mathematics

Article

A New Newton Method with Memory for Solving Nonlinear Equations

Xiaofeng Wang * and Yuxi Tao

School of Mathematics and Physics, Bohai University, Jinzhou 121000, China; yxtbhu@163.com
* Correspondence: w200888w@163.com

Received: 27 November 2019; Accepted: 29 December 2019; Published: 10 January 2020

Abstract: A new Newton method with memory is proposed by using a variable self-accelerating parameter. Firstly, a modified Newton method without memory with invariant parameter is constructed for solving nonlinear equations. Substituting the invariant parameter of Newton method without memory by a variable self-accelerating parameter, we obtain a novel Newton method with memory. The convergence order of the new Newton method with memory is $1+\sqrt{2}$. The acceleration of the convergence rate is attained without any additional function evaluations. The main innovation is that the self-accelerating parameter is constructed by a simple way. Numerical experiments show the presented method has faster convergence speed than existing methods.

Keywords: simple roots; newton method; nonlinear equation; self-accelerating parameter; computational efficiency

MSC: 65H05; 65B99

---

1. Introduction

Using information from the current and previous iterations, iterative methods with memory for solving nonlinear equations can attain high convergence order and computational efficiency without any additional function evaluations. Traub [1] first proposed the following iterative method with memory with the convergence order $1+\sqrt{2}\approx 2.414$.

$$\begin{cases} w_n = x_n + T_n f(x_n),\ T_n = -\frac{1}{f[x_n,x_{n-1}]}, \\ x_{n+1} = x_n - \frac{f(x_n)}{f[x_n,w_n]}, \end{cases} \tag{1}$$

where $f[x_n,x_{n-1}] = \{f(x_n)-f(x_{n-1})\}/(x_n-x_{n-1})$ is a divided difference and the parameter T_n is called self-accelerating parameter. Method (1) is defined as TRM in this paper.

Inspired by Traub's idea, Džunić et al. [2] obtained a modified Newton method with order $1+\sqrt{2}\approx 2.414$.

$$\begin{cases} w_n = x_n + T_n f(x_n),\ T_n = -\frac{1}{2f[x_n,x_{n-1}]}, \\ x_{n+1} = x_n - \frac{f(x_n)}{f'(w_n)}, \end{cases} \tag{2}$$

where the self-accelerating parameter T_n is calculated by interpolating polynomial of Hermite–Birkhoff type. Method (2) is defined as DZM in this paper. McDougall and Wotherspoon [3] proposed the following method with convergence order $1+\sqrt{2}\approx 2.414$.

$$\begin{cases} x_n^* = x_n - \frac{f(x_n)}{f'(\frac{x_{n-1}+x_{n-1}^*}{2})}, \\ x_{n+1} = x_n - \frac{f(x_n)}{f'(\frac{x_n+x_n^*}{2})}, \end{cases} \tag{3}$$

where $x_0^* = x_0$ and $x_1 = x_0 - f(x_0)/f'(x_0)$. Method (3) is defined as MWM in this paper. Many efficiency iterative methods with memory have been studied in recent years, see [4–10]. Most of them achieved higher convergence order by using the self-accelerating parameters. The self-accelerating parameters usually are constructed by the interpolation method. In this paper, a new way to construct the self-accelerating parameter is proposed and a simple modified Newton method with memory is obtained.

The major innovative work of this paper is that we present a novel way to construct the self-accelerating parameter. In Section 2, we derive a modified Newton method with convergence order 2 for solving nonlinear equations. In Section 3, based on the modified Newton method, a new iterative method with memory is obtained by using the novel self-accelerating parameter. The order of convergence of new method with memory is increased from 2 to $1+\sqrt{2}$ without any additional functional evaluations. In Section 4, some numerical tests are used to compare the new methods with some well-known methods. Numerical experiments show that the new method has faster convergence speed than the existing methods.

2. Modified Newton Method

It is well known that Newton method [11] converges quadratically. Newton method is defined as NM in this paper. If the sequence $\{x_n\}$ is generated by NM, which converges to a simple root ζ of nonlinear equation, then the sequence $\{x_n\}$ satisfies the following relation:

$$\lim_{n\to\infty} \frac{x_{n+1}-\zeta}{(x_n-\zeta)^2} = \lim_{n\to\infty} \frac{e_{n+1}}{e_n^2} = c_2 \tag{4}$$

where $c_2 = f''(\zeta)/(2f'(\zeta))$ is the asymptotic error constant, $e_n = x_n - \zeta$ and $e_{n+1} = x_{n+1} - \zeta$.

