Research Paper
Computational Methods for Differential Equations
http://cmde.tabrizu.ac.ir
Vol. 10, No. 4, 2022, pp. 1007-1026
DOI:10.22034/cmde.2022.46651.1961

A two-step method adaptive with memory with eighth-order for solving nonlinear equations and its dynamic

Vali Torkashvand[1,2,*]

[1]Department of Mathematics, Shahre-Qods Branch, Islamic Azad University, Tehran, Iran.

[2]Department of Basic Science, Shahid Sattari Aeronatuical Uinversity of Science and Technology, Tehran, Iran.

Abstract

In this work, we have constructed the with memory two-step method with four convergence degrees by entering the maximum self-accelerator parameter(three parameters). Then, using Newton's interpolation, a with-memory method with a convergence order of 7.53 is constructed. Using the information of all the steps, we will improve the convergence order by one hundred percent, and we will introduce our method with convergence order 8. Numerical examples demonstrate the exceptional convergence speed of the proposed method and confirm theoretical results. Finally, we have presented the dynamics of the adaptive method and other without-memory methods for complex polynomials of degrees two, three, and four. The basins of attraction of existing with-memory methods are present and compared to illustrate their performance.

Keywords. Nonlinear equations, Basin of attraction, Adaptive methods , *R*-order convergence, Self accelerating parameter.
2010 Mathematics Subject Classification. 65H05, 65H99, 41A25, 65B99.

1. Introduction

In this paper, we will use memorized methods to solve nonlinear equations. Since it is almost impossible to solve nonlinear equations analytically, we have paid more attention to iterative methods with memory which efficiency index and convergence order are higher than without-memory methods of the same class. Traub can be called the first someone who proposed one-step with-memory methods that solved such equations [31]. Traub's method with-memory is as follows:

$$\begin{cases} \gamma_k = -\frac{1}{f(x_k)-f(x_{k-1})}, k = 1, 2, 3, \cdots, \\ x_{k+1} = x_k - \frac{\gamma_k f(x_k)^2}{f(x_k+\gamma_k f(x_k))-f(x_k)}, k = 0, 1, 2, \cdots. \end{cases} \tag{1.1}$$

Petkovic et al.'s [23] and Neta's method [22] can also be called one of the first two- and three-step with-memory methods. These methods have used inverse interpolation. In 2011, Petkovic et al. [23] proposed the with-memory following-methods that have the order of convergence 4.561:

$$\begin{cases} N(x) = x - \frac{f(x)}{f'(x)}, \phi(t) = \frac{1}{f(t)-f(x_k)}(\frac{t-x_k}{f(t)-f(x_k)} - \frac{1}{f'(x_k)}), y_{-1} = N(x_0), \\ y_k = N(x_k) + f^2(x_k)\phi(y_{k-1}), x_{k+1} = N(x_k) + f^2(x_k)\phi(y_k)), k = 0, 1, 2, \cdots. \end{cases} \tag{1.2}$$

Received: 22 June 2021 ; Accepted: 14 January 2022.
∗ Corresponding author. Email: torkashvand1978@gmail.com.

Also, Neta obtained a three-step with-memory method by convergence order of 10.815[22]:

$$\begin{cases} w_k = x_k - \frac{f(x_k)}{f'(x_k)} + (f(w_{k-1})\phi_z - f(z_{k-1})\phi_w)\frac{f^2(x_k)}{f(w_{k-1})-f(z_{k-1})}, \\ \phi_w = \frac{w_{k-1}-x_k}{(f(w_{k-1})-f(x_k))^2} - \frac{1}{(f(w_{k-1})-f(x_k))f'(x_k)}, \\ \phi_z = \frac{z_{k-1}-x_k}{(f(z_{k-1})-f(x_k))^2} - \frac{1}{(f(z_{k-1})-f(x_k))f'(x_k)}, \\ z_k = x_k - \frac{f(x_k)}{f'(x_k)} + (f(w_k)\phi_z - f(z_{k-1})\psi_w)\frac{f^2(x_k)}{f(w_k)-f(z_{k-1})}, \\ \psi_w = \frac{w_k-x_k}{(f(w_k)-f(x_k))^2} - \frac{1}{(f(w_k)-f(x_k))f'(x_k)}, \\ \psi_z = \frac{z_k-x_k}{(f(z_k)-f(x_k))^2} - \frac{1}{(f(z_k)-f(x_k))f'(x_k)}, \\ x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} + (f(w_k)\phi_z - f(z_k)\psi_w)\frac{f^2(x_k)}{f(w_k)-f(z_k)}. \end{cases} \tag{1.3}$$

In this work, we expand the paper presented by Lotfi et al. [17], which has a convergence order of 6, by entering two new self-acceleration parameters into with-memory methods with seventh-order and 7.53. Also, we have developed the with-memory methods with the maximum convergence order, i.e. 8, using the new adaptation technique. Now, we introduce some people who have used memorization methods to solve nonlinear equations. Lotfi-Assari [16] and Zafar et al. [36] proposed the maximum self-acceleration parameter in the without-memory. Soleymani [28] and Cordero [5, 6] have used without- and with-memory methods in solving equations that do not have exact roots. Wang [35] used Secant-type with-memory methods to solve nonlinear equations. Petkovic [24] proposed with-memory method based on weight function. Torkashvand improved the maximum efficiency index with-memory methods for solving nonlinear equations [12, 32].

Our presentation has unfolded in what follows. Section two of the article is two sub-sections. In the first part, new methods without-memory with three parameters are introduced. In the second part, with-memory methods with convergence orders of 6, 7, and 7.53 are made. In section 3, we have presented the main theorem by increasing convergence order from 4 to 8. The efficiency index of the new methods has reached the maximum possible value. In section 4, some examples are considered to illustrate our main results. One can show the dynamical behavior of the with-memory method in section 5. Concluding remarks are given in section 6.

2. New Family Without- and With-Memory Methods

2.1. **Without-Memory Methods.** In this section, we have derived a new family of iterative methods derivative-free and maintain high-order convergence. In 2014, Lotfi et al. in [17] proposed some one-parametric two-step optimal iterative methods without-memory for nonlinear equations:

$$\begin{cases} w_k = x_k + \lambda f(x_k), p1(t_k) = \frac{t_k-w_k}{x_k-w_k}f(x_k) + \frac{t_k-x_k}{w_k-x_k}f(w_k), \\ p2(t_k) = \frac{(t_k-w_k)(t_k-y_k)}{(x_k-w_k)(x_k-y_k)}f(x_k) + \frac{(t_k-x_k)(t_k-y_k)}{(w_k-x_k)(w_k-y_k)}f(w_k) + \frac{(t_k-x_k)(t_k-w_k)}{(y_k-x_k)(y_k-w_k)}f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k)}, k = 0, 1, 2, \cdots. \end{cases} \tag{2.1}$$

We show this method with T4.

Theorem 2.1. *Let $I \subseteq \mathbf{R}$ be an open interval, $f : I \to \mathbf{R}$ be a scalar function which has a simple root α in the open-interval I, and also the initial approximation x_0 is sufficiently close to the simple zero, then, the iteration methods (2.1) four-order satisfies the following errors equation [17]:*

$$e_{k+1} = (1 + \lambda f'(\alpha))^2 c_2(c_2^2 - c_3)e_k^4 + O(e_k^5), \tag{2.2}$$

where $c_k = \frac{f^{(k)}(\alpha)}{k!f'(\alpha)}$ for $k = 2, 3, \cdots$.

Creating with-memory methods that have their maximum accelerator parameter is our motivation. Because by using these parameters, convergence order can improve up to one hundred. For this purpose, we first introduce the following two-step method by entering a self-accelerator parameter (γ):

$$\begin{cases} w_k = x_k + \lambda f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k) + \gamma f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k)}, k = 0, 1, 2, \cdots . \end{cases} \tag{2.3}$$

Theorem 2.2. *Let $I \subseteq \mathbf{R}$ be an open-interval, $f : I \to \mathbf{R}$ be a scalar function which has a simple root α in the open interval I, and also the initial approximation x_0 is sufficiently close to the simple zero, then, the iteration methods (2.3) four-order satisfies the following errors equation:*

$$e_{k+1} = (1 + \lambda f'(\alpha))^2(\gamma + c_2)(c_2(\gamma + c_2) - c_3)e_k^4 + O(e_k^5). \tag{2.4}$$

Proof. By using Taylor's expansion of $f(x)$ about α and taking into account that $f(\alpha) = 0$, we obtain:

$$f(x_k) = f(\alpha)(e_k + c_2 e_k^2 + c_3 e_k^3 + c_4 e_k^4 + O(e_k^5)). \tag{2.5}$$

Then, computing $e_{k,w} = w_k - \alpha$, we attain $w_k = x_k + \lambda f(x_k)$:

$$e_{k,w} = e_k + e_k f'(\alpha)\lambda + \lambda f'(\alpha)c_2 e_k^2 + \lambda f'(\alpha)c_3 e_k^3 + \lambda f'(\alpha)c_4 e_k^4 + O(e_k^5), \tag{2.6}$$

and

$$\begin{aligned} f(w_k) &= f'(\alpha)(e_k + \lambda f'(\alpha)e_k(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3) + c_2(e_k + \lambda f'(\alpha)e_k \\ &\quad (1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3))^2 + c_3(e_k + \lambda f'(\alpha)e_k(1 + e_k c_2 + c_3 e_k^2 \\ &\quad + c_4 e_k^3))^3 + c_4(e_k + \lambda f'(\alpha)e_k(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3))^4) + O(e_k^5). \end{aligned} \tag{2.7}$$

Now by the Eqs. (2.5), (2.6), and (2.7), we get :

$$\begin{aligned} p1'(w_k) &= (-e_k^2 c_2 - e_k^3 c_3 - e_k^4 c_4 + e_k f'(\alpha)\lambda(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3) \\ &\quad + c_2(e_k + e_k f'(\alpha)\lambda(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3))^2 + c_3(e_k + e_k f'(\alpha)\lambda(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3))^3 \\ &\quad + c_4(e_k + e_k f'(\alpha)\lambda(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3))^4)(e_k\lambda(1 + e_k c_2 + c_3 e_k^2 + c_4 e_k^3))^{-1} + O(e_k^5). \end{aligned} \tag{2.8}$$

