Accepted Manuscript

An Ostrowski-type method with memory using a novel self-accelerating parameter

Xiaofeng Wang

PII: S0377-0427(17)30191-7
DOI: http://dx.doi.org/10.1016/j.cam.2017.04.021
Reference: CAM 11100

To appear in: *Journal of Computational and Applied Mathematics*

Received date: 19 October 2016
Revised date: 20 February 2017

Please cite this article as: X. Wang, An Ostrowski-type method with memory using a novel self-accelerating parameter, *Journal of Computational and Applied Mathematics* (2017), http://dx.doi.org/10.1016/j.cam.2017.04.021

An Ostrowski-type method with memory using a novel self-accelerating parameter

Xiaofeng Wang [a,*]

[a] School of Mathematics and Physics, Bohai University, Jinzhou 121000 Liaoning, China

Abstract

In this paper, an Ostrowski-type method with memory is proposed for solving nonlinear equations. To this end, we first present an optimal fourth-order Ostrowski-type method without memory. Based on this method without memory, an Ostrowski-type method with memory is given by using a simple self-accelerating parameter. The new self-accelerating parameter is constructed by a novel way and has the properties of simple structure and easy calculation, which do not increase the computational cost of the iterative method. The convergence order of the new iterative method is increased from 4 to $2+\sqrt{5}\approx 4.2361$, $\left(5+\sqrt{13}\right)/2\approx 4.30278$ and $2+\sqrt{6}\approx 4.4495$, respectively. Numerical experiments are made to show the performance of the new method, which support the theoretical results. From the comparison with some known methods, it is observed that the new method occupies less computing time.

MSC: 65H05 65B99

Keywords: Ostrowski's method; Iterative method with memory; Self-accelerating parameter; Root-finding;

1. Introduction

In this paper, we consider iterative method to find a simple root of a nonlinear equation $f(x)=0$, where $f: I\subset R\to R$ for an open interval I is a scalar function. There are different methods for solving the root a of a nonlinear equation $f(x)=0$, the most well known of these methods is the classical Newton's method [1]: $x_{n+1}=x_n-f(x_n)/f'(x_n)$. If the sequence $\{x_n\}_0^\infty$ generated by Newton's method converges to a simple root a of nonlinear equation, then the sequence satisfies the following expression

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

where $c_2=f''(a)/(2f'(a))$ is the asymptotic error constant, $e_{n+1}=x_{n+1}-a$ and $e_n=x_n-a$. Equation (1) means that Newton's method converges quadratically.

To improve the convergence order and computational efficiency of the Newton's method, many

efficient multipoint iterative methods have been proposed in recent years, see [2-26] and references therein. Among them, the multipoint method with memory is a special kind of multipoint iterative method, which can further improve the computational efficiency of the multipoint method without memory without any additional functional evaluations. At present, there are mainly two ways for constructing the multipoint methods with memory, one is the inverse interpolation method, and the other is self-accelerating method. There are some representative iterative methods for inverse interpolation method. For example, Neta [2] first derived a fast three-step method of order 10.815. Using the same strategy, Petković et al. [3] proposed a two-step method of order 4.5612 and Gustavo [4] defined three derivative-free iterative methods with arbitrary high order convergence. For the self-accelerating method, the first iterative method is presented by Traub [5] in 1964, which is a derivative-free method using a self-accelerating parameter constructed by Newton interpolation. Motivated by Traub's idea, a two-step derivative-free iterative method is proposed by Petković et al. [6], which uses a self-accelerating parameter calculated by secant approach. Furthermore, Džunić et al. [7-8] gave some derivative-free methods based on a self-accelerating parameter, the self-accelerating parameter is calculated by Newton interpolation polynomial. Lotfi et al. [9] derived an effective way to convert the classical King's method [10] to a new derivative-free iterative method without memory and obtain a derivative-free iterative method with memory. We [11-13] proposed some Newton-type multipoint iterative methods with memory, in which the self-accelerating parameter is calculated by Hermite interpolation polynomial. For the self-accelerating method, the self-accelerating parameters are calculated by using information from the current and previous iterations, which do not increase the computational cost of the iterative method.

