{2r+1}, \] \hfill (46)
\[ e_{n+1} \sim (c_2 + c_2f'(\xi) - T_n)(c_2^2(1 + f'(\xi)) - c_3(1 + f'(\xi)) - c_2T_n)c_{n}^4 \]
\[ \sim -c_3(1 + f'(\xi))(c_2^2(2 + 2f'(\xi) + f'(\xi)^2) - c_3(2 + 3f'(\xi) + f'(\xi)^2))c_{n-1}(D_{n-1,r}c_{n-1})^4 \]
\[ \sim -c_3(1 + f'(\xi))(c_2^2(2 + 2f'(\xi) + f'(\xi)^2) - c_3(2 + 3f'(\xi) + f'(\xi)^2))D_{n-1,r}^{4r+1}. \] \hfill (47)

Comparing exponents of \( e_{n-1} \) in relations ((23), (46)) and ((22), (47)), we get
\[ \begin{align*}
2r + 1 &= rp, \\
4r + 1 &= r^2.
\end{align*} \] \hfill (48)

Solving system (48), we obtain \( r = 2 + \sqrt{3} \approx 4.2361 \) and \( p = \sqrt{5} \approx 2.2361 \). Therefore, the R-order of method (20), when \( T_n \) is calculated by (35), is at least 4.2361. From (43) and (45), we know that self-accelerating parameters (35) and (36) have the same error level. So, the R-order of method (20), when \( T_n \) is calculated by (36), is at least 4.2361.

The proof is completed. \( \square \)

**Theorem 4.** Let varying parameters \( T_n \) be calculated by (37) or (38) in method (20), respectively. If an initial value \( x_0 \) is sufficiently close to a simple zero \( \xi \) of \( f(x) \), then the R-order of convergence of iterative method (20) is at least \( 2 + \sqrt{6} \approx 4.4495 \).

**Proof.** According to (10), (13), (14) and (18), we have
\[ x_n - y_n = -(c_2(1 + f'(\xi)) + T_{n-1})c_{n-1}^2 + (c_2^2(2 + 2f'(\xi) + f'(\xi)^2) - c_3(2 + 3f'(\xi) + f'(\xi)^2))c_{n-1}^2 \]
\[ -2c_2(1 + f'(\xi))T_{n-1}c_{n-1}^2 + (-c_2^2(3 + 3f'(\xi) + 2f'(\xi)^2 + f'(\xi)^3) + c_2^2(3 + 4f'(\xi)) \]
\[ + 3f'(\xi)^2)T_{n-1} - (1 + f'(\xi))(c_4(3 + 3f'(\xi) + f'(\xi)^2) + c_3(3 + 2f'(\xi))T_{n-1}) \]
\[ + c_2(2c_3(3 + 4f'(\xi) + 3f'(\xi)^2 + f'(\xi)^3) + 7f'(\xi)^2))c_{n-1}^2 + O(c_{n-1}^3). \] \hfill (49)
\[ z_n - x_n = -c_2(1 + f'(\xi))c_{n-1}^2 + (-c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_2(2 + 3f'(\xi) + f'(\xi)^2))c_{n-1}^2 \]
\[ + (c_2^2(4 + 5f'(\xi) + 3f'(\xi)^2 + f'(\xi)^3) + c_4(3 + 6f'(\xi) + 4f'(\xi)^2 + f'(\xi)^3) \]
\[ - c_2c_3(7 + 10f'(\xi) + 7f'(\xi)^2 + 2f'(\xi)^3))c_{n-1}^2 + O(c_{n-1}^3). \] \hfill (50)
\[ y_{n-1} - w_{n-1} = (-1 - f'(\xi))c_{n-1} + (c_2 - T_{n-1})c_{n-1}^2 + (-c_2^2(2 + 2f'(\xi) + f'(\xi)^2) + c_3(2 + 2f'(\xi) \]
\[ + f'(\xi)^2)) + c_2(1 + f'(\xi))T_{n-1})c_{n-1}^2 + O(c_{n-1}^3). \] \hfill (51)
\[ z_{n-1} - w_{n-1} = (-1 - f'(\xi))c_{n-1} + c_2c_{n-1}^2 - (c_2^2 - c_3)(2 + 2f'(\xi) + f'(\xi)^2)c_{n-1}^2 + (c_2^2(4 \]
\[ + 5f'(\xi) + 3f'(\xi)^2 + f'(\xi)^3) + c_3(3 + 5f'(\xi) + 4f'(\xi)^2 + f'(\xi)^3) \]
\[ - c_2c_3(7 + 10f'(\xi) + 7f'(\xi)^2 + 2f'(\xi)^3))c_{n-1}^2 + O(c_{n-1}^3). \] \hfill (52)
\[w_{n-1} - x_{n-1} = f'(\xi)e_{n-1} + c_2f'(\xi)e_{n-1}^2 + c_3f'(\xi)e_{n-1}^3 + c_4f'(\xi)e_{n-1}^4 + O(e_{n-1}^5),\]