Furthermore, using $e_{k,y} = y_k - \alpha$, we have:

$$\begin{aligned} y_k &= \alpha + \frac{f(x_k)}{p1'(w_k) + \gamma f(w_k)} \\ &= (1 + \lambda f'(\alpha))(\gamma + c_2)e_k^2 + (-\gamma(2 + 2\lambda f'(\alpha) + (\lambda f'(\alpha))^2)c_2 - (2 + 2\lambda f(\alpha) + (\lambda f'(\alpha))^2)c_2^2 - \\ &\quad (1 + \lambda f'(\alpha))(\gamma^2(1 + \lambda f(\alpha)) - (2 + \lambda f'(\alpha))c_3))e_k^3 + (\gamma(5 + 7\lambda f'(\alpha) + 4(\lambda f'(\alpha))^2 + (\lambda f'(\alpha))^3)c_2^2 \\ &\quad + (4 + 5\lambda f'(\alpha) + 3(\lambda f'(\alpha))^2 + (\lambda f'(\alpha))^3)c_2^3 - \gamma(4 + 7\lambda f'(\alpha) + 5(\lambda f'(\alpha))^2 + (\lambda f'(\alpha))^3)c_3 \\ &\quad + c_2(\gamma^2(3 + 5\lambda f'(\alpha) + 3(\lambda f'(\alpha))^2 + (\lambda f'(\alpha))^3 - (7 + 10\lambda f'(\alpha) + 7(\lambda f'(\alpha))^2 + 2(\lambda f'(\alpha))^3)c_3) \\ &\quad + (1 + \lambda f'(\alpha))(\gamma^3(1 + \lambda f'(\alpha))^2 + (3 + 3\lambda f'(\alpha) + (\lambda f'(\alpha))^2)c_4))e_k^4 + O(e_k^5). \end{aligned} \tag{2.9}$$

For $f(y_k)$, we also have:

$$\begin{aligned} f(y_k) &= f'(\alpha)(1+\lambda f'(\alpha))(\gamma+c_2)e_k^2 - f'(\alpha)((\gamma+\gamma\lambda f'(\alpha))^2 + (2+\lambda f'(\alpha)(2+\lambda f'(\alpha)))c_2 \\ & (\gamma+c_2) - 2c_3 - \lambda f'(\alpha)(3+\lambda f'(\alpha))c_3)e_k^3 + f'(\alpha)((\gamma+\gamma\lambda f'(\alpha))^3 + \gamma(7+\lambda f'(\alpha) \\ & (11+\lambda f'(\alpha)(6+\lambda f'(\alpha))))c_2^2 + (5+\lambda f'(\alpha)(7+\lambda f'(\alpha)(4+\lambda f'(\alpha))))c_2^3 - \gamma(4+\lambda f'(\alpha) \\ & (7+\lambda f'(\alpha)(5+\lambda f'(\alpha))))c_3 + c_2(\gamma^2(1+\lambda f'(\alpha))(4+\lambda f'(\alpha)(3+\lambda f'(\alpha))) - (7+\lambda f'(\alpha) \\ & (10+\lambda f'(\alpha)(7+2\lambda f'(\alpha))))c_3 + (1+\lambda f'(\alpha))(3+\lambda f'(\alpha)(3+\lambda f'(\alpha)))c_4)e_k^4 + O(e_k^5). \end{aligned} \tag{2.10}$$

Additionally, by using relations (2.5)-(2.10), we gain:

$$\begin{aligned} p2'(y_k) &= f'(\alpha)(1+\lambda f'(\alpha))(2c_2(\gamma+c_2)-c_3)e_k^2 - f'(\alpha)(2\gamma(2+\lambda f'(\alpha)(2+\lambda f'(\alpha)))c_2^2 \\ & +2(2+\lambda f'(\alpha)(2+\lambda f'(\alpha)))c_2^3 + c_2(2(\gamma+\gamma\lambda f'(\alpha))^2 - (6+\lambda f'(\alpha)(8+3\lambda f'(\alpha)))c_3 \\ & -(1+\lambda f'(\alpha))(2+\lambda f'(\alpha))(\gamma c_3 - c_4)e_k^3 + f'(\alpha)(2\gamma(5+\lambda f'(\alpha)(7+\lambda f'(\alpha) \\ & \times(4+\lambda f'(\alpha))))c_2^3 + 2(4+\lambda f'(\alpha)(5+\lambda f'(\alpha)(3+\lambda f'(\alpha))))c_2^4 - \lambda f'(\alpha)(\gamma+\gamma\lambda f'(\alpha))^2 c_3 \\ & +(4+\lambda f'(\alpha)(7+\lambda f'(\alpha)(5+\lambda f'(\alpha))))c_3^2 + c_2^2(2\gamma^2(1+\lambda f'(\alpha))(3+\lambda f'(\alpha)(2+\lambda f'(\alpha))) \\ & -(16+\lambda f'(\alpha)(21+5\lambda f'(\alpha)(3+\lambda f'(\alpha))))c_3) + c_2(2(\gamma+\gamma\lambda f'(\alpha))^3 - \gamma(8+\lambda f'(\alpha)(11 \\ & +3\lambda f'(\alpha)(3+\lambda f'(\alpha))))c_3) + (8+\lambda f'(\alpha))(13+\lambda f'(\alpha)(3+\lambda f'(\alpha))))c_4) + (1+\lambda f'(\alpha)) \\ & \times(\gamma(2+\lambda f'(\alpha)(2+\lambda f'(\alpha)))c_4 - (3+\lambda f'(\alpha)(3+\lambda f'(\alpha)))c_5))e_k^4 + O(e_k^5). \end{aligned} \tag{2.11}$$

Finally, the error equation of the without memory method (2.3) will be as follows:

$$e_{k+1} = (1+\lambda f'(\alpha))^2(\gamma+c_2)(c_2(\gamma+c_2)-c_3)e_k^4 + O(e_k^5). \tag{2.12}$$

This complets the proof. □

Then, by introducing another self-accelerating parameter, we obtain the following tri-parameter without-memory method:

$$\begin{cases} w_k = x_k + \lambda f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), \\ p2(t_k) = \frac{(t_k-w_k)(t_k-y_k)}{(x_k-w_k)(x_k-y_k)} f(x_k) + \frac{(t_k-x_k)(t_k-y_k)}{(w_k-x_k)(w_k-y_k)} f(w_k) + \frac{(t_k-x_k)(t_k-w_k)}{(y_k-x_k)(y_k-w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k)+\gamma f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k)+\beta(y_k-x_k)(y_k-w_k)}, k = 0,1,2,\cdots. \end{cases} \tag{2.13}$$

Theorem 2.3. *Let $I \subseteq \mathbf{R}$ be an open-interval, $f : I \to \mathbf{R}$ be a scalar function which has a simple root α in the open interval I, and also the initial approximation x_0 is sufficiently close to the simple zero, then, the iteration methods (2.13) four-order satisfies the following errors equation:*

$$e_{k+1} = \frac{(1+\lambda f'(\alpha))^2(\gamma+c_2)(\beta+f'(\alpha)c_2(\gamma+c_2)-f'(\alpha)c_3)}{f'(\alpha)} e_k^4 + O(e_k^5). \tag{2.14}$$

Proof. We use the self-explained mathematical approach to avoid tedious and humdrum algebraic manipulation. First, we define Taylor's series of $f(x)$ as follows:

$$In[1] : f[e\_] = fla(e + c_2e^2 + c_3e^3 + c_4e^4);$$

where $e = x - \alpha$, $fla = f'(\alpha)$. Note that since α is a simple zero of $f(x)$, then $f'(\alpha) \neq 0$, $f(\alpha) = 0$. We define

$$In[2] : f[x_-, y_-] = \frac{f[x] - f[y]}{x - y};$$
$$In[3] : ew = e + \lambda f[e]//FullSimplify$$

$$Out[3] = (1 + \lambda fla)e_k + O(e_k^2) \tag{2.15}$$
$$In[4] : p1[t_-] = \frac{t - ew}{e - ew} f[e] + \frac{t - e}{ew - e} f[ew];$$
$$In[5] : ey = e - Series[\frac{f[e]}{p1'[ew] + \gamma f[ew]}, \{e, 0, 4\}]//FullSimplify$$

$$Out[5] = (1 + \lambda fla)(\gamma + c_2)e_k^2 + O(e_k^3) \tag{2.16}$$
$$In[6] : p2[t_-] = \tfrac{(t-ew)(t-ey)}{(e-ey)(e-ew)} f[e] + \tfrac{(t-e)(t-ey)}{(ew-ey)(ew-e)} f[ew] + \tfrac{(t-ew)(t-e)}{(ey-e)(ey-ew)} f[ey]$$
$$In[7] : e_{k+1} = ey - Series[\tfrac{f[ey]}{p2'[ey]+\beta(ey-e)(ey-ew)}, \{e, 0, 4\}]//FullSimplify$$

$$Out[7] = e_{k+1} = \tfrac{1}{fla}(1 + \lambda fla)^2(\gamma + c_2)(\beta + flac_2(\gamma + c_2) - flac_3)e_k^4 + O(e_k^5). \tag{2.17}$$

This complets the proof. □

In the next section, we are going to construct new iterative methods with-memory from (2.1), (2.3), and (2.13) using self-accelerating parameters.

2.2. **With-Memory Methods.** We observe from (2.2) that the convergence order of the family (2.1) is four when $\lambda \neq \frac{-1}{f'(\alpha)}$. With the choice $\lambda = \frac{-1}{f'(\alpha)}$, it can be proved that the order of the family (2.1) would be 6. However, the exact value of $f'(\alpha)$ is not available in practice and such acceleration of convergence can not be realized. Lotfi et al. approximated the parameter λ by λ_k and $\lambda_k = \frac{-1}{N_3'(x_k)} \approx \frac{-1}{f'(\alpha)}$, where $N_3(t) = N_3(t; x_k, x_{k-1}, w_{k-1}, y_{k-1})$. Finally, they one-parameter family with-memory method proposed(LSNKKM):

$$\begin{cases} \lambda_k = -\frac{1}{N_3'(x_k)}, k = 1, 2, 3, \cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), y_k = x_k - \frac{f(x_k)}{p1'(w_k)}, \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k)}, k = 0, 1, 2, \cdots. \end{cases} \tag{2.18}$$

Theorem 2.4. *If an initial estimation x_0 is close enough to a simple root α of $f(x) = 0$ being f a real sufficiently differentiable function, then the R-order of convergence of the two-point method with memory (2.18) is at least 6 [17].*

Theorem 2.2 states that prposed method (2.3) have order of convergence 4, if $\lambda \neq \frac{-1}{f'(\alpha)}$, and $\gamma \neq -c_2 = -\frac{f''(\alpha)}{2f'(\alpha)}$. Now if we approximate parameters λ and γ with λ_k, and γ_k, then, using approximations: $\lambda_k = \frac{-1}{f'(\alpha)} \approx \frac{-1}{N_3'(x_k)}$, and $\gamma_k = -\frac{f''(\alpha)}{2f'(\alpha)} \approx -\frac{N_4''(w_k)}{N_4'(w_k)}$ where $N_3(t) = N_3(t; x_k, x_{k-1}, w_{k-1}, y_{k-1})$ and $N_4(t) = N_4(t; w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1})$, we construct the following derivative-free with-memory method of Steffensen's type:

$$\begin{cases} \lambda_k = -\frac{1}{N_3'(x_k)}, \gamma_k = -\frac{N_4''(w_k)}{N_4'(w_k)}, k = 1, 2, 3, \cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k) + \gamma_k f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k)}, k = 0, 1, 2, \cdots. \end{cases} \tag{2.19}$$