By increasing the number of self-accelerating parameters in the multipoint iterative method with memory, the convergence order of the multipoint iterative method can be improved greatly. Using two self-accelerating parameters, Cordero et al. [14] and Džunić et al. [15, 16] obtained some efficient derivative-free methods and we [17] also proposed some Newton-type multipoint iterative methods with memory. Using three self-accelerating parameters, Soleymani et al. [18], Lotfi et al. [19] and Wang et al. [20] presented some efficient derivative-free iterative methods, respectively. Using $n+1$self-accelerating parameters, we [21] derived a general derivative-free iterative method with the maximal convergence order $\left(2^{n+1}-1+\sqrt{2^{2(n+1)}+1}\right)/2$. Džunić method [8,16] and Lotfi method [19] can be seen as the special cases of our method [21]. Other multipoint iterative methods

with self-accelerating parameter are discussed in [22, 23].

By studying the above mentioned methods, we concluded that the self-accelerating methods can be expanded from the fixed-parameter multipoint methods without memory. Choosing suitable variable parameter to substitute the fixed-parameter of the multipoint methods without memory, we can improve the convergence order of the iterative method without memory and obtain the iterative method with memory. The variable parameter is called self-accelerating parameter. At present, most of the self-accelerating parameters are constructed by the interpolation polynomial. The self-accelerating parameter of the derivative-free multipoint iterative method is calculated by Newton interpolation polynomial. The self-accelerating parameter of the Newton-type multipoint iterative method is calculated by Hermite interpolation polynomial. For the self-accelerating parameter, the higher degree of interpolation polynomial may generate higher convergence order of the multipoint iterative method with memory and more complex computation of the self-accelerating parameter. Complex self-accelerating parameter will increase the computing time of the multipoint iterative method. So, a simple structure self-accelerating parameter is more suitable for practical application.

The main purpose of this paper is to give a new Ostrowski-type method with memory and some novel self-accelerating parameters with simple structure. This paper is organized as follows. In Section 2, based on Ostrowski's method [24], we derive Ostrowski-type method without memory for solving nonlinear equations and prove that the new method has the optimal local convergence order four. Some novel self-accelerating parameters with simple structure are given in Section 3. Based on the new iterative method without memory, a new Ostrowski-type method with memory is obtained by using the new self-accelerating parameter. The self-accelerating parameters are calculated by a new method without any additional functional evaluations. The maximal convergence order of the new Ostrowski-type methods with memory is 4.4495. Since acceleration of convergence is obtained without additional function evaluations, the new Ostrowski-type method with memory has higher computational efficiency. Numerical examples are given in Section 4 to confirm theoretical results. Section 5 is a short conclusion.

2. A new Ostrowski-type method without memory

The well known Ostrowski's method [24] can be written as

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

which has the optimal convergence order 4. The second step of method (2) is equivalent to $x_{n+1} = y_n - \dfrac{f(y_n)}{2f[x_n, y_n] - f'(x_n)}$.

Based on the method (2), we construct the following iterative method

$$\begin{cases} z_n = x_n - \dfrac{f(x_n)}{f'(x_n)}, \\ y_n = z_n - \lambda(z_n - x_n)^2, \\ x_{n+1} = y_n - \dfrac{f(y_n)}{2f[x_n, y_n] - f'(x_n)}, \end{cases} \tag{3}$$

where $\lambda \in R$ is a parameter. For the method (3), we have the following convergence analysis.