We consider the following scheme:

$$\begin{cases} y_n = x_n - \frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} = y_n - T(y_n - x_n)^2, \end{cases} \tag{5}$$

where $T \in R$. The first step of our method (5) is the Newton method.

Theorem 1. *Let $f : I \subset R \to R$ be differentiable in an open interval I and $\zeta \in I$ be a simple zero of f. Then order of convergence of the iterative method (5) is two and its error equation meets the following equation:*

$$e_{n+1} = (c_2 - T)e_n^2 + O(e_n^3), \tag{6}$$

where $e_n = x_n - \zeta$, $c_2 = f^{(2)}(\zeta)/(2f'(\zeta))$, and $T \in R - \{0\}$.

Proof. Let $c_n = (1/n!)f^{(n)}(\zeta)/f'(\zeta)$, $n = 2, 3, \cdots$. Using the Taylor expansion of $f(x)$ around $x = \zeta$ and taking $f(\zeta) = 0$ into account, we get:

$$f(x_n) = f'(\zeta)[e_n + c_2e_n^2 + c_3e_n^3 + c_4e_n^4 + c_5e_n^5 + O(e_n^6)], \tag{7}$$

$$f'(x_n) = f'(\zeta)[1 + 2c_2e_n + 3c_3e_n^2 + 4c_4e_n^3 + 5c_5e_n^4 + O(e_n^5)]. \tag{8}$$

From Equations (5), (7) and (8), we get:

$$y_n - \zeta = x_n - \zeta - f(x_n)/f'(x_n) = c_2e_n^2 + (-2c_2^2 + 2c_3)e_n^3 + (4c_2^3 - 7c_2c_3 + 3c_4)e_n^4 + O(e_n^5). \tag{9}$$

Using Equations (5) and (9), we have:

$$e_{n+1} = x_{n+1} - \zeta = y_n - \zeta - T(y_n - x_n)^2 = (c_2 - T)e_n^2 + 2(c_3 - c_2^2 + c_2T)e_n^3 + O(e_n^4). \tag{10}$$

The proof is completed. □

3. New Newton Method with Memory

In this Section, a new way to construct the self-accelerating parameter will be given. The new Newton method (5) can be accelerated with the use of information from the current and previous iterations. The minimization of the error relation Equation (9) can be obtained by recalculating the free parameter $T = T_n = c_2$. If the variable parameter T_n satisfies $\lim_{n\to\infty} T_n = c_2$, then the asymptotic convergence constant to be zero in Equation (10). From Equation (4), $T_n = (x_{n+1} - \zeta)/(x_n - \zeta)^2$ can be the self-accelerating parameter. But, the zero ζ is unknown in Equation (4), we can use the sequence information of method (5) from the current and previous iterations to approximate ζ and obtain the new self-accelerating parameter T_n. The new self-accelerating parameter T_n is given by the following formulas:

Formula 1:

$$T_n = \frac{y_{n-1} - y_n}{(x_n - x_{n-1})^2} \tag{11}$$

Formula 2:

$$T_n = \frac{y_{n-1} - y_n}{(y_{n-1} - x_{n-1})^2} \tag{12}$$

Formula 3:

$$T_n = \frac{y_{n-1} - y_n}{(y_{n-1} - x_{n-1})(x_n - x_{n-1})} \tag{13}$$

Replacing the parameter T in Equation (5) with T_n, we obtain the following iterative method with memory:

$$\begin{cases} y_n = x_n - \frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} = y_n - T_n(y_n - x_n)^2, \end{cases} \tag{14}$$

where T_n is calculated by using one of Formulas (11)–(13). The parameter T_n depends on the iterative sequence information x_{n-1}, y_{n-1}, x_n and y_n.

The convergence order of the iterative method with memory Equation (14) will be estimated by the concept of R-order of convergence [11] and the following Theorem (see [12] (p. 287)).

Theorem 2. *If the errors of approximations $e_k = x_k - a$ obtained in an iterative method (IM) satisfy:*

$$e_{k+1} \sim \prod_{i=0}^{n} (e_{k-i})^{m_i},\ k \geq k(\{e_k\}),$$

then the R-order of convergence of IM, defined by $O_R(\mathrm{IM}, a)$, satisfies the inequality $O_R(\mathrm{IM}, a) \geq s^*$, where $m_i \in R$ is a real number and s^* is the unique positive solution of the equation $s^{n+1} - \sum_{i=0}^{n} m_i s^{n-i} = 0$.