Theorem 2.5. *If an initial approximation x_0 is sufficiently close to a simple zero α of $f(x)$, then the order of convergence of the Steffensen-like method with memory (2.19) is at least 7 [14].*

Now, we modify two new families of three-parametric with-memory methods. We look at the error equation (2.14). As can be seen that if we set $\lambda = \frac{-1}{f'(\alpha)}, \gamma = -c_2 = \frac{f''(\alpha)}{-2f'(\alpha)}$, and $\beta = \frac{f'''(\alpha)}{6}$, then at least the coefficient of e_k^4 disappears. However, we do not know α, and consequently, $f'(\alpha), f''(\alpha)$, and $f'''(\alpha)$ can not be computed. On the other hand, we can approximate α using available data and therefore improve the order of convergence. Hence, to this end, the following approximates are applied :

$$\begin{cases} \lambda_k = \frac{-1}{f'(\alpha)} \simeq -\frac{1}{N_3'(x_k)}, \\ \gamma_k = -\frac{f''(\alpha)}{2f'(\alpha)} \simeq -\frac{N_4''(w_k)}{2N_4'(w_k)}, \\ \beta_k = \frac{f'''(\alpha)}{6} \simeq \frac{N_5'''(y_k)}{6}, \end{cases} \tag{2.20}$$

where $k = 1, 2, \cdots$, the $N_3'(x_k), N_4'(w_k), N_4''(w_k)$ and $N_5'''(y_k)$ are Newton's interpolating polynomials of third till fifth degree, $N_3(t) = N_3(t; x_k, x_{k-1}, w_{k-1}, y_{k-1})$,
$N_4(t) = N_4(t; w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1})$ and $N_5(t) = N_5(t; y_k, w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1})$, set through best available approximations. Replacing the fixed parameters λ_k, γ_k, and β_k in the iterative formula (2.20) by the varying x_0, λ_0, γ_0, and β_0, we propose the following new methods with-memory, x_0, λ_0, γ_0 and β_0 are given, and $w_0 = x_0 + \lambda_0 f(x_0)$ (TM7.53):

$$\begin{cases} \lambda_k = -\frac{1}{N_3'(x_k)}, \gamma_k = -\frac{N_4''(w_k)}{N_4'(w_k)}, \beta_k = \frac{N_5'''(y_k)}{6}, k = 1, 2, 3, \cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), k = 0, 1, 2, \cdots, \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k) + \gamma_k f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k) + \beta_k (y_k - x_k)(y_k - w_k)}. \end{cases} \tag{2.21}$$

Theorem 2.6. *If an initial approximation x_0 is adequately close to a root α of $f(x) = 0$, then the convergence order of the family of two-parametric methods with-memory (2.21) is at least 7.53.*

Proof. : We will use Herzberger's matrix method [13] to determine the order of convergence. Note that the lower bound of order for a single-step s-point method (2.21) $x_k = \varphi(x_{k-1}, x_{k-2}, \cdots, x_{k-s})$ is the spectral radius of a matrix $M^{(s)} = (m_{ij})$, associated to the method with elements:

$$\begin{cases} m_{1,j} = amountofinformationrequiredatpointx_{k-j}, j = 1, 2, 3, \cdots, s, \\ m_{i,i-1} = 1, i = 2, 3, \cdots, s, \\ m_{i,j} = 0otherwise \end{cases} \tag{2.22}$$

On the other hand, the lower bound of the order of an s-step method $\varphi = \varphi_1 \circ \varphi_2 \circ \cdots \circ \varphi_s$ is the spectral radius of the product of matrices $M = M_1.M_2.\cdots.M_s$. We can express each approximation x_{k+1}, y_k, and w_k as a function of available information $f(y_k), f(w_k)$, and $f(x_k)$ from the k-th iteration and $f(y_{k-1}), f(w_{k-1})$, and $f(x_{k-1})$ from the previous iteration, depending on the accelerating technique. From the relations (2.21) and (2.22), we construct the corresponding matrices as follows:

$$x_{k+1} = \varphi_1(y_k, w_k, x_k, y_{k-1}, w_{k-1}, x_{k-1}); \Rightarrow M_1 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$y_k = \varphi_2(w_k, x_k, y_{k-1}, w_{k-1}, x_{k-1}, y_{k-2}); \Rightarrow M_2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$w_k = \varphi_3(x_k, y_{k-1}, w_{k-1}, x_{k-1}, y_{k-2}, w_{k-2}); \Rightarrow M_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Hence, we obtain:

$$M = M_1.M_2.M_3 = \begin{pmatrix} 4 & 4 & 4 & 4 & 0 & 0 \\ 2 & 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

its eigenvalues are $(\frac{1}{2}(7+\sqrt{65}), \frac{1}{2}(7-\sqrt{65}), 0, 0, 0, 0)$. Since the spectral radius of the matrix M is $\frac{1}{2}(7+\sqrt{65}) \approx 7.53113$, we conclude that the R-order of the methods with-memory (2.21) is at least 7.53113. The proof of Theorem 2.5 is finished. □

3. New Family Adaptive Methods

Now the question is whether it is possible to build a with-memory method with maximum convergence improvement. Is there a repetitive method whose efficiency index is two? We are going to do it a more efficient way, say recursive adaptively. Constructing a recursive adaptive method with-memory, we have used the information in all the previous iterations, i.e., from the beginning to the current iteration. Thus, as iterations proceed, the degree of interpolation polynomials increases, and the best-updated approximations for computing the self-accelerator λ_k, γ_k, and β_k are obtained. Indeed, we have developed the following recursive adaptive method with-memory. Let x_0, λ_0, γ_0, and β_0 be given suitably. Then:

$$\begin{cases} \lambda_k = -\frac{1}{N'_{3k}(x_k)}, \gamma_k = -\frac{N''_{3k+1}(w_k)}{2N'_{3k+1}(w_k)}, \beta_k = \frac{N'''_{3k+2}(y_k)}{6}, k = 1, 2, 3, \cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), k = 0, 1, 2, \cdots, \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k) + \gamma_k f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k) + \beta_k (y_k - x_k)(y_k - w_k)}. \end{cases} \tag{3.1}$$

In what follows, we discuss the general convergence analysis of the recursive adaptive method with-memory (3.1). It should be noted that the convergence order varies as the iteration go ahead. First, we need the following Lemma:

Lemma 3.1. *If* $\lambda_k = -\frac{1}{N'_{3k}(x_k)}, \gamma_k = -\frac{N''_{3k+1}(w_k)}{2N'_{3k+1}(w_k)}$, *and* $\beta_k = \frac{N'''_{3k+2}(y_k)}{6}$, *then:*

$$\begin{cases} (1 + \lambda_k f'(\alpha)) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \\ (c_2 + \gamma_k) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \\ (\beta_k + f'(\alpha) c_2 (\gamma + c_2) - f'(\alpha) c_3) \sim \prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y}, \end{cases} \tag{3.2}$$

where $e_s = x_s - \alpha, e_{s,w} = w_s - \alpha, e_{s,y} = y_s - \alpha$.

Proof. The proof is similar to *Lemma*3.1 in [33]. □

Theorem 3.2. *Let* x_0 *be a suitable initial guess to the simple root* α *of* $f(x) = 0$. *Also, suppose the initial values* λ_0, γ_0, *and* β_0 *are chosen appropriately. Then the R-order of the recursive adaptive method with memory (3.1) can be obtained from the following system of nonlinear equations:*

$$\begin{cases} r^k r_1 - (1 + r_1 + r_2)(1 + r + r^2 + r^3 + \cdots + r^{k-1}) - r^k = 0, \\ r^k r_2 - 2(1 + r_1 + r_2)(1 + r + r^2 + r^3 + \cdots + r^{k-1}) - 2r^k = 0, \\ r^{k+1} - 4(1 + r_1 + r_2)(1 + r + r^2 + r^3 + \cdots + r^{k-1}) - 4r^k = 0, \end{cases} \tag{3.3}$$

where r, r_1 *and* r_2 *are the convergence order of the sequences* $\{x_k\}, \{w_k\}$, *and* $\{y_k\}$, *respectively.*

Proof. Let $\{x_k\},\{w_k\}$, and $\{y_k\}$, be convergent with orders r, r_1, and r_2, respectively. Then:

$$\begin{cases} e_{k+1} \sim e_k^{r} \sim e_{k-1}^{r^2} \sim \ldots \sim e_0^{r^{k+1}}, \\ e_{k,w} \sim e_k^{r_1} \sim e_{k-1}^{r_1 r} \sim \ldots \sim e_0^{r_1 r^k}, \\ e_{k,y} \sim e_k^{r_2} \sim e_{k-1}^{r_2 r} \sim \ldots \sim e_0^{r_2 r^k}, \end{cases} \tag{3.4}$$

Now, by Lemma 3.1 and relation (3.4), we obtain:

$$\begin{aligned} (1+\lambda_k f'(\alpha)) \sim \textstyle\prod_{s=0}^{k-1} e_s e_{s,w} e_{s,y} &= (e_0 e_{0,w} e_{0,y}) \ldots (e_{k-1} e_{k-1,w} e_{k-1,y}) \\ &= (e_0 e_0^{r_1} e_0^{r_2})(e_0^{r} e_0^{r_1 r} e_0^{r_2 r}) \ldots (e_0^{r^{k-1}} e_0^{r^{k-1} r_1} e_0^{r^{k-1} r_2}) \\ &= e_0^{(1+r_1+r_2)+(1+r_1+r_2)r+\ldots+(1+r_1+r_2)r^{k-1}} \\ &= e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})}. \end{aligned}$$

Similarly, we get :

$$(\gamma_k + c_2) \sim e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})}, \tag{3.5}$$

and

$$(\beta_k + f'(\alpha)c_2(\gamma_k+c_2) - f'(\alpha)c_3) \sim e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})}. \tag{3.6}$$