Theorem 1. Let $a \in I$ be a simple zero of a sufficiently differentiable function $f : I \subset R \to R$ for an open interval I. Then the iterative method defined by (3) is of fourth-order convergence and it satisfies the following error equation

$$e_{n+1} = (c_2 - \lambda)(c_2^2 - c_3 - c_2\lambda)e_n^4 + O(e_n^5). \tag{4}$$

Proof . Let $e_n = x_n - a$, $c_n = (1/n!) f^{(n)}(a) / f'(a)$, $n = 2, 3, \cdots$. Using the Taylor expansion and taking into account $f(a) = 0$, we have

$$f(x_n) = f'(a)[e_n + c_2 e_n^2 + c_3 e_n^3 + c_4 e_n^4 + c_5 e_n^5 + O(e_n^6)], \tag{5}$$

$$f'(x_n) = f'(a)[1 + 2c_2 e_n + 3c_3 e_n^2 + 4c_4 e_n^3 + 5c_5 e_n^4 + O(e_n^5)]. \tag{6}$$

From (3), (5) and (6), we get

$$\begin{aligned} e_{n,z} &= z_n - a = x_n - a - f(x_n) / f'(x_n) \\ &= c_2 e_n^2 + (-2c_2^2 + 2c_3)e_n^3 + (4c_2^3 - 7c_2 c_3 + 3c_4)e_n^4 + O(e_n^5). \end{aligned} \tag{7}$$

Using (3) and (7), we have

$$\begin{aligned} e_{n,y} &= y_n - a = z_n - a - \lambda(z_n - x_n)^2 \\ &= (c_2 - \lambda)e_n^2 + 2(-2c_2^2 + c_3 + c_2\lambda)e_n^3 + (4c_2^3 - 7c_2 c_3 + 3c_4 - 5c_2^2\lambda + 4c_3\lambda)e_n^4 + O(e_n^5). \end{aligned} \tag{8}$$

By a similar argument to that of (5), we have

$$f(y_n) = f'(a)[(c_2 - \lambda)e_n^2 + 2(-c_2^2 + c_3 + c_2\lambda)e_n^3 + (5c_2^3 + 3c_4 - 7c_2^2\lambda + 4c_3\lambda + c_2(-7c_3 + \lambda^2))e_n^4 + O(e_n^5)]. \tag{9}$$

From (5), (8) and (9), we obtain

$$f[x_n, y_n] = f'(a)(1 + c_2 e_n + (c_2^2 + c_3 - c_2\lambda)e_n^2 + (-2c_2^3 + 3c_2c_3 + c_4 + 2c_2^2\lambda - c_3\lambda)e_n^3 + O(e_n^4)). \quad (10)$$

Together with (5) and (8)-(10), we obtain the error equation

$$\begin{aligned} e_{n+1} &= x_{n+1} - a = y_n - a - f(y_n)(2f[x_n, y_n] - f'(x_n))^{-1} \\ &= (c_2 - \lambda)(c_2^2 - c_3 - \lambda c_2)e_n^4 + O(e_n^5). \end{aligned} \quad (11)$$

The proof is completed.

The developed method requires only three functional evaluations per step and reaches the optimal order of convergence four, which agrees with the conjecture of Kung-Traub [25].

Remark 1. From theorem 1, we know that the sequences $\{x_n\}$, $\{y_n\}$ and $\{z_n\}$ generated by the iterative method (3) converge to the simple root a of $f(x)$, provided that $f(x)$ is a sufficiently differentiable function. From equations (7), (8) and (11), we obtain

$$\lim_{n\to\infty} \frac{z_n - a}{(x_n - a)^2} = c_2, \quad (12)$$

$$\lim_{n\to\infty} \frac{y_n - a}{(x_n - a)^2} = c_2 - \lambda, \quad (13)$$

and

$$\lim_{n\to\infty} \frac{x_{n+1} - a}{(x_n - a)^4} = \lim_{n\to\infty} \frac{e_{n+1}}{e_n^4} = (c_2 - \lambda)(c_2^2 - c_3 - \lambda c_2). \quad (14)$$