Theorem 3. *Let the self-accelerating parameter T_n be calculated by (11), (12) or (13) in the iterative method (14), respectively. If x_0 is an initial approximation, which is sufficiently close to a simple root ζ of $f(x)$, then the R-order of convergence of the iterative methods (14) is at least $1 + \sqrt{2} \approx 2.414$.*

Proof. Let the sequence $\{x_n\}$ be generated by an iterative method, which converges to the root ζ of $f(x)$ with the R-order $O_R(\mathrm{IM}, a) \geq r$, we obtain:

$$e_{n+1} \sim D_{n,r} e_n^r,\ e_n = x_n - \zeta, \tag{15}$$

when $n \to \infty$, $D_{n,r}$ tends to the asymptotic error constant in Equation (15). Therefore,

$$e_{n+1} \sim D_{n,r}(D_{n-1,r}e_{n-1}^{r})^{r} = D_{n,r}D_{n-1,r}^{r}e_{n-1}^{r^2}. \tag{16}$$

The error relations of the method (14) with memory can be obtained by using Equation (6), which satisfies:

$$e_{n+1} = x_{n+1} - a \sim (c_2 - T_n)e_n^2 + O(e_n^3). \tag{17}$$

From Equations (8) and (9), we get:

$$y_{n-1} - y_n = c_2e_{n-1}^2 + 2(c_3 - c_2^2)e_{n-1}^3 + (3c_2^3 + 3c_4 + 2c_2^2T_{n-1} - c_2(7c_3 + T_{n-1}^2))e_{n-1}^4 + O(e_n^5), \tag{18}$$

$$x_n - x_{n-1} = -e_{n-1} + (c_2 - T_{n-1})e_{n-1}^2 + 2(-c_2^2 + c_3 + c_2T_{n-1})e_{n-1}^3 + O(e_{n-1}^4), \tag{19}$$

$$y_{n-1} - x_{n-1} = -e_{n-1} + c_2e_{n-1}^2 + 2(c_3 - c_2^2)e_{n-1}^3 + O(e_{n-1}^4). \tag{20}$$

According to Equations (11), (18) and (19), we get:

$$T_n = \frac{y_{n-1} - y_n}{(x_n - x_{n-1})^2} = c_2 + 2(c_3 - c_2T_{n-1})e_{n-1} + O(e_{n-1}^2), \tag{21}$$

$$c_2 - T_n \sim -2(c_3 - c_2T_{n-1})e_{n-1} + O(e_{n-1}^2). \tag{22}$$

From Equations (12), (18) and (20), we obtain:

$$T_n = \frac{y_{n-1} - y_n}{(y_{n-1} - x_{n-1})^2} = c_2 + 2c_3e_{n-1} + O(e_{n-1}^2), \tag{23}$$

$$c_2 - T_n \sim -2c_3e_{n-1} + O(e_{n-1}^2). \tag{24}$$

Using Equation (13), and Equations (18)–(20), we have:

$$T_n = \frac{y_{n-1} - y_n}{(y_{n-1} - x_{n-1})(x_n - x_{n-1})} = c_2 + (2c_3 - c_2T_{n-1})e_{n-1} + O(e_{n-1}^2), \tag{25}$$

$$c_2 - T_n \sim -(2c_3 - c_2T_{n-1})e_{n-1} + O(e_{n-1}^2). \tag{26}$$

According to Equations (19), (24), (25) and (27), we get:

$$e_{n+1} \sim (c_2 - T_n)e_n^2 \sim -2(c_3 - c_2T_{n-1})D_{n-1,r}^2e_{n-1}^{2r+1}. \tag{27}$$

Comparing exponents of e_{n-1} in relations Equations (16) and (27), we obtain the following equation:

$$r^2 - 2r - 1 = 0. \tag{28}$$

The positive solution of Equation (28) is given by $r = 1 + \sqrt{2} \approx 2.414$. Therefore, the R-order of convergence of the method (14), when T_n is calculated by (11), is at least 2.414.

From Equations (22), (24) and (26), we can see that Formula 1, Formula 2 and Formula 3 have the same error level. Thus, the convergence order of iterative method (14) with memory is $1 + \sqrt{2} \approx 2.414$, when Equations (12) or (13) is used to compute the parameter T_n, respectively.