By considering the errors of w_k, y_k, and x_{k+1} in relations (3.3), (3.5), and (3.6), we conclude:

$$e_{k,w} \sim (1+\lambda_k f'(\alpha)) e_k \sim e_0^{(1+r_1+r_2)(1+r+\ldots+r^{k-1})} e_0^{r^k}, \tag{3.7}$$

$$e_{k,y} \sim (1+\lambda_k f'(\alpha))(\gamma_k+c_2) e_k^2 \sim e_0^{((1+r_1+r_2)(1+r+\ldots+r^{k-1}))^2} e_0^{2r^k}, \tag{3.8}$$

$$e_{k+1} \sim (1+\lambda_k f'(\alpha))^2 (\gamma_k+c_2)(\beta_k + f'(\alpha)c_2(\gamma_k+c_2) - f'(\alpha)c_3) e_k^4 \sim e_0^{((1+r_1+r_2)(1+r+\ldots+r^{k-1}))^4} e_0^{4r^k}. \tag{3.9}$$

To obtain the desired result, it is enough to match the right-hand-side of the relations (3.4), (3.7), (3.8), and (3.9). Then:

$$\begin{cases} r^k r_1 - (1+r_1+r_2)(1+r+r^2+r^3+\cdots+r^{k-1}) - r^k = 0, k = 1,2,\cdots, \\ r^k r_2 - 2(1+r_1+r_2)(1+r+r^2+r^3+\cdots+r^{k-1}) - 2r^k = 0, \\ r^{k+1} - 4(1+r_1+r_2)(1+r+r^2+r^3+\cdots+r^{k-1}) - 4r^k = 0. \end{cases} \tag{3.10}$$

This completes the proof of the theorem. □

Remark 3.3. For $k=1$, we use the information from the current and the previous steps. In this case, the convergence order of the with-memory method can be computed from the following system:

$$\begin{cases} r r_1 - (1+r_1+r_2) - r = 0, \\ r r_2 - 2(1+r_1+r_2) - 2r = 0, \\ r^2 - 4(1+r_1+r_2) - 4r = 0. \end{cases} \tag{3.11}$$

It has been shown by TM7.53. This system of equations has the solution:
$r_1 = \frac{1}{8}(7+\sqrt{65}) \simeq 1.88278, r_2 = \frac{1}{4}(7+\sqrt{65}) \simeq 3.76556$ and $r = \frac{1}{2}(7+\sqrt{65}) \simeq 7.53113$.

Now, we propose iteration-schemes with-memory (TM7.94) as follows:

$$\begin{cases} \lambda_k = -\frac{1}{N_6'(x_k)}, \gamma_k = -\frac{N_7''(w_k)}{2N_7'(w_k)}, \beta_k = \frac{N_8'''(y_k)}{6}, k = 2,3,4,\cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), k = 0,1,2,\cdots, \\ p2(t_k) = \frac{(t_k-w_k)(t_k-y_k)}{(x_k-w_k)(x_k-y_k)} f(x_k) + \frac{(t_k-x_k)(t_k-y_k)}{(w_k-x_k)(w_k-y_k)} f(w_k) + \frac{(t_k-x_k)(t_k-w_k)}{(y_k-x_k)(y_k-w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k)+\gamma_k f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k)+\beta_k(y_k-x_k)(y_k-w_k)}. \end{cases} \tag{3.12}$$

Grade 9 Newton's interpolating polynomials are used to obtain a method with a higher convergence order as follows (TM7.99):

$$\begin{cases} \lambda_k = -\frac{1}{N_9'(x_k)}, \gamma_k = -\frac{N_{10}''(w_k)}{2N_{10}'(w_k)}, \beta_k = \frac{N_{11}'''(y_k)}{6}, k = 3,4,5,\cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), k = 0,1,2,\cdots, \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k) + \gamma_k f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k) + \beta_k (y_k - x_k)(y_k - w_k)}. \end{cases} \tag{3.13}$$

Also, we earn our proposed two-step with-memory method for $k = 4;5;6;...$, by(denoted (TM8)):

$$\begin{cases} \lambda_k = -\frac{1}{N_{12}'(x_k)}, \gamma_k = -\frac{N_{13}''(w_k)}{2N_{13}'(w_k)}, \beta_k = \frac{N_{14}'''(y_k)}{6}, k = 4,5,6,\cdots, \\ w_k = x_k + \lambda_k f(x_k), p1(t_k) = \frac{t_k - w_k}{x_k - w_k} f(x_k) + \frac{t_k - x_k}{w_k - x_k} f(w_k), k = 0,1,2,\cdots, \\ p2(t_k) = \frac{(t_k - w_k)(t_k - y_k)}{(x_k - w_k)(x_k - y_k)} f(x_k) + \frac{(t_k - x_k)(t_k - y_k)}{(w_k - x_k)(w_k - y_k)} f(w_k) + \frac{(t_k - x_k)(t_k - w_k)}{(y_k - x_k)(y_k - w_k)} f(y_k), \\ y_k = x_k - \frac{f(x_k)}{p1'(w_k) + \gamma_k f(w_k)}, x_{k+1} = y_k - \frac{f(y_k)}{p2'(y_k) + \beta_k (y_k - x_k)(y_k - w_k)}. \end{cases} \tag{3.14}$$

Where :

- $N_6(t) = N_6(t; x_k, x_{k-1}, w_{k-1}, y_{k-1}, x_{k-2}, w_{k-2}, y_{k-2})$, as an interpolation polynomial of sixth degree, passing through the best seven saved points $x_k, x_{k-1}, w_{k-1}, y_{k-1}, x_{k-2}, w_{k-2}, y_{k-2}$, for any $k \in \{2,3,4,\cdots\}$.
- $N_7(t) = N_7(t; w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}, x_{k-2}, w_{k-2}, y_{k-2})$, as an interpolation polynomial of seventh degree, passing through the best eight saved points $w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}, x_{k-2}, w_{k-2}, y_{k-2}$, for any $k \in \{2,3,4,\cdots\}$.
- $N_8(t) = N_8(t; y_k, w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}, x_{k-2}, w_{k-2}, y_{k-2})$, as an interpolation polynomial of eighth degree, passing through the best nine saved points $y_k, w_k, x_k, x_{k-1}, w_{k-1}, y_{k-1}, x_{k-2}, w_{k-2}, y_{k-2}$, for any $k \in \{2,3,4,\cdots\}$.

The convergence analysis of (3.12), (3.13), and (3.14) be established as follows:

Theorem 3.4. *Consider the same assumptions as in Theorem refthm2. Then, convergence R-order of the improved Steffensen's methods with-memory (3.12), (3.13), and (3.14) are 7.94449, 7.99315 and 7.99915, respectively.*

Proof. The convergence of each of the methods mentioned in equations (3.12), (3.13), and (3.14) is given separately.
Method TM7.94:

Let $\{x_k\}, \{w_k\}$, and $\{y_k\}$, be convergent with orders r, p, and q, respectively. Then:

$$e_{k+1} \sim e_k^r \sim e_{k-1}^{r^2} \sim e_{k-2}^{r^3} \sim e_{k-3}^{r^4} \sim e_{k-4}^{r^5}, \tag{3.15}$$

$$e_{k,w} \sim e_k^p \sim e_{k-1}^{pr} \sim e_{k-2}^{r^2 p} \sim e_{k-3}^{r^3 p} \sim e_{k-4}^{r^4 p}, \tag{3.16}$$

$$e_{k,y} \sim e_k^q \sim e_{k-1}^{qr} \sim e_{k-2}^{r^2 q} \sim e_{k-3}^{r^3 q} \sim e_{k-4}^{r^4 q}. \tag{3.17}$$

Using Theorem 2.1 and error equations (2.15), (2.16), and (2.17) we obtain:

$$e_{k,w} \sim (1 + \lambda_k f'(\alpha)) e_k, \tag{3.18}$$

$$e_{k,y} \sim (1 + \lambda_k f'(\alpha))(\gamma_k + c_2) e_k^2, \tag{3.19}$$

$$e_{k+1} \sim (1+\lambda_k f'(\alpha))^2(\gamma_k+c_2)(\beta_k+f'(\alpha)c_2(\gamma_k+c_2)-f'(\alpha)c_3)e_k^4. \tag{3.20}$$

And

$$(1+\lambda_k f'(\alpha)) \sim e_{k-2}e_{k-1}e_{k-2,w}e_{k-1,w}e_{k-2,y}e_{k-1,y}, \tag{3.21}$$

$$(\gamma_k+c_2) \sim e_{k-2}e_{k-1}e_{k-2,w}e_{k-1,w}e_{k-2,y}e_{k-1,y}, \tag{3.22}$$

$$(\beta_k+f'(\alpha)c_2(\gamma_k+c_2)-f'(\alpha)c_3) \sim e_{k-2}e_{k-1}e_{k-2,w}e_{k-1,w}e_{k-2,y}e_{k-1,y}, \tag{3.23}$$

Combining (3.16), (3.18), (3.21), and, (3.17), (3.19), (3.21), (3.22) also (3.15), (3.20), (3.21), (3.22), (3.23), we get

$$e_{k,w} \sim e_{k-2}^{1+r+p+rp+q+qr}, \tag{3.24}$$

and

$$e_{k,y} \sim e_{k-2}^{2(1+r+p+rp+q+qr)}, \tag{3.25}$$

also

$$e_{k+1} \sim e_{k-2}^{4(1+r+p+rp+q+qr)}, \tag{3.26}$$

Comparing the right and left side of error equations (3.15), (3.26), and (3.16), (3.24), and (3.17), (3.25), we have:

$$\begin{cases} r^2p=(1+r+r^2+p+pr+q+qr), \\ r^2q=2(1+r+r^2+p+pr+q+qr), \\ r^3=4(1+r+r^2+p+pr+q+qr). \end{cases} \tag{3.27}$$

The positive real solution of this system is $p_1 \simeq 1.98612, p_2 \simeq 3.97225$ and $r \simeq 7.94449$. We conclude that the R-order of the with-memory methods (3.12) is at least 7.94449.