3. A new Ostrowski-type method with memory and some novel self-accelerating parameters

In this section, we will improve the convergence order of the method (3) by using a simple self-accelerating parameter λ_n to substitute the parameter λ in (3). Assume that the parameter λ_n satisfies $\lim_{n\to\infty} \lambda_n = c_2 = f''(a)/(2f'(a))$, then the asymptotic convergence constant to be zero in (4). In general, the self-accelerating parameter λ_n can be calculated by Hermite interpolation polynomial. For example, using Hermite interpolation polynomial $H_2(x) = f(x_n) + f[x_n, x_n](x - x_n) + f[x_n, x_n, y_{n-1}](x - x_n)^2$ to approximate the function $f(x)$, we can construct the self-accelerating parameter $\lambda_n = H_2''(x_n)/(2f'(x_n))$. Using the different degrees of Hermite interpolation polynomials, we can obtain different self-accelerating parameters. But, we do not use interpolation method to construct the self-accelerating parameter in this paper. Here, we will give a novel method to construct the self-accelerating parameter.

From the equation (12), we can see that the asymptotic error constant of the fist step of the new method (3) is c_2. Thus, the following expression

$$\lambda_n = \frac{z_n - a}{(x_n - a)^2} \tag{15}$$

can be the self-accelerating parameter. Since the root a in (15) is unknown, we use information from the current and previous iterations to approximate the equation (15) and obtain some new self-accelerating parameters.

Using $z_{n-1} - x_n$ and $z_{n-1} - x_{n-1}$ to substitute $z_n - a$ and $x_n - a$ in (15), respectively, we obtain the following scheme

Scheme 1:

$$\lambda_n = \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})^2}. \tag{16}$$

From (16), we get

$$\lambda_n = \frac{(z_{n-1} - a) - (x_n - a)}{[(z_{n-1} - a) - (x_{n-1} - a)]^2} = \frac{\frac{z_{n-1} - a}{(x_{n-1} - a)^2} - \frac{x_n - a}{(x_{n-1} - a)^2}}{(\frac{z_{n-1} - a}{x_{n-1} - a} - 1)^2}. \tag{17}$$

According to (12) and (14), we have

$$\frac{z_{n-1} - a}{x_{n-1} - a} \to 0 \tag{18}$$

and

$$\frac{x_n - a}{(x_{n-1} - a)^2} \to 0, \tag{19}$$

provided that $n \to \infty$.

Thus, we obtain

$$\lim_{n\to\infty} \lambda_n = \lim_{n\to\infty} \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})^2} = \lim_{n\to\infty} \frac{\frac{z_{n-1} - a}{(x_{n-1} - a)^2} - \frac{x_n - a}{(x_{n-1} - a)^2}}{(\frac{z_{n-1} - a}{x_{n-1} - a} - 1)^2} = \lim_{n\to\infty} \frac{z_{n-1} - a}{(x_{n-1} - a)^2} = c_2. \tag{20}$$

From (20), we know that Scheme 1 can be as the self-accelerating parameter.

Using $z_{n-1} - x_n$ and $(z_{n-1} - x_{n-1})(y_{n-1} - x_{n-1})$ to substitute $z_n - a$ and $(x_n - a)^2$ in (15), respectively, we obtain the following scheme

Scheme 2: $$\lambda_n = \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})(y_{n-1} - x_{n-1})}. \tag{21}$$

By a similar argument to that of (16), we have

$$\lim_{n\to\infty}\lambda_n=\lim_{n\to\infty}\frac{z_{n-1}-x_n}{(z_{n-1}-x_{n-1})(y_{n-1}-x_{n-1})}=\lim_{n\to\infty}\frac{\frac{z_{n-1}-a}{(x_{n-1}-a)^2}-\frac{x_n-a}{(x_{n-1}-a)^2}}{(\frac{z_{n-1}-a}{x_{n-1}-a}-1)(\frac{y_{n-1}-a}{x_{n-1}-a}-1)}, \tag{22}$$