This completes the proof. □

4. Numerical Examples

Now, the modified Newton method (5) without memory and method (14) with memory are used compare with Newton's method (NM), method TRM (1), method DZM (2) and method MWM

(3) for solving some nonlinear equations. Tables 1–7 give the absolute errors $|x_k - \zeta|$, where the root ζ is computed by 1200 significant digits. We use the parameters $T = 0.1$ and $T_0 = 0.1$ in the first iteration. The ρ is the computational convergence order [13], which is used to approach the theoretical convergence order of iterative method. The ρ is defined as follows:

$$\rho \approx \frac{\ln(\left|x_{n+1} - x_n\right| / |x_n - x_{n-1}|)}{\ln(|x_n - x_{n-1}| / |x_{n-1} - x_{n-2}|)}. \tag{29}$$

We use the following test functions:

$$
\begin{aligned}
&f_1(x) = e^{(x+2-x^2)} - 1, && \zeta = -1, && x_0 = -0.6. \\
&f_2(x) = \sin(x) - x/3, && \zeta \approx 2.2788626600758283, && x_0 = 3.27 \\
&f_3(x) = 10xe^{-x^2} - 1, && \zeta \approx 1.6796306104284499, && x_0 = 2.1 \\
&f_4(x) = xe^{-x^2} - \sin^2 x + 3\cos x + 5, && \zeta \approx -1.2076478271309189, && x_0 = -1.28. \\
&f_5(x) = \arcsin(x^2 - 1) - 0.5x + 1, && \zeta \approx 0.59481096839836918, && x_0 = 0.0998 \\
&f_6(x) = \ln(x^2 + x + 2) - x + 1, && \zeta \approx 4.15259073675715827, && x_0 = 2.55 \\
&f_7(x) = x^5 + x^4 + 4x^2 - 15, && \zeta \approx 1.34742809896830498, && x_0 = 1.6 \\
&f_8(x) = \ln(x^2 - 2x + 2) + \exp(x^2 - 4x + 4)\sin(x - 1), && \zeta = 1, && x_0 = 0.54 \\
&f_9(x) = x^3 - 10, && \zeta \approx 2.15443469003188372, && x_0 = 2 \\
&f_{10}(x) = x^2 \sin x - \cos x, && \zeta \approx 0.895206045384231850, && x_0 = 1
\end{aligned}
$$

Tables 1–10 show that the new Newton methods with memory (14) present an increased rate of convergence over Newton method with no additional cost. The new method (14) has higher precision than other methods.

Table 1. Numerical results for function $f_1(x)$.

Methods	$\lvert x_1-\zeta\rvert$	$\lvert x_2-\zeta\rvert$	$\lvert x_3-\zeta\rvert$	$\lvert x_4-\zeta\rvert$	ρ
NM	0.94848×10^{-1}	0.11122×10^{-1}	0.14567×10^{-3}	0.24760×10^{-7}	2.0021081
(4)	0.88625×10^{-1}	0.87717×10^{-2}	0.82591×10^{-4}	0.72764×10^{-8}	2.0013387
TRM	0.12906	0.59074×10^{-2}	0.57541×10^{-5}	0.26531×10^{-12}	2.4361321
DZM	0.14873	0.75261×10^{-2}	0.10585×10^{-4}	0.11429×10^{-11}	2.4428540
MWM	0.10080	0.53146×10^{-2}	0.50328×10^{-5}	0.21028×10^{-12}	2.4404239
((14), (11))	0.95990×10^{-1}	0.14885×10^{-2}	0.27327×10^{-6}	0.15929×10^{-15}	2.4716282
((14), (12))	0.96476×10^{-1}	0.10035×10^{-2}	0.79743×10^{-7}	0.63708×10^{-17}	2.4629052
((14), (13))	0.96229×10^{-1}	0.12496×10^{-2}	0.45916×10^{-7}	0.86370×10^{-18}	2.4185119

Table 2. Numerical results for function $f_2(x)$.

Methods	$\lvert x_1-\zeta\rvert$	$\lvert x_2-\zeta\rvert$	$\lvert x_3-\zeta\rvert$	$\lvert x_4-\zeta\rvert$	ρ
NM	0.70105×10^{-1}	0.18137×10^{-2}	0.12688×10^{-5}	0.62159×10^{-12}	1.9998571
(4)	0.12622×10^{-1}	0.46131×10^{-4}	0.60882×10^{-9}	0.10605×10^{-18}	1.9999960
TRM	0.75217×10^{-1}	0.39197×10^{-3}	0.16457×10^{-8}	0.15821×10^{-21}	2.4209303
DZM	0.77161×10^{-1}	0.42001×10^{-3}	0.19382×10^{-8}	0.23514×10^{-21}	2.4206129
MWM	0.71467×10^{-1}	0.45293×10^{-3}	0.25703×10^{-8}	0.46040×10^{-21}	2.4297966
((14), (11))	0.12626×10^{-1}	0.50711×10^{-4}	0.88637×10^{-11}	0.20764×10^{-26}	2.3130350
((14), (12))	0.12624×10^{-1}	0.48520×10^{-4}	0.64621×10^{-11}	0.44654×10^{-27}	2.3504314
((14), (13))	0.12625×10^{-1}	0.49664×10^{-4}	0.77151×10^{-11}	0.10957×10^{-26}	2.3275581