Method TM7.99:
Similar to the previous method:

$$(1+\lambda_k f'(\alpha)) \sim e_{k-3}e_{k-2}e_{k-1}e_{k-3,w}e_{k-2,w}e_{k-1,w}e_{k-3,y}e_{k-2,y}e_{k-1,y}, \tag{3.28}$$

$$(\gamma_k+c_2) \sim e_{k-3}e_{k-2}e_{k-1}e_{k-3,w}e_{k-2,w}e_{k-1,w}e_{k-3,y}e_{k-2,y}e_{k-1,y}, \tag{3.29}$$

$$(\beta_k+f'(\alpha)c_2(\gamma_k+c_2)-f'(\alpha)c_3) \sim e_{k-3}e_{k-2}e_{k-1}e_{k-3,w}e_{k-2,w}e_{k-1,w}e_{k-3,y}e_{k-2,y}e_{k-1,y}, \tag{3.30}$$

Combining (3.16), (3.18), (3.28), and, (3.17), (3.19), (3.28), (3.29) also (3.15), (3.20), (3.28), (3.29), (3.30) we get

$$e_{k,w} \sim e_{k-3}^{1+r+r^2+p+rp+r^2p+q+qr+r^2q}, \tag{3.31}$$

and

$$e_{k,y} \sim e_{k-3}^{1+r+r^2+p+rp+r^2p+q+qr+r^2q} \tag{3.32}$$

also

$$e_{k+1} \sim e_{k-3}^{1+r+r^2+p+rp+r^2p+q+qr+r^2q} \tag{3.33}$$

Comparing the right and left side of error equations (3.16), (3.31), and (3.17), (3.32), also (3.15), (3.33), we obtained this system equation:

$$\begin{cases} r^3p = (1+r+r^2+p+pr+r^2p+q+qr+r^2q), \\ r^3q = 2(1+r+r^2+p+pr+r^2p+q+qr+r^2q), \\ r^4 = 4(1+r+r^2+p+pr+r^2p+q+qr+r^2q). \end{cases} \tag{3.34}$$

The positive solution of this system is $p \simeq 1.99829, q \simeq 3.99657$, and $r \simeq 7.99315$. Therefore, the R-order of the with-memory methods (3.13) is at least 7.99315.

Method TM8

Using the result of the two methods TM7.94, TM7.99 and Lemma 3.1, we have

$$\begin{cases} (1+\lambda_k f'(\alpha)) \sim \prod_{s=0}^{k-4} e_s e_{s,w} e_{s,y}, \\ (\gamma_k + c_2) \sim \prod_{s=0}^{k-4} e_s e_{s,w} e_{s,y}, \\ (\beta_k + f'(\alpha)c_2(\gamma_k+c_2) - f'(\alpha)c_3) \sim \prod_{s=0}^{k-4} e_s e_{s,w} e_{s,y}, \end{cases} \tag{3.35}$$

Compare the right and left side of error equations (3.8), (3.29), and (3.9), (3.29), and (3.15), (3.35), we have:

$$e_{k,w} \sim e_{k-4}^{1+r+r^2+r^3+p+pr+r^2p+r^3p+q+qr+r^2q+r^3q}, \tag{3.36}$$

and

$$e_{k,y} \sim e_{k-4}^{1+r+r^2+r^3+p+pr+r^2p+r^3p+q+qr+r^2q+r^3q} \tag{3.37}$$

also

$$e_{k+1} \sim e_{k-4}^{1+r+r^2+r^3+p+pr+r^2p+r^3p+q+qr+r^2q+r^3q} \tag{3.38}$$

Comparing the right and left side of error equations (3.15), (3.38), and (3.16), (3.36), also (3.17), (3.37), we have. Similarly, we find the final system equation, which is given by

$$\begin{cases} r^4p = (1+r+r^2+r^3+p+pr+r^2p+r^3p+q+qr+r^2q+r^3q), \\ r^4q = 2(1+r+r^2+r^3+p+pr+r^2p+r^3p+q+qr+r^2q+r^3q), \\ r^5 = 4(1+r+r^2+r^3+p+pr+r^2p+r^3p+q+qr+r^2q+r^3q). \end{cases} \tag{3.39}$$

Since positive solution of this system is $p = 1.99979, q = 3.99957$ and, $r = 7.99915 \approx 8$, and therefore, we conclude that the R-order of the with-memory methods (3.14) is at least $7.99915 \approx 8$.
Therfore, the proof of Theorem 3.4 is completed. □

4. Numerical results and comparisons

One can find the errors $|x_k - \alpha|$ of the root approximations in Tables 1 and 2, where $M(-h)$ stands for $M \times 10^{-h}$. Tables $1-2$ contain each test function, the initial estimation values, and the last value of the computational order of convergence COC[25] computed by the expressions

$$COC = \frac{\log|f(x_n)/f(x_{n-1})|}{\log|f(x_{n-1})/f(x_{n-2})|} \tag{4.1}$$

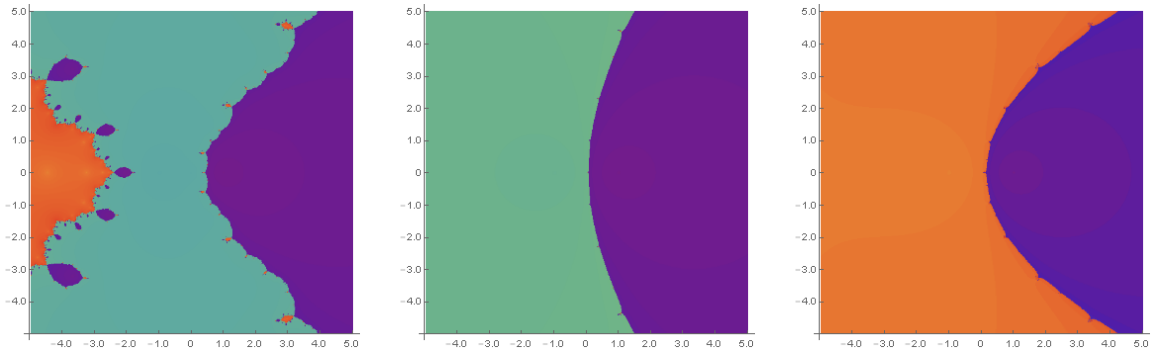
Traub's idea has been used to compare different methods. Traub [31] proposed the concept of efficiency index as a measure for comparing different methods. From $p^{1/n}$, wherein p is the order of convergence and n is the whole number of evaluations per iteration. The package Mathematica 10, with 5000 arbitrary precision arithmetic, has been used in our computations. We employ Campos et al.'s method (CCTVM)[4], Cordero et al.'s method (CLBTM), (CLKTM)[5, 6], Chun and Lee method (CLM)[7], Dzunic's method (DM)[11], Traub's method (1.1)(TM), Kansal et al.'s method (KKBM) [15], Mohamadi et al.'s method (MLAM) [19], Maheshwari's method (MM) [21], Petkovic et al.'s method (PDNM) [23], (PIDM) [24], Neta's method (1.3), Lotfi et al.'s method (2.18), Jaiswal's method (JM) (2.19), Soleymani's method (SM)[28], Torkashvand and Kazemi(TKM)[33], Wang's method (WM) [35],TM7(2.21) and TM8(3.1) to solve nonlinear equations given in Examples 1-3. Tables 1 and 2 show that the efficiency index as well as the convergence improvement of the new adaptive methods is higher than all existing methods. In other words, TM7.53 and TM8 have efficiency indices $7.5319^{\frac{1}{3}} \simeq 1.9602$, and $8^{\frac{1}{3}} = 2$. In all numerical examples performed, we

have assumed the initial value of λ_0, γ_0, and β_0 to be 0.1. In these tables symbol, EI is the abbreviation from the efficiency Index. In Table 3, TNE stand for the total number of function evaluations and compares the convergence order improvement of the with-memory methods. We present some numerical test results with the following functions:

$$\begin{cases} f_1(x) = e^{-x+x^3} - \cos(x^2-1) + x^3 + 1, \alpha = -1, x_0 = -1.4, \\ f_2(x) = \log(1+x^2) + e^{x^2-3x}\sin(x), \alpha = 0, x_0 = 0.6, \\ f_3(x) = e^{-5x}(x-2)(x^{10}+x+2), \alpha = 2, x_0 = 2.2. \\ f_4(x) = \frac{8}{17} - \sqrt{6} + \frac{x^3}{1+x^4} + \sqrt{8+x^4}\sin(\frac{\pi}{2+x^2}), \alpha = -2, x_0 = -2.3. \\ f_5(x) = x\log 1 + x\sin x + e^{-1+x^2+x\cos x}\sin(\pi x), \alpha = 0, x_0 = 0.6. \end{cases} \tag{4.2}$$

5. Dynamical Analysis

Dynamical properties of the iterative method give us pivotal information about their numerical features as their stability and reliability. Some significant results concerning the dynamic performances have been obtained in [9, 10, 14, 17, 18, 20, 34]. In what follows, we have compared iterative methods (2.1), (2.18), and (2.21) by using the basins of attraction for three complex polynomials $p_1(z) = z^2 - 1, p_2(z) = z^3 - 1, p_3(z) = z^4 - 1$. We have used a similar method as in [9, 18, 20, 27] to generate the basins of attraction. The basins of attraction for the zeros of a polynomial and an iterative method, we take a grid of 500×500 points in a rectangle $D = [-5, 5] \times [-5, 5] \subset C$, and we use these points as z_0. Whenever the sequence generated by the iterative method achieves a zero z^* of polynomial $p_i(x)$, then we take with a tolerance $|z - z^*| < 10^{-10}$, and a maximum of 25 iterations. Therefore, we determine that z_0 is in the basin of attraction of the zero and we paint this point in a color previously selected for this root. Figures (1), (2), (3), (4), (5), (6), (7), and (8) show that the basins of attraction with-memory method are higher than the without-memory-method. Figures (4), (5), (6), (7), (8), and (9) show that the accelerator parameters play a decisive role in increasing the absorption domain of a repetitive-method. Also, the smaller the size of the self-accelerator parameters, the greater the stability savings.



(a):T4 (2.1), $\gamma = 1$ (b):LSNKKM, $\gamma = 0.1$ (c):TM7.53 , $\gamma = \lambda = \beta = 0.1$

Figure 1. Coparision methods for finding the roots of the polynomial $p_1(z)$

It can be seen from Figures (10), (11), and (12) that the memory methods proposed TM7.53 (2.21) have the basin of the attraction and stability region greater than Kung-Traub and Ostrwoski's method. Besides,the method TM7.53 does not require the calculation of a function derivative.