From (13), we get

$$\frac{y_{n-1}-a}{x_{n-1}-a}\to 0 \quad (n\to\infty) \tag{23}$$

According to (18), (19), (22) and (23), we obtain

$$\lim_{n\to\infty}\lambda_n=\lim_{n\to\infty}\frac{z_{n-1}-x_n}{(z_{n-1}-x_{n-1})(y_{n-1}-x_{n-1})}=\lim_{n\to\infty}\frac{z_{n-1}-a}{(x_{n-1}-a)^2}=c_2. \tag{24}$$

From (24), we know that Scheme 2 can be as the self-accelerating parameter. Using combination method, we construct Schemes 3 and 4.

Scheme 3: $\lambda_n=\dfrac{1}{(x_{n-1}-z_{n-1})}\left\{\dfrac{(x_n-z_{n-1})}{(z_{n-1}-x_{n-1})}+\dfrac{2\,(z_n-x_n)}{(x_n-y_{n-1})}\right\}.$ (25)

By a similar argument to that of (15), we have

$$\begin{aligned}\lim_{n\to\infty}\lambda_n&=\lim_{n\to\infty}\frac{1}{(x_{n-1}-z_{n-1})}\left\{\frac{(x_n-z_{n-1})}{(z_{n-1}-x_{n-1})}+\frac{2\,(z_n-x_n)}{(x_n-y_{n-1})}\right\}\\&=\lim_{n\to\infty}\left\{\frac{(z_{n-1}-x_n)}{(z_{n-1}-x_{n-1})^2}+\frac{2\,(z_n-x_n)}{(x_{n-1}-z_{n-1})(x_n-y_{n-1})}\right\},\\&=\lim_{n\to\infty}\frac{(z_{n-1}-x_n)}{(z_{n-1}-x_{n-1})^2}+\lim_{n\to\infty}\frac{2\,(z_n-x_n)}{(x_{n-1}-z_{n-1})(x_n-y_{n-1})},\\&=c_2+\lim_{n\to\infty}\frac{2\,(z_n-x_n)}{(x_{n-1}-z_{n-1})(x_n-y_{n-1})},\end{aligned} \tag{26}$$

$$\lim_{n\to\infty}\frac{2\,(z_n-x_n)}{(x_{n-1}-z_{n-1})(x_n-y_{n-1})}=2\lim_{n\to\infty}\frac{\frac{z_n-a}{(x_{n-1}-a)^2}-\frac{x_n-a}{(x_{n-1}-a)^2}}{(1-\frac{z_{n-1}-a}{x_{n-1}-a})(\frac{x_n-a}{x_{n-1}-a}-\frac{y_{n-1}-a}{x_{n-1}-a})}$$

$$=2\lim_{n\to\infty}\frac{\frac{z_n-a}{x_n-a}\cdot\frac{x_n-a}{(x_{n-1}-a)^2}-\frac{x_n-a}{(x_{n-1}-a)^2}}{(1-\frac{z_{n-1}-a}{x_{n-1}-a})(\frac{x_n-a}{x_{n-1}-a}-\frac{y_{n-1}-a}{x_{n-1}-a})}=0 \tag{27}$$

Using (26) and (27), we obtain

$$\lim_{n\to\infty}\lambda_n=\lim_{n\to\infty}\frac{1}{(x_{n-1}-z_{n-1})}\left\{\frac{(x_n-z_{n-1})}{(z_{n-1}-x_{n-1})}+\frac{2\,(z_n-x_n)}{(x_n-y_{n-1})}\right\}=c_2. \tag{28}$$

Scheme 4:

$$\lambda_n=\frac{1}{3(x_{n-1}-z_{n-1})}\left\{\frac{6(z_n-x_n)}{(x_n-y_{n-1})}+\frac{5(x_n-z_{n-1})}{(z_{n-1}-x_{n-1})}-\frac{4(x_n-z_{n-1})}{(y_{n-1}-x_{n-1})}\right\}-\frac{2(x_n-z_{n-1})(y_{n-1}-x_{n-1})}{3(x_n-x_{n-1})^3}. \tag{29}$$