Table 3. Numerical results for function $f_3(x)$.

Methods	$\|x_1-\zeta\|$	$\|x_2-\zeta\|$	$\|x_3-\zeta\|$	$\|x_4-\zeta\|$	ρ
NM	0.30435	0.55801×10^{-1}	0.29660×10^{-2}	0.84137×10^{-5}	2.0000000
(4)	0.34603	0.72828×10^{-1}	0.56224×10^{-2}	0.33441×10^{-4}	2.0000000
TRM	0.30327	0.26131×10^{-1}	0.13826×10^{-3}	0.46450×10^{-9}	2.4143040
DZM	0.22084	0.15866×10^{-1}	0.40011×10^{-4}	0.23479×10^{-10}	2.4143179
MWM	1.0781	0.76166	0.49633×10^{-1}	0.29384×10^{-2}	2.4617091
((14), (11))	0.41260	0.11831×10^{-1}	0.79447×10^{-4}	0.46393×10^{-10}	2.4100989
((14), (12))	0.42135	0.31627×10^{-2}	0.44595×10^{-5}	0.83226×10^{-14}	2.4114555
((14), (13))	0.41681	0.76732×10^{-2}	0.30342×10^{-4}	0.20993×10^{-11}	2.4100124

Table 4. Numerical results for function $f_4(x)$.

Methods	$\|x_1-\zeta\|$	$\|x_2-\zeta\|$	$\|x_3-\zeta\|$	$\|x_4-\zeta\|$	ρ
NM	0.75636×10^{-2}	0.87698×10^{-4}	0.11555×10^{-7}	0.20057×10^{-15}	2.0000262
(4)	0.79660×10^{-2}	0.10389×10^{-3}	0.17298×10^{-7}	0.47939×10^{-15}	2.0000322
TRM	0.23918×10^{-1}	0.92015×10^{-4}	0.45676×10^{-9}	0.43316×10^{-22}	2.4552455
DZM	0.35531×10^{-1}	0.18403×10^{-3}	0.27090×10^{-8}	0.30474×10^{-20}	2.4728272
MWM	0.76346×10^{-2}	0.16742×10^{-4}	0.57306×10^{-11}	0.12575×10^{-26}	2.4218493
((14), (11))	0.80886×10^{-2}	0.18633×10^{-4}	0.38111×10^{-11}	0.12863×10^{-27}	2.4624220
((14), (12))	0.80871×10^{-2}	0.17135×10^{-4}	0.96295×10^{-11}	0.63623×10^{-26}	2.4286928
((14), (13))	0.80878×10^{-2}	0.17881×10^{-4}	0.34831×10^{-11}	0.37567×10^{-27}	2.3794489

Table 5. Numerical results for function $f_5(x)$.

Methods	$\|x_1-\zeta\|$	$\|x_2-\zeta\|$	$\|x_3-\zeta\|$	$\|x_4-\zeta\|$	ρ
NM	0.27263×10^{-1}	0.20801×10^{-3}	0.11512×10^{-7}	0.35250×10^{-16}	2.0000391
(4)	0.16256×10^{-4}	0.43857×10^{-10}	0.31925×10^{-21}	0.16916×10^{-43}	2.0000000
TRM	0.28294×10^{-1}	0.20717×10^{-4}	0.87450×10^{-12}	0.11208×10^{-29}	2.4262050
DZM	0.28942×10^{-1}	0.23529×10^{-4}	0.11550×10^{-11}	0.22203×10^{-29}	2.4238736
MWM	0.27506×10^{-1}	0.34876×10^{-4}	0.20046×10^{-11}	0.99755×10^{-29}	2.3897642
((14), (11))	0.16256×10^{-4}	0.40870×10^{-10}	0.11705×10^{-25}	0.19533×10^{-62}	2.3661816
((14), (12))	0.16256×10^{-4}	0.43873×10^{-10}	0.15341×10^{-25}	0.50629×10^{-62}	2.3602927
((14), (13))	0.16256×10^{-4}	0.42412×10^{-10}	0.13516×10^{-25}	0.32510×10^{-62}	2.4126796

Table 6. Numerical results for function $f_6(x)$.