\[z_n - x_n = -(c_2 + c_2f'(\xi) - T_{n-1})(c_2^2(1 + f'(\xi)) - c_3(1 + f'(\xi)) - c_4T_{n-1})e_{n-1}^4 + O(e_{n-1}^5),\]

\[z_n - w_n = -(1 + f'(\xi))(c_2 + c_2f'(\xi) - T_{n-1})(c_2^2(1 + f'(\xi)) - c_3(1 + f'(\xi)) - c_4T_{n-1})e_{n-1}^4 + O(e_{n-1}^5).\]

According to (37) and (49)–(55), we get

\[T_n = \frac{z_n - x_n}{x_n - x_{n-1}} \frac{x_n + y_{n-1} - w_{n-1} - z_n}{z_n - x_{n-1}} - \frac{2z_n - x_n - y_n}{(x_n - y_{n-1})(z_n - x_{n-1})^2} \frac{(x_n - y_{n-1})^2(1 + f'(\xi))}{(z_n - x_{n-1})^2(z_n - x_{n-1})^2},\]

where

\[A_1 = (-c_2^2(1 + 3f'(\xi) + 6f'(\xi)^2 + 3f'(\xi)^3) + c_2^2(2 + 2f'(\xi) + f'(\xi)^2)T_{n-1} + (1 + f'(\xi))(c_4 + c_4f'(\xi) + c_3T_{n-1}) + c_2(c_3f'(\xi)(3 + 3f'(\xi) + f'(\xi)^2) + T_{n-1}^2)),\]

\[c_2(1 + f'(\xi)) - T_n = A_1e_{n-1}^2 + O(e_{n-1}^3),\]

Using (24), (25) and (58), we obtain

\[e_{n,y} \sim (c_2 + c_2f'(\xi) - T_n)e_{n}^2 \sim A_1e_{n-1}^2(D_{n-1,r}e_{n-1}^r)^2 = A_1D_{n-1}^2e_{n-1}^{2r+2},\]

\[e_{n+1} \sim (c_2 + c_2f'(\xi) - T_n)[c_2^2(1 + f'(\xi)) - c_3(1 + f'(\xi)) - c_4T_n]e_{n}^4 \sim -A_1c_3(1 + f'(\xi))e_{n-1}^2(D_{n-1,r}e_{n-1}^r)^4 = -A_1c_3(1 + f'(\xi))D_{n-1}^4e_{n-1}^{4r+4}.\]

By comparing exponents of \(e_{n-1}\) in relations ((23), (59)) and ((22), (60)), we get

\[\begin{cases} 2r + 2 = r_p, \\ 4r + 2 = r^2. \end{cases}\]

Solving system (61), we get \(r = 2 + \sqrt{5} \approx 4.4495\) and \(p = \sqrt{5} \approx 2.4495\). Therefore, the \(R\)-order of method (20) is at least 4.4495, when \(\lambda_n\) is calculated by (37). According to (38) and (49)–(55), we get

\[T_n = \frac{(z_n - x_{n-1})(y_{n-1} - x_{n-1})}{(x_n - x_{n-1})^2} \frac{x_n + y_{n-1} - w_{n-1} - z_n}{z_n - x_{n-1}} - \frac{2z_n - x_n - y_n}{(x_n - y_{n-1})(z_n - x_{n-1})^2} \frac{(x_n - y_{n-1})^2(1 + f'(\xi))}{(z_n - x_{n-1})^2(z_n - x_{n-1})^2},\]

where

\[A_2 = (c_2c_3f'(\xi)(3 + 3f'(\xi) + f'(\xi)^2) - c_2^2(1 + 4f'(\xi) + 8f'(\xi)^2 + 4f'(\xi)^3) + c_2^2(3 + 4f'(\xi) + 2f'(\xi)^2)\lambda_{n-1} + (1 + f'(\xi))(c_4 + c_4f'(\xi) + c_3T_{n-1})),\]

\[c_2(1 + f'(\xi)) - T_n = A_2e_{n-1}^2 + O(e_{n-1}^3).\]