6. Concluding Remarks

In this work, we proposed the Steffensen-Type adaptive methods with-memory for solving nonlinear equations. In Tables $1 - 2$, we have examined some methods with different kinds of convergence order. The last column of tables shows the efficiency index. Proposed methods (2.21), (3.12), (3.13), and (3.14) do not need any derivatives and can

TABLE 1. Comparison table for methods with memory on $f_i(x), i = 1, 2, 3, 4$

$f_1(x) = e^{-x+x^3} - \cos(x^2-1) + x^3 + 1, \alpha = -1, x_0 = -1.4$					
Methods	$\lvert x_1 - \alpha\rvert$	$\lvert x_2 - \alpha\rvert$	$\lvert x_3 - \alpha\rvert$	COC	EI
CLM [7]	0.29422(−5)	0.13127(−44)	0.20618(−359)	8.0000	1.6818
MM [21]	0.21481(−2)	0.13684(−10)	0.22069(−43)	4.0000	1.5874
LSNKKM(2.18)	0.17352(−2)	0.12400(−15)	0.73823(−97)	6.0000	1.8171
TM(1.1)	0.24592(−1)	0.92160(−5)	0.35695(−12)	2.4661	1.5703
JM(2.19)	0.41584(−2)	0.72557(−15)	0.69716(−107)	7.0000	1.9129
TM7(2.21)	0.44331(−2)	0.99689(−17)	0.25924(−130)	7.5334	1.9603
TM7.94(3.12)	0.44331(−3)	0.99039(−19)	0.32428(−150)	7.9445	1.9954
TM8(3.14)	0.51732(−4)	0.41862(−24)	0.27043(−196)	8.0000	2.0000
$f_2(x) = \log(1+x^2) + e^{x^2-3x}\sin(x), \alpha = 0, x_0 = 0.6$					
CLM [7]	0.11441(−2)	0.93950(−21)	0.19171(−165)	8.0000	1.6818
MM [21]	0.27871(−1)	0.13198(−4)	0.64728(−18)	4.0000	1.5874
LSNKKM(2.18)	0.37954(−1)	0.63632(−8)	0.43596(−47)	6.0000	1.8171
TM(1.1)	0.80232(−1)	0.26521(−2)	0.20874(−5)	2.4682	1.5710
JM(2.19)	0.45799(−1)	0.18860(−7)	0.58696(−53)	7.0000	1.9129
TM7(2.21)	0.39198(−1)	0.51000(−9)	0.85761(−68)	7.5319	1.9602
TM7.94(3.12)	0.65831(−2)	0.57039(−14)	0.98728(−115)	7.9445	1.9954
TM8(3.14)	0.73578(−2)	0.25340(−15)	0.11402(−122)	8.0000	2.0000
$f_3(x) = e^{-5x}(x-2)(x^{10}+x+2), \alpha = 2, x_0 = 2.2$					
CLM [7]	0.80210(−7)	0.13529(−58)	0.88646(−473)	8.0000	1.6818
MM [21]	0.15871(−2)	0.11197(−12)	0.34946(−53)	4.0000	1.5874
LSNKKM(2.18)	0.95635(−3)	0.31158(−20)	0.14698(−125)	6.0000	1.8171
TM(1.1)	0.20469(−1)	0.97531(−7)	0.22483(−19)	2.3839	1.5440
JM(2.19)	0.70416(−3)	0.36818(−22)	0.14255(−157)	7.0000	1.9129
TM7(2.21)	0.20122(−2)	0.36808(−23)	0.14354(−176)	7.5307	1.9601
TM7.94(3.12)	0.11257(−3)	0.45879(−13)	0.13214(−120)	7.9445	1.9954
TM8(3.14)	0.17122(−1)	0.19683(−15)	0.17525(−128)	8.0000	2.0000
$f_4(x) = \frac{8}{17} - \sqrt{6} + \frac{x^3}{1+x^4} + \sqrt{8+x^4}\sin(\frac{\pi}{2+x^2}), \alpha = -2, x_0 = -2.3.$					
CLM [7]	0.32748(−6)	0.19764(−52)	0.34783(−422)	8.0000	1.6818
MM [21]	0.41214(−3)	0.14311(−13)	0.20874(−55)	4.0000	1.5874
LSNKKM(2.18)	0.36857(−3)	0.78764(−21)	0.49509(−127)	6.0000	1.8171
TM(1.1)	0.10201(−1)	0.28305(−5)	0.13002(−13)	2.4256	1.5574
JM(2.19)	0.66688(−4)	0.13751(−29)	0.21659(−209)	7.0000	1.9129
TM7(2.21)	0.20401(−4)	0.53307(−38)	0.17813(−288)	7.5310	1.9601
TM7.94(3.12)	0.45831(−2)	0.24578(−24)	0.21545(−220)	7.9445	1.9954
TM8(3.14)	0.57116(−4)	0.34511(−30)	0.10693(−244)	8.0000	2.0000

be used even for non-smooth functions. The efficiency index of the proposed adaptive family with-memory is $8^{\frac{1}{3}} = 2$, which is much better than optimal methods without-memory. Besides, it is higher than all the methods mention in the references [1–3, 8, 26, 27, 29, 30]. We observed a 100% improvement in the order convergence of adaptive methods theoretically with given numerical examples. Tables 1, 2 and 3 show that all of the methods work in concordance with theoretical results.

TABLE 2. Comparison table for methods with memory on $f_5(x)$

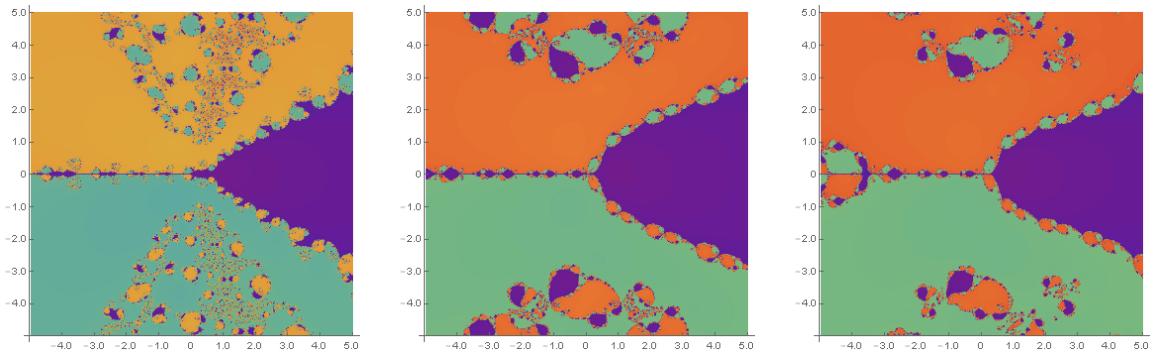
$f_5(x) = x\log 1 + x\sin x + e^{-1+x^2+x\cos x}\sin(\pi x), \alpha = 0, x_0 = 0.6.$					
Methods	$\lvert x_1 - \alpha\rvert$	$\lvert x_2 - \alpha\rvert$	$\lvert x_3 - \alpha\rvert$	COC	EI
CLM [7]	0.56319(−2)	0.57874(−17)	0.70857(−137)	8.0000	1.6818
MM [21]	0.74205(−1)	0.11104(−3)	0.49843(−15)	4.0000	1.5874
LSNKKM(2.18)	0.23476(0)	0.82605(−5)	0.27931(−29)	6.0000	1.8171
TM(1.1)	0.47811(0)	0.56230(−1)	0.12602(−2)	2.4825	1.5756
JM(2.19)	0.18448(0)	0.50259(−5)	0.92967(−37)	7.0000	1.9129
TM7(2.21)	0.19440(0)	0.23473(−5)	0.36314(−44)	7.5500	1.9618
TM7.94(3.12)	0.19440(0)	0.32541(−3)	0.25488(−42)	7.9445	1.9954
TM8(3.14)	0.19440(0)	0.36233(−5)	0.30096(−43)	8.0000	2.0000

TABLE 3. Comparison of the Percentage Improvement of Cconvergence Order Methods

With-Memory Methods	Optimal Order	TNE	p	Percentage Increase
CCTVM[4]	4.0000	3	4.2361	%5.90
CLBTM [5]	4.0000	3	6.0000	%50
CLKTM [6]	4.0000	3	6.0000	%50
DM[11]	4.0000	3	7.0000	%75
JM(2.19)	4.0000	3	7.0000	%75
JM [14]	8.0000	4	14.000	%75
LSNKKM (2.18)	4.0000	3	6.0000	%50
KKBM[15]	4.0000	3	7.0000	%75
MLAM[19]	4.0000	3	5.9500	%48.75
NM[22]	8.0000	4	10.8153	%35.19
PDNM[23]	4.0000	3	4.5616	%14.04
PIDM[24]	4.0000	3	4.4494	%11.24
SM[28]	8.0000	4	9.5826	%19.78
SM[28]	8.0000	4	10.0000	%20
TM (1.1)	2.0000	2	2.4100	%20.5
TKM[33]	2.0000	2	3.5616	%78.08
TKM[33]	8.0000	3	14.0000	%75
TKM[33]	16.0000	4	28.0000	%75
TKM[33]	4.0000	3	7.0000	%75
WM[35]	4.0000	3	4.4494	%11.235
TM7(2.21)	4.0000	3	7.5313	%88.34
TM7.94(3.12)	4.0000	3	7.9445	%98.61
TM7.99(3.13)	4.0000	3	7.9932	%99.83
TM8(3.14)	4.0000	3	8.0000	%100

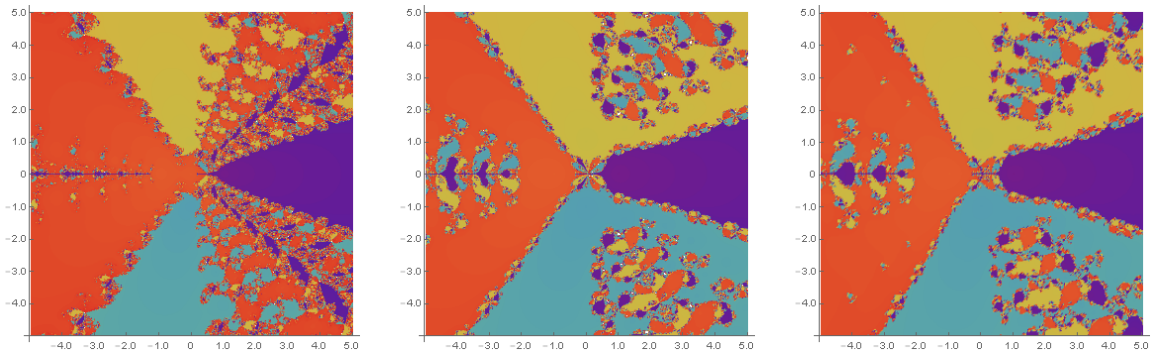
ACKNOWLEDGMENT

The author is thankful to two anonymous referees for careful reading and noteworthy comments which improved the quality of this manuscript.