Methods	$\|x_1-\zeta\|$	$\|x_2-\zeta\|$	$\|x_3-\zeta\|$	$\|x_4-\zeta\|$	ρ
NM	0.29535	0.49290×10^{-2}	0.14646×10^{-5}	0.12945×10^{-12}	2.0000000
(4)	0.61660×10^{-1}	0.14948×10^{-3}	0.88593×10^{-9}	0.31121×10^{-19}	2.0000000
TRM	0.27891	0.56316×10^{-3}	0.30706×10^{-9}	0.19340×10^{-24}	2.4145853
DZM	0.25753	0.45021×10^{-3}	0.18172×10^{-9}	0.54152×10^{-25}	2.4146224
MWM	0.30214	0.18597×10^{-2}	0.23420×10^{-8}	0.37578×10^{-22}	2.4118507
((14), (11))	0.61558×10^{-1}	0.25106×10^{-3}	0.11263×10^{-9}	0.65893×10^{-25}	2.4137298
((14), (12))	0.61726×10^{-1}	0.83517×10^{-4}	0.57140×10^{-11}	0.35958×10^{-28}	2.4138161
((14), (13))	0.61651×10^{-1}	0.15848×10^{-3}	0.30362×10^{-10}	0.24693×10^{-26}	2.4136285

Table 7. Numerical results for function $f_7(x)$.

Methods	$\lvert x_1-\zeta\rvert$	$\lvert x_2-\zeta\rvert$	$\lvert x_3-\zeta\rvert$	$\lvert x_4-\zeta\rvert$	ρ
NM	0.51377×10^{-1}	0.29778×10^{-2}	0.94541×10^{-5}	0.94955×10^{-10}	2.0006167
(4)	0.48084×10^{-1}	0.23460×10^{-2}	0.53104×10^{-5}	0.27139×10^{-10}	2.0004304
TRM	0.18438	0.77549×10^{-2}	0.11858×10^{-4}	0.12280×10^{-11}	2.4807622
DZM	0.21488	0.93015×10^{-2}	0.19515×10^{-4}	0.39877×10^{-11}	2.4978160
MWM	0.53263×10^{-1}	0.11016×10^{-2}	0.95638×10^{-7}	0.12209×10^{-16}	2.4360930
((14), (11))	0.50758×10^{-1}	0.32220×10^{-3}	0.62828×10^{-8}	0.10908×10^{-19}	2.4969174
((14), (12))	0.50874×10^{-1}	0.43838×10^{-3}	0.12463×10^{-7}	0.86553×10^{-19}	2.4544229
((14), (13))	0.50815×10^{-1}	0.37972×10^{-3}	0.12832×10^{-9}	0.89731×10^{-24}	2.1874410

Table 8. Numerical results for function $f_8(x)$.

Methods	$\lvert x_1-\zeta\rvert$	$\lvert x_2-\zeta\rvert$	$\lvert x_3-\zeta\rvert$	$\lvert x_4-\zeta\rvert$	ρ
NM	0.16079	0.83050×10^{-1}	0.15408×10^{-1}	0.40910×10^{-3}	2.0000000
(4)	0.15943	0.85941×10^{-1}	0.17723×10^{-1}	0.57918×10^{-3}	2.0000000
TRM	0.26086	0.10572	0.12367×10^{-1}	0.48440×10^{-4}	2.4142058
DZM	0.28894	0.11489	0.16070×10^{-1}	0.88980×10^{-4}	2.4141992
MWM	0.18521	0.69823×10^{-1}	0.46207×10^{-2}	0.58955×10^{-5}	2.4141998
((14), (11))	0.26973	0.60674×10^{-2}	0.75745×10^{-5}	0.36530×10^{-12}	2.4142457
((14), (12))	0.26546	0.17813×10^{-2}	0.72310×10^{-5}	0.52756×10^{-12}	2.4141430
((14), (13))	0.26758	0.38913×10^{-2}	0.11303×10^{-4}	0.88936×10^{-12}	2.4141334

Table 9. Numerical results for function $f_9(x)$.