Equations (58) and (64) have the same error level. Therefore, the \(R\)-order of method (20), when \(T_n\) is calculated by (38), is at least 4.4495. The proof is completed.
Remark 2. Theorems 3 and 4 prove that the self-accelerating parameters \((35)-(38)\) of method \((20)\) are efficient. In fact, the self-accelerating parameter \(T_n\) can be constructed by many schemes. Here, we get some other schemes as follows:

\[
\begin{align*}
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})^2}, \\
T_n &= \frac{z_{n-1} - x_n}{(y_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(y_{n-1} - x_{n-1})^2}, \\
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(y_{n-1} - x_{n-1}) (y_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1}) (y_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1}) (y_{n-1} - x_{n-1})}, \\
T_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1}) (y_{n-1} - x_{n-1})}. \\
\end{align*}
\]

Using the results of Theorem 3, we can prove that the R-order of method \((20)\), is at least 4.2361, when \(T_n\) is calculated by any schemes of \((65)-(73)\). Here, we omit the proving process.

4. Numerical Results

The new methods \((7)\) and \((20)\) are employed to solve nonlinear equations \(f_i(x) \quad (i = 1, 2, 3)\) and compared with Ren’s method \((\alpha = 0)\) (RM, \((6)\)), Petković’s method \((\text{PM}, (2))\), Zheng’s method \((\text{ZM}, (3))\) and Wang’s method \((\text{WM}, (4))\).

Tables 1–3 show the absolute errors \(|x_k - \xi|\) in the first four iterations, where the root \(\xi\) is computed with 1200 significant digits. The first iteration uses initial parameters \(T = 0.1\) and \(T_0 = 0.1\). The approximate computational order of convergence \(\rho\) is defined by \([24]\):

\[
\rho \approx \frac{\ln(|x_{n+1} - x_n| / |x_n - x_{n-1}|)}{\ln(|x_n - x_{n-1}| / |x_{n-1} - x_{n-2}|)},
\]

Test functions are used as follows:

\[
\begin{align*}
f_1(x) &= \cos(x) - x, \quad \xi \approx 0.7390851332151606, \quad x_0 = 0.5. \\
f_2(x) &= 10x e^{-x^2} - 1, \quad \xi \approx 1.6796306104284499, \quad x_0 = 1.8. \\
f_3(x) &= \sin(x) - \frac{1}{3}x, \quad \xi \approx 2.2788626600758283, \quad x_0 = 2.0.
\end{align*}
\]
Tables 1–3 show that our methods have better convergence behaviors than some existing methods in this paper. Numerical results prove the validity of theory.

**Remark 3.** If we use high-order Newton’s interpolation polynomial to calculate the self-accelerating parameter, our method (20) will obtain higher convergence order. High-order interpolation polynomial is complex, which is disadvantageous to reduce computation. Therefore, we give a simple way to construct self-accelerating parameter.

5. Conclusions

In this paper, a new self-accelerating type method is proposed for solving nonlinear equations, which is a modified scheme of Ren’s method. The new method reaches the highest convergence order 4.4495 by using a self-accelerating parameter. More importantly, a novel technique is applied to construct a self-accelerating parameter, which is different from the secant method and interpolation method. The new self-accelerating parameter does not increase the computational cost of the new
method. The new method is compared in performance with the existing methods. Numerical examples confirm the validity of theoretical results and show that the new method has a better convergence feature than some existing methods.

**Author Contributions:** methodology, X.W., Q.F.; writing—original draft preparation, X.W.; writing—review and editing, X.W., Q.F. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the National Natural Science Foundation of China (Nos. 61976027 and 61572082), Natural Science Foundation of Liaoning Province of China (No. 20180551262), Educational Commission Foundation of Liaoning Province of China (No. LJ2019010) and University-Industry Collaborative Education Program (No.201901077017).