Using the similar arguments to that of (16), (21) and (25), we have

$$\lim_{n\to\infty}\lambda_n=\lim_{n\to\infty}\frac{1}{3(x_{n-1}-z_{n-1})}\left\{\frac{6(z_n-x_n)}{(x_n-y_{n-1})}+\frac{5(x_n-z_{n-1})}{(z_{n-1}-x_{n-1})}-\frac{4(x_n-z_{n-1})}{(y_{n-1}-x_{n-1})}\right\}-\frac{2(x_n-z_{n-1})(y_{n-1}-x_{n-1})}{3(x_n-x_{n-1})^3}$$

$$=c_2 \tag{30}$$

In order to save space, we omit the detail calculating course of (30).

Remark 2. Now, we obtain the following one-parameter Ostrowski-type iterative method with memory

$$\begin{cases} z_n=x_n-\dfrac{f(x_n)}{f'(x_n)}, \\ y_n=z_n-\lambda_n(z_n-x_n)^2, \\ x_{n+1}=y_n-\dfrac{f(y_n)}{2f[x_n,y_n]-f'(x_n)}. \end{cases} \tag{31}$$

The self-accelerating parameter λ_n is calculated by using one of the formulas (16), (21), (25) and (29).

The concept of the *R*-order of convergence [1] and the following assertion (see [26, p.287]) will be applied to estimate the convergence order of the Ostrowski-type method with memory (31).

Theorem 2. If the errors of approximations $e_j=x_j-a$ obtained in an iterative root-finding 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, denoted with $O_R(\mathrm{IM},a)$, satisfies the inequality $O_R(\mathrm{IM},a)\geq s^*$, where 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 varying parameters λ_n in the iterative method (31) be calculated by (16) or (21), respectively. If an initial approximation x_0 is sufficiently close to a simple root a of $f(x)$, then the *R*-order of convergence of the iterative methods (31) with memory is at least $2+\sqrt{5}\approx 4.2361$.

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

$$e_{n+1} \sim D_{n,r} e_n^r,\ e_n = x_n - a, \tag{32}$$

where $D_{n,r}$ tends to the asymptotic error constant D_r of (IM) when $n \to \infty$. So,

$$e_{n+1} \sim D_{n,r}(D_{n-1,r} e_{n-1}^r)^r = D_{n,r} D_{n-1,r}^r e_{n-1}^{r^2}. \tag{33}$$

It is similar to the derivation of (33). We assume that the iterative sequence $\{y_n\}$ has the *R*-order p, then

$$e_{n,y} \sim D_{n,p} e_n^p \sim D_{n,p}(D_{n-1,r} e_{n-1}^r)^p = D_{n,p} D_{n-1,r}^p e_{n-1}^{rp}. \tag{34}$$

Using (8) and (11), we obtain the corresponding error relations for the methods with memory (31)

$$e_{n,y} = y_n - a \sim (c_2 - \lambda_n) e_n^2, \tag{35}$$

$$e_{n+1} = x_{n+1} - a \sim (c_2 - \lambda_n)(c_2^2 - c_3 - \lambda_n c_2) e_n^4. \tag{36}$$

Here, the higher order terms in (35)-(36) are omitted.