Methods	$\lvert x_1-\zeta\rvert$	$\lvert x_2-\zeta\rvert$	$\lvert x_3-\zeta\rvert$	$\lvert x_4-\zeta\rvert$	ρ
NM	0.12163×10^{-1}	0.68924×10^{-4}	0.22050×10^{-8}	0.22568×10^{-17}	2.0000021
(4)	0.94218×10^{-2}	0.32385×10^{-4}	0.38193×10^{-9}	0.53121×10^{-19}	2.0000006
TRM	0.30098×10^{-1}	0.30992×10^{-4}	0.61645×10^{-11}	0.25374×10^{-27}	2.4451069
DZM	0.51419×10^{-1}	0.91890×10^{-4}	0.91924×10^{-10}	0.16729×10^{-24}	2.4567254
MWM	0.12244×10^{-1}	0.11606×10^{-4}	0.29500×10^{-12}	0.21862×10^{-30}	2.3871597
((14), (11))	0.94532×10^{-2}	0.10315×10^{-5}	0.27668×10^{-14}	0.22492×10^{-35}	2.4604765
((14), (12))	0.94518×10^{-2}	0.23608×10^{-5}	0.75329×10^{-14}	0.19241×10^{-34}	2.4237873
((14), (13))	0.94525×10^{-2}	0.17017×10^{-5}	0.17253×10^{-14}	0.36236×10^{-36}	2.4102325

Table 10. Numerical results for function $f_{10}(x)$.

Methods	$\lvert x_1-\zeta\rvert$	$\lvert x_2-\zeta\rvert$	$\lvert x_3-\zeta\rvert$	$\lvert x_4-\zeta\rvert$	ρ
NM	0.64944×10^{-2}	0.29855×10^{-4}	0.63224×10^{-9}	0.28353×10^{-18}	1.9999992
(4)	0.55399×10^{-2}	0.18642×10^{-4}	0.21175×10^{-9}	0.27319×10^{-19}	1.9999991
TRM	0.81871×10^{-2}	0.31872×10^{-5}	0.41565×10^{-13}	0.27704×10^{-32}	2.4320824
DZM	0.97808×10^{-2}	0.45593×10^{-5}	0.10149×10^{-12}	0.23626×10^{-31}	2.4348947
MWM	0.65206×10^{-2}	0.36644×10^{-5}	0.49133×10^{-13}	0.44736×10^{-32}	2.4185946
((14), (11))	0.55571×10^{-2}	0.14647×10^{-5}	0.13672×10^{-13}	0.33166×10^{-33}	2.4427552
((14), (12))	0.55575×10^{-2}	0.10649×10^{-5}	0.13873×10^{-14}	0.43888×10^{-36}	2.4197491
((14), (13))	0.55573×10^{-2}	0.12658×10^{-5}	0.61333×10^{-14}	0.34024×10^{-34}	2.4361645

Tables 11 and 12 give the mean CPU time (in seconds) of the programs after 50 performances. The stopping criterion is $\left|x_{k+1}-x_k\right|<10^{-150}$ in Table 11. The stopping criterion is $\left|x_{k+1}-x_k\right|<10^{-300}$ in Table 12.

Tables 11 and 12 show that method (14) costs less computing time than other methods. The main reason is that the structure of self-accelerating parameter of our method (14) is simple.

Table 11. Mean CPU time of iterative method for solving different functions.

f	NM	(4)	TRM	DZM	MWM	(14), (11)	(14), (12)	(14), (13)
f_1	0.5134	0.5031	0.5659	0.5744	0.5556	0.4319	0.4766	0.4766
f_2	0.5878	0.5850	0.7844	0.7350	0.6500	0.5728	0.5650	0.5566
f_3	0.8081	0.8681	0.9600	1.0516	1.1266	0.8572	0.7184	0.6994
f_4	1.2675	1.2962	1.9419	1.8603	1.4175	1.0644	1.0656	1.0181
f_5	0.5203	0.4634	0.7459	0.6128	0.5431	0.4503	0.4487	0.4409
f_6	0.6728	0.6025	0.8975	0.7478	0.7506	0.6663	0.6787	0.5956
f_7	0.5103	0.5294	0.6100	0.6184	0.5731	0.4772	0.4738	0.4781
f_8	1.4609	1.3928	1.6875	1.8528	1.6544	1.0784	1.0856	1.0734
f_9	0.6159	0.4653	0.4900	0.5288	0.3941	0.4475	0.3847	0.3697
f_{10}	0.9247	0.8631	1.2713	1.3084	1.1272	0.7844	0.7681	0.7512
Average time	0.7882	0.7569	0.9954	0.9890	0.8792	0.6830	0.6665	0.6460