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

**References**

1. Traub, J.F. *Iterative Method for the Solution of Equations*; Prentice Hall: New York, NY, USA, 1964.
2. Petković, M.S.; Ilić, S.; Džunić, J. Derivative free two-point methods with and without memory for solving nonlinear equations. *Appl. Math. Comput.* 2010, 217, 1887–1895. [CrossRef]
3. Zheng, Q.; Zhao, P.; Zhang, L.; Ma, W. Variants of Steffensen-secant method and applications. *Appl. Math. Comput.* 2010, 216, 3486–3496. [CrossRef]
4. Wang, X.; Zhang, T. A new family of Newton-type iterative methods with and without memory for solving nonlinear equations. *Calcolo* 2014, 51, 1–15. [CrossRef]
5. Wang, X.; Zhang, T. Some Newton-type iterative methods with and without memory for solving nonlinear equations. *Int. J. Comput. Methods* 2014, 11, 1350078. [CrossRef]
6. Wang, X.; Qin, Y.; Qian, W.; Zhang, S.; Fan, X. A Family of Newton Type Iterative Methods for Solving nonlinear Equations. *Algorithms* 2015, 8, 786–798. [CrossRef]
7. Džunić, J.; Petković, M.S.; Petković, L.D. Three-point methods with and without memory for solving nonlinear equations. *Appl. Math. Comput.* 2012, 218, 4917–4927. [CrossRef]
8. Džunić, J.; Petković, M.S. On generalized multipoint root-solvers with memory. *J. Comput. Appl. Math.* 2015, 236, 2909–2920. [CrossRef]
9. Zaka Ullah, M.; Kosari, S.; Soleymani, F.; Khaksar Haghani, F.; Al-Fhaid, A.S. A super-fast tri-parametric iterative method with memory. *Appl. Math. Comput.* 2016, 289, 486–491.
10. Lotfi, T.; Assari, P. New three- and four-parametric iterative with memory methods with efficiency index near 2. *Appl. Math. Comput.* 2015, 270, 1004–1010. [CrossRef]
11. Cordero, A.; Lotfi, T.; Bakhtiari, P.; Torregrosa, J.R. An efficient two-parametric family with memory for nonlinear equations. *Numer. Algorithms* 2015, 68, 323–335. [CrossRef]
12. Soleymani, F.; Lotfi, T.; Tavakoli, E.; Haghani, F.K. Several iterative methods with memory using accelerators. *Appl. Math. Comput.* 2015, 254, 452–458. [CrossRef]
13. Sharma, J.R.; Guha, R.K.; Gupta, P. Some efficient derivative free methods with memory for solving nonlinear equations. *Appl. Math. Comput.* 2012, 219, 699–707. [CrossRef]
14. Wang, X.; Zhang, T.; Qin, Y. Efficient two-step derivative-free iterative methods with memory and their dynamics. *Int. J. Comput. Math.* 2016, 93, 1423–1446. [CrossRef]
15. Wang, X.; Zhang, T. Efficient n-point iterative methods with memory for solving nonlinear equations. *Numer. Algorithms* 2015, 70, 357–375. [CrossRef]
16. Campos, B.; Cordero, A.; Torregrosa, J.R.; Vindel, P. A multidimensional dynamical approach to iterative methods with memory. *Appl. Math. Comput.* 2015, 217, 701–715. [CrossRef]
17. Cordero, A.; Jordan, C.; Torregrosa, J.R. A dynamical comprison between iterative methods with memory: Are the derivative good for the memory? *J. Comput. Appl. Math.* 2017, 318, 335–347. [CrossRef]
18. Petković, M.S.; Neta, B.; Petković, L.D.; Džunić, J. Multipoint methods for solving nonlinear equations: A survey. *Appl. Math. Comput.* 2014, 226, 635–660. [CrossRef]
19. Iliev, A.; Kyurkchiev, N. *Nontrivial Methods in Numerical Analysis: Selected Topics in Numerical Analysis*; Lap Lambert Academic Publishing: Saarbrucken, Germany, 2010.
20. Wang, X. A new Newton method with memory for solving nonlinear equations. *Mathematics* **2020**, *8*.

21. Wang, X. A new accelerating technique applied to a variant of Cordero-Torregrosa method. *J. Comput. Appl. Math.* **2018**, *330*, 695–709. [CrossRef]

22. Ren, H.; Wu, Q.; Bi, W. A class of two-step Steffensen type methods with fourth-order convergence. *Appl. Math. Comput.* **2009**, *209*, 206–210. [CrossRef]

23. Kung, H.T.; Traub, J.F. Optimal order of one-point and multipoint iterations. *J. Appl. Comput. Math.* **1974**, *21*, 643–651. [CrossRef]

24. Cordero, A.; Torregrosa, J.R. Variants of Newton’s Method using fifth-order quadrature formulas. *Appl. Math. Comput.* **2007**, *190*, 686–698. [CrossRef]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).