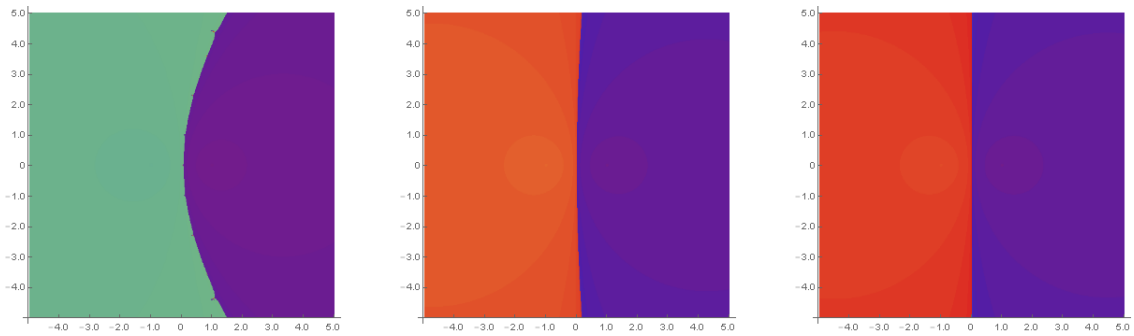
(a):T4 (2.1), $\gamma = 1$ (b):LSNKKM, $\gamma = 0.1$ (c):TM7.53 , $\gamma = \lambda = \beta = 0.1$

FIGURE 2. Coparision methods for finding the roots of the polynomial $p_2(z)$



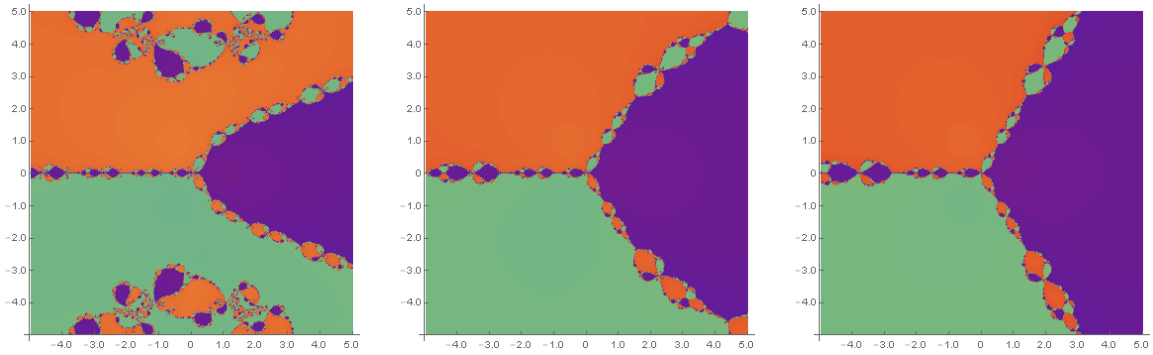
(a):T4 (2.1), $\gamma = 1$ (b):LSNKKM, $\gamma = 0.1$ (c):TM7.53 , $\gamma = \lambda = \beta = 0.1$

FIGURE 3. Coparision methods for finding the roots of the polynomial $p_3(z)$



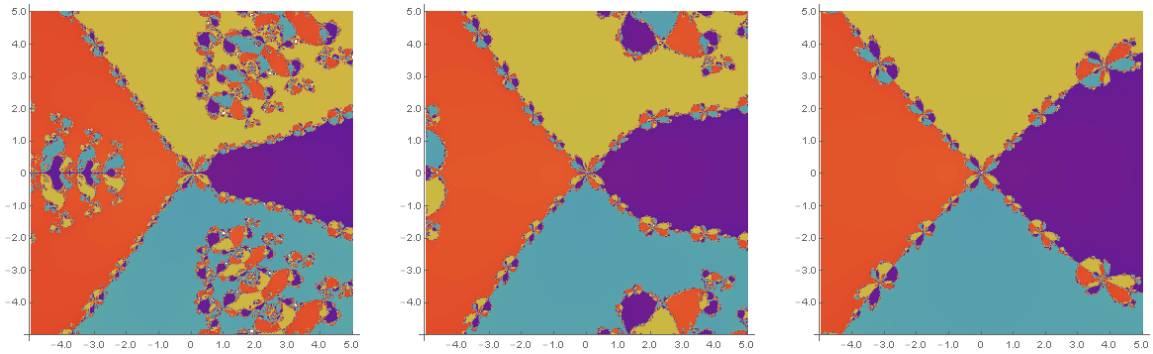
(a):LSNKKM, $\gamma = 0.1$ (b):LSNKKM, $\gamma = 0.01$ (c):LSNKKM, $\gamma = 0.001$

FIGURE 4. LSNKKM (2.18) Finding the roots of the polynomial $p_1(z)$



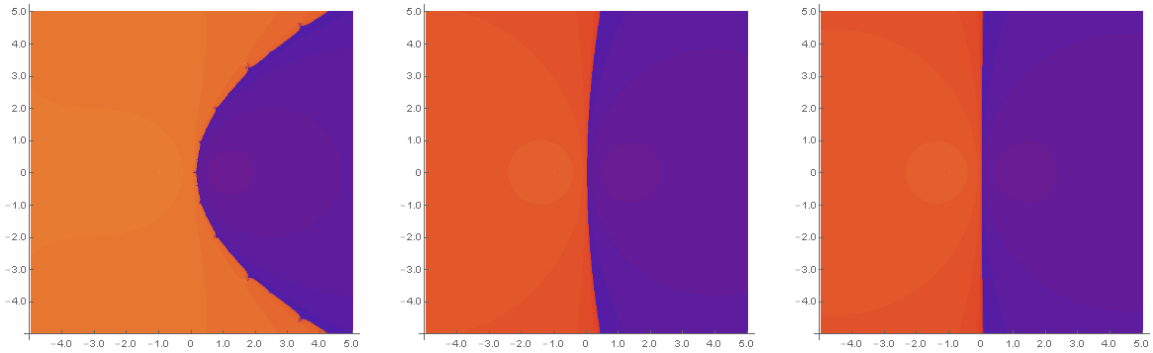
(a):LSNKKM, $\gamma = 0.1$ (b):LSNKKM, $\gamma = 0.01$ (c):LSNKKM, $\gamma = 0.001$

FIGURE 5. LSNKKM (2.18) Finding the roots of the polynomial $p_2(z)$



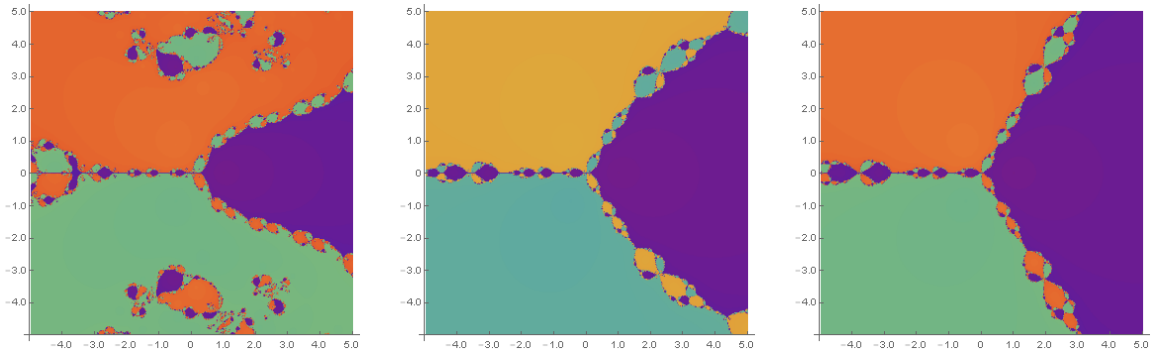
(a):LSNKKM, $\gamma = 0.1$ (b):LSNKKM, $\gamma = 0.01$ (c):LSNKKM, $\gamma = 0.001$

FIGURE 6. LSNKKM (2.18) Finding the roots of the polynomial $p_3(z)$



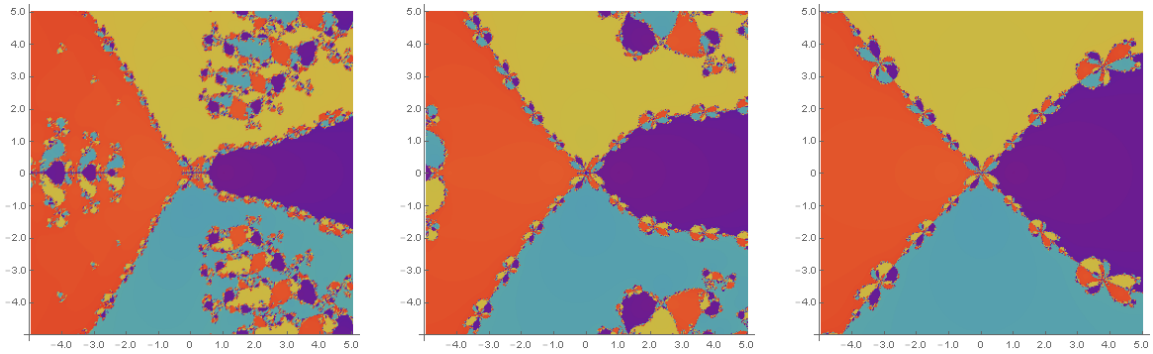
(a):TM7.53 , $\gamma = \lambda = \beta = 0.1$ (b):TM7.53 , $\gamma = \lambda = \beta = 0.01$ (c):TM7.53, $\gamma = \lambda = \beta = 0.001$

FIGURE 7. TM7.53 (2.21) for finding the roots of the polynomial $p_1(z)$



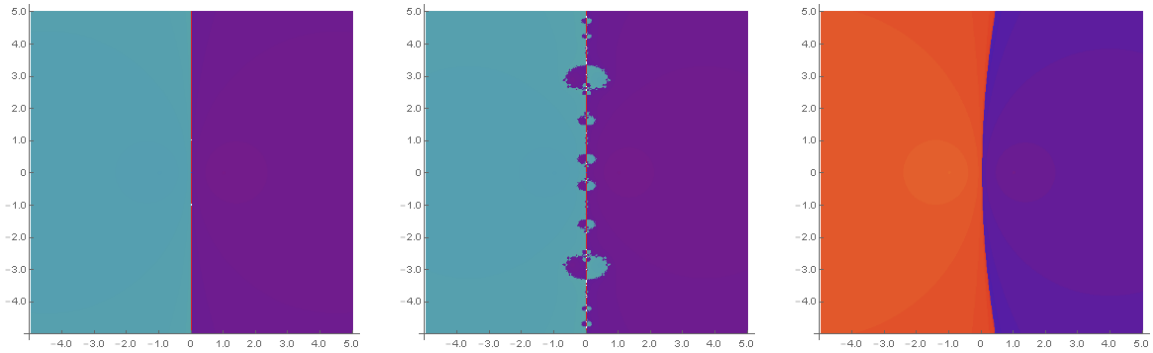
(a):TM7.53 , $\gamma = \lambda = \beta = 0.1$ (b):TM7.53 , $\gamma = \lambda = \beta = 0.01$ (c):TM7.53, $\gamma = \lambda = \beta = 0.001$