Table 12. Mean CPU time of iterative method for solving different functions.

f	NM	(4)	TRM	DZM	MWM	(14), (11)	(14), (12)	(14), (13)
f_1	0.5566	0.5756	0.6578	0.6913	0.6338	0.5537	0.5613	0.5303
f_2	0.6709	0.6722	0.8000	0.7675	0.6566	0.6550	0.6359	0.6231
f_3	0.9303	0.9456	0.9384	0.9900	1.2163	0.8447	0.8584	0.8013
f_4	1.3044	1.2966	1.8487	2.0506	1.6509	1.1947	1.1206	1.1788
f_5	0.5759	0.5263	0.8594	0.7037	0.6691	0.5147	0.5144	0.5031
f_6	0.7769	0.7506	0.9684	0.7653	0.6759	0.6763	0.6866	0.6728
f_7	0.5622	0.5491	0.7256	0.6763	0.6300	0.5587	0.5622	0.5128
f_8	1.5131	1.5178	1.7459	1.8494	1.7747	1.1788	1.2019	1.1725
f_9	0.4113	0.4203	0.5397	0.4684	0.3684	0.3419	0.3322	0.3372
f_{10}	1.0009	0.8831	1.4750	1.4819	1.2506	0.8875	0.8728	0.8816
Average time	0.7372	0.7192	0.9620	0.9454	0.8310	**0.6561**	**0.6488**	**0.6412**

5. Conclusions

In this paper, we obtain a new way to construct the variable self-accelerating parameter. Using a novel self-accelerating parameter, we obtain a modified Newton method (14) with memory. By theoretical analysis and numerical experiments, we confirm that the modified Newton method with memory achieves convergence order $1+\sqrt{2}$ requiring a function and a derivative evaluation per iteration. The convergence speed of the modified Newton method (14) is faster than that of other methods. Thus, the new method (14) in this contribution can be considered as an improvement of Newton's method.

Author Contributions: Methodology, X.W., Y.T.; writing—review and editing, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was financially supported by the National Science Foundation of Liaoning Province of China (Grant No. 20180551262), the National Natural Science Foundation of China (Grants No. 61976027 and 61572028) and the Educational Commission Foundation of Liaoning Province of China (Grant No. LJ2019010).

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. Džunić, J. Modified Newton's method with memory. *Ser. Math. Inform.* **2013**, *28*, 429–441.
3. McDougall, T.J.; Wotherspoon, S.J. A simple modification of Newton's method to achieve convergence of order $1+\sqrt{2}$. *Appl. Math. Lett.* **2014**, *29*, 20–25. [CrossRef]
4. Chicharro, F.I.; Cordero, A.; Torregrosa, J.R. Dynamics of iterative families with memory based on weight functions procedure. *J. Comput. Appl. Math.* **2019**, *354*, 286–298. [CrossRef]

5. 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]
6. Cordero, A.; Lotfi, T.; Bakhtiari, P.; Torregrosa, J.R. An efficient two-parametric family with memory for nonlinear equations. *Numer. Algor.* **2015**, *68*, 323–335. [CrossRef]
7. Wang, X. An Ostrowski-type method with memory using a novel self-accelerating parameter. *J. Comput. Appl. Math.* **2018**, *330*, 710–720. [CrossRef]
8. Wang, X. A new accelerating technique applied to a variant of Cordero-Torregrosa method. *J. Comput. Appl. Math.* **2018**, *330*, 695–709. [CrossRef]
9. Wang, X. A family of Newton-type iterative methods using some special self-accelerating parameters. *Int. J. Comput. Math.* **2018**, *95*, 2112–2127. [CrossRef]
10. Soleymani, F.; Lotfi, T.; Tavakoli, E.; Khaksar Haghani, F. Several iterative methods with memory using self-accelerators. *Appl. Math. Comp.* **2015**, *254*, 452–458. [CrossRef]
11. Ortega, J.M.; Rheinbolt, W.C. *Iterative Solution of Nonlinear Equations in Several Variables*; Academic Press: New York, MY, USA, 1970.
12. Alefeld, G.; Herzberger, J. *Introduction to Interval Computation*; Academic Press: New York, NY, USA, 1983.
13. Cordero, A.; Torregrosa, J.R. Variants of Newton's Method using fifth-order quadrature formulas. *Appl. Math. Comput.* **2007**, *190*, 686–698. [CrossRef]