Scheme 1, λ_n is calculated by (16):

Substituting λ by λ_n and n by $n-1$ in (7) and (11), we get

$$\begin{aligned} z_{n-1} - x_n &= (z_{n-1} - a) - (x_n - a) = e_{n-1,z} - e_n \\ &= c_2 e_{n-1}^2 + (-2c_2^2 + 2c_3) e_{n-1}^3 + (3c_2^3 + 3c_4 + 2c_2^2 \lambda_{n-1} - c_3 \lambda_{n-1} - c_2(6c_3 + \lambda_{n-1}^2)) e_{n-1}^4 + O(e_n^5), \end{aligned} \tag{37}$$

$$\begin{aligned} z_{n-1} - x_{n-1} &= (z_{n-1} - a) - (x_{n-1} - a) = e_{n-1,z} - e_{n-1} \\ &= -e_{n-1} + c_2 e_{n-1}^2 + (-2c_2^2 + 2c_3) e_{n-1}^3 + (4c_2^3 - 7c_2 c_3 + 3c_4) e_{n-1}^4 + O(e_n^5). \end{aligned} \tag{38}$$

Using (37) and (38), we have

$$\begin{aligned} \lambda_n &= \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})^2} = \frac{e_{n-1,z} - e_n}{(e_{n-1,z} - e_{n-1})^2} \\ &= \frac{c_2 e_{n-1}^2 + (-2c_2^2 + 2c_3) e_{n-1}^3 + (3c_2^3 + 3c_4 + 2c_2^2 \lambda_{n-1} + c_3 \lambda_{n-1} + c_2(6c_3 + \lambda_{n-1}^2)) e_{n-1}^4 + O(e_n^5)}{(e_{n-1} - c_2 e_{n-1}^2 - (-2c_2^2 + 2c_3) e_{n-1}^3 - (4c_2^3 - 7c_2 c_3 + 3c_4) e_{n-1}^4 + O(e_n^5))^2} \\ &= \frac{c_2 + (-2c_2^2 + 2c_3) e_{n-1} + (3c_2^3 + 3c_4 + 2c_2^2 \lambda_{n-1} + c_3 \lambda_{n-1} + c_2(6c_3 + \lambda_{n-1}^2)) e_{n-1}^2 + O(e_n^3)}{(1 - c_2 e_{n-1} - (-2c_2^2 + 2c_3) e_{n-1}^2 - (4c_2^3 - 7c_2 c_3 + 3c_4) e_{n-1}^3 + O(e_n^4))^2} \\ &= \frac{c_2 + (-2c_2^2 + 2c_3) e_{n-1} + (3c_2^3 + 3c_4 + 2c_2^2 \lambda_{n-1} + c_3 \lambda_{n-1} + c_2(6c_3 + \lambda_{n-1}^2)) e_{n-1}^2 + O(e_n^3)}{1 - 2c_2 e_{n-1} + (5c_2^2 - 4c_3) e_{n-1}^2 + (18c_2 c_3 - 12c_2^3 - 6c_4) e_{n-1}^3 + O(e_n^4)} \\ &\sim [c_2 + (-2c_2^2 + 2c_3) e_{n-1} + (3c_2^3 + 3c_4 + 2c_2^2 \lambda_{n-1} + c_3 \lambda_{n-1} + c_2(6c_3 + \lambda_{n-1}^2)) e_{n-1}^2 + O(e_n^3)] \\ &\quad \times \{1 + [2c_2 e_{n-1} - (5c_2^2 - 4c_3) e_{n-1}^2 + O(e_n^3)] + [2c_2 e_{n-1} - (5c_2^2 - 4c_3) e_{n-1}^2 + O(e_n^3)]^2\} \\ &\sim c_2 + 2c_3 e_{n-1} - (2c_2^3 - 2c_2 c_3 - 3c_4 - 3\lambda_{n-1} c_2^2 + 3c_3 \lambda_{n-1}) e_{n-1}^2 + O(e_{n-1}^3), \end{aligned} \tag{39}$$

$$c_2 - \lambda_n \sim -2c_3 e_{n-1}. \tag{40}$$

According to (36) and (40), we get

$$e_{n+1} \sim -2c_3 e_{n-1}(c_2^2 - c_3 - \lambda_n c_2)e_n^4 \sim 2c_3^2 e_{n-1}(D_{n-1,r} e_{n-1}^r)^4$$
$$\sim 2c_3^2 D_{n-1,r}^4 e_{n-1}^{4r+1}. \quad (41)$$

By comparing exponents of e_{n-1} appearing in the relations (33) and (41), we get the following equations

$$4r + 1 = r^2. \quad (42)$$

Positive solution of the equation (42) is given by $r = 2 + \sqrt{5} \approx 4.2361$. Therefore, the *R*-order of the method with memory (31), when λ_n is calculated by (16), is at least 4.2361.