FIGURE 8. TM7.53 (2.21) for finding the roots of the polynomial $p_2(z)$



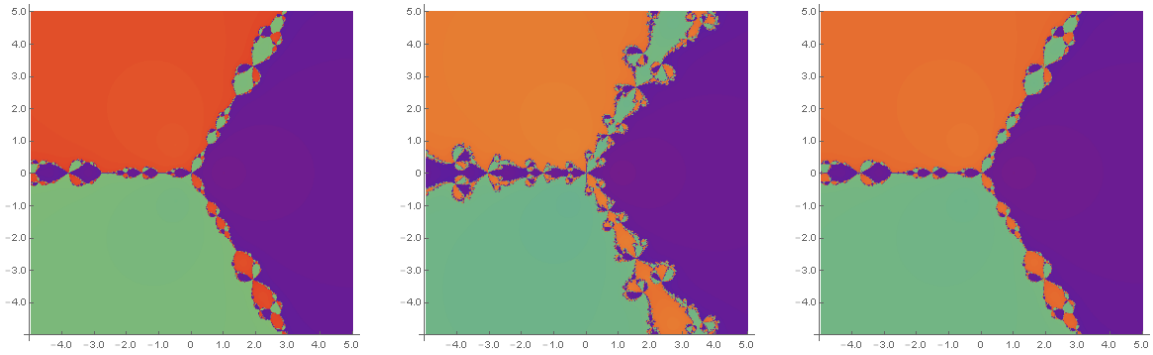
(a):TM7.53 , $\gamma = \lambda = \beta = 0.1$ (b):TM7.53 , $\gamma = \lambda = \beta = 0.01$ (c):TM7.53, $\gamma = \lambda = \beta = 0.001$

FIGURE 9. TM7.53 (2.21) for finding the roots of the polynomial $p_3(z)$



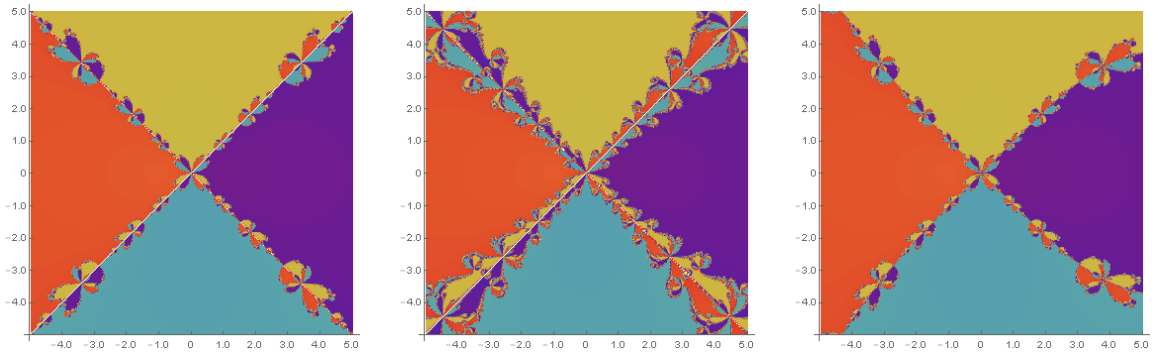
(a):Ostrowski's Method (b):Kung-Traub's Method (c):TM7.53 Method (2.21)

FIGURE 10. Coparision methods for finding the roots of the polynomial $p_1(z)$



(a):Ostrowski's Method (b):Kung-Traub's Method (c):TM7.53 Method (2.21)

FIGURE 11. Coparision methods for finding the roots of the polynomial $p_2(z)$



(a):Ostrowski's Method (b):Kung-Traub's Method (c):TM7.53 Method (2.21)

FIGURE 12. Coparision methods for finding the roots of the polynomial $p_3(z)$

References

[1] I. K. Argyros and L. U. Uko, *An improved convergence analysis of a one-step intermediate Newton iterative scheme for nonlinear equations*, Journal of Applied Mathematics and Computing., *38* (2012), 243–256.

[2] I. K. Argyros and S. George, *Extended Kung-Traub-Type method for solving equations*, TWMS journal of pure and applied mathematics, *12*(2) (2021), 193-198.

[3] R. Behl, A. Cordero, S. S. Motsa, and J. R. Torregrosa, *Multiplicity anomalies of an optimal fourth-order class of iterative methods for solving nonlinear equations*, Nonlinear Dynamics., *91*(1) (2018), 81–112.

[4] B. Campos, A. Cordero, J. R. Torregrosa, and P. Vindel, *Stability of King's family of iterative methods with memory*, Journal of Computational and Applied Mathematics., *318* (2017), 504–514.

[5] A. Cordero, T. Lotfi, P. Bakhtiari, and J. R. Torregrosa, *An efficient two-parametric family with memory for nonlinear equations*, Numerical Algorithm., *68*(2) (2014), 323–335.

[6] A. Cordero, T. Lotfi, A. Khoshandi, and J. R. Torregrosa, *An efficient Steffensen-like iterative method with memry*, Bull. Math. Soc. Sci. Math. Roum, Tome., *58*(1) (2015), 49–58.

[7] C. Chun and M. Y. Lee, *A new optimal eighth-order family of iterative methods for the solution of nonlinear equations*, Applied Mathematics and Computation., *223* (2013), 506–519.

[8] C. Chun and B. Neta, *An analysis of a new family of eighth-order optimal methods*, Applied Mathematics and Computation, *245* (2014), 86–107.

[9] C. Chun, M. Y. Lee, B. Neta, and J. Dzunic, *On optimal fourth-order iterative methods free from second derivative and their dynamics*, Applied Mathematics and Computation., *218*(11) (2012), 6427–6438.

[10] P. A. Delshad and T. Lotfi, *On the local convergence of Kung-Traub's two-point method and its dynamics*, Applications of Mathematics., *65*(4) (2020), 379–406.

[11] J. Dzunic, *On efficient two-parameter methods for solving nonlinear equations*, Numerical Algorithm., *63* (2013), 549-569.

[12] M. A. Fariborzi Araghi, T. Lotfi, and V. Torkashvand, *A general efficient family of adaptive method with memory for solving nonlinear equations*, Bull. Math. Soc. Sci. Math. Roumanie Tome., *62*(1) (2019), 37–49.

[13] J. Herzberger, *Uber Matrixdarstellungen fur Iterationverfahren bei nichtlinearen Gleichungen*, Computing., *12* (1974), 215–222.

[14] J. P. Jaiswal, *Two Bi-Accelerator Improved with Memory Schemes for Solving Nonlinear Equations*, Discrete Dynamics in Nature and Society., *2015* (2015), 1–7.

[15] M. Kansal, V. Kanwar, and S. Bhatia, *Efficient derivative-free variants of Hansen-Patrick's family with memory for solving nonlinear equations*, Numerical Algorithms., *73*(4) (2016), 1017–1036.

[16] T. Lotfi and P. Assari, *Two new three and four parametric with memory methods for solving nonlinear equations*, International Journal Industrial Mathematics., *7*(3) (2015), 269–276.

[17] T. Lotfi, F. Soleymani, Z. Noori, A. Kilicman, and F. Khaksar Haghani, *Efficient iterative methods with and without memory possessing high efficiency indices*, Discrete Dynamics in Nature and Society., *2014* (2014), 1–9.

[18] T. Lotfi, S. Sharifi, M. Salimi, and S. Siegmund, *A new class of three-point methods with optimal convergence order eight and its dynamics*, Numerical Algorithms., *68*(2) (2015), 261–288.

[19] M. Mohamadi Zadeh, T. Lotfi, and M. Amirfakhrian, *Developing two efficient adaptive Newton-type methods with memory*, Mathematical Methods in the Applied Sciences., *42*(17) (2019), 5687–5695.

[20] M. Moccari and T. Lotfi, *On a two-step optimal Steffensen-type method: Relaxed local and semi-local convergence analysis and dynamical stability*, Journal of Mathematical Analysis and Applications., *468*(1) (2018), 240–269.

[21] A. M Maheshwari, *A fourth order iterative method for solving nonlinear equations*, Applied Mathematics and Computation., *211* (2009), 383–391.

[22] B. Neta, *A new family of higher order methods for solving equations*, International Journal of Computer Mathematics., *14* (1983), 191–195.

[23] M. S. Petkovic, J. Dzunic, and B. Neta, *Interpolatory multipoint methods with memory for solving nonlinear equations*, Applied Mathematics and Computation., *218* (2011), 2533–2541.

[24] M. S. Petkovic, S. Ilic, and J. Dzunic, *Derivative free two-point methods with and without memory for solving nonlinear equations*, Applied Mathematics and Computation., *217* (2010), 1887–1895.

[25] M. S. Petkovic, B. Neta, L. D. Petkovic, and J. Dzunic, *Multipoint methods for solving nonlinear equations*, Elsevier, Amsterdam, 2013.
[26] J. Raj Sharma and R. Sharma, *A new family of modified Ostrowski's methods with accelerated eighth order convergence*, Numerical Algorithms., *54* (2010), 445–458.
[27] M. Salimi, T. Lotfi, S. Sharifi, and S. Siegmund, *Optimal Newton-Secant like methods without memory for solving nonlinear equations with its dynamics*, International Journal of Computer Mathematics., *94*(9) (2017), 1759–1777.
[28] F. Soleymani, *Some optimal iterative methods and their with memory variants*, Journal of the Egyptian Mathematical Society., *21*(2) (2013), 133–141.
[29] F. Soleymani, *Some High-Order Iterative Methods for Finding All the Real Zeros*, Thai Journal of Mathematics., *12*(2) (2014), 313–327.
[30] F. Soleymani, *New class of eighth-order iterative zero-finders and their basins of attraction*, Afrika Matematika., *25*(1) (2014), 67–79.
[31] J. F. Traub, *Iterative Methods for the Solution of Equations*, Prentice Hall, New York, USA, 1964.
[32] V. Torkashvand, *Structure of an Adaptive with Memory Method with Efficiency Index 2*, International Journal of Mathematical Modelling Computations., *9*(4) (2019), 239–252.
[33] V. Torkashvand and M. Kazemi, *On an Efficient Family with Memory with High Order of Convergence for Solving Nonlinear Equations*, International Journal Industrial Mathematics., *12*(2) (2020), 209–224.
[34] H. Veiseh, T. Lotfi, and T. Allahviranloo, *A study on the local convergence and dynamics of the two-step and derivative-free Kung-Traub's method*, Computational and Applied Mathematics., *37*(3) (2018), 2428–2444.
[35] X. Wang, *A new accelerating technique applied to a variant of Cordero-Torregrosa method*, Journal of Computational and Applied Mathematics., *330* (2018), 695–709.
[36] F. Zafar, A. Cordero, J. R. Torregrosa, and A. Rafi, *A class of four parametric with- and without-memory root finding methods*, Computational and Mathematical Methods., *1*(3) (2019), 1–13.