Scheme 2, λ_n is calculated by (21):

Substituting λ by λ_n and n by $n-1$ in (8), we obtain

$$y_{n-1} - x_{n-1} = e_{n-1,y} - e_{n-1} = -e_{n-1} + (c_2 - \lambda_{n-1})e_{n-1}^2 + 2(-2c_2^2 + c_3 + c_2\lambda_{n-1})e_{n-1}^3$$
$$+(4c_2^3 - 7c_2c_3 + 3c_4 - 5c_2^2\lambda_{n-1} + 4c_3\lambda_{n-1})e_{n-1}^4 + O(e_{n-1}^5). \quad (43)$$

Using (37), (38) and (43), we get

$$\lambda_n = \frac{z_{n-1} - x_n}{(z_{n-1} - x_{n-1})(y_{n-1} - x_{n-1})} = \frac{e_{n-1,z} - e_n}{(e_{n-1,z} - e_{n-1})(e_{n-1,y} - e_{n-1})}$$

$$= \frac{c_2 + (-2c_2^2 + 2c_3)e_{n-1} + (3c_2^3 + 3c_4 + 2c_2^2\lambda_{n-1} + c_3\lambda_{n-1} + c_2(6c_3 + \lambda_{n-1}^2))e_{n-1}^2 + O(e_n^3)}{(1 - c_2e_{n-1} - (-2c_2^2 + 2c_3)e_{n-1}^2 + O(e_n^3))(1 - (c_2 - \lambda_{n-1})e_{n-1} - 2(-c_2^2 + c_3 + c_2\lambda_{n-1})e_{n-1}^2 + O(e_n^3))}$$

$$= c_2 - (c_2\lambda_{n-1} - 2c_3)e_{n-1} - (2c_2^3 - 2c_2c_3 - 3c_4 - 3c_2^2\lambda_{n-1} + 3c_3\lambda_{n-1})e_{n-1}^2 + O(e_{n-1}^3), \quad (44)$$

$$c_2 - \lambda_n \sim (c_2\lambda_{n-1} - 2c_3)e_{n-1}. \quad (45)$$

From (40) and (45), we can see that scheme 1 and scheme 2 have the same error level. Therefore, we conclude that the *R*-order of the methods with memory (31), when λ_n is calculated by (21), is at least 4.2361.

The proof is completed.

Theorem 4. Let the varying parameters λ_n in the iterative method (31) be calculated by (25) and (29), respectively. If an initial approximation x_0 is sufficiently close to a simple root a of $f(x)$, then the *R*-order of convergence of the iterative methods (31) with memory is at least $\left(5 + \sqrt{13}\right)/2 \approx 4.30278$ and $2 + \sqrt{6} \approx 4.4495$, respectively.

Proof. **Scheme 3**, λ_n is calculated by (25):

According to (7), (8) and (21), we have

$$x_n - z_n = e_n - e_{n,z} = e_n - c_2e_n^2 - (-2c_2^2 + 2c_3)e_n^3 - (4c_2^3 - 7c_2c_3 + 3c_4)e_n^4 + O(e_n^5)$$
$$= (c_2 - \lambda_{n-1})(c_2^2 - c_3 - c_2\lambda_{n-1})e_{n-1}^4 - 2(2c_2^4 + c_3^2 - (4c_2^3 + c_4 + c_3\lambda_{n-1})\lambda_{n-1}$$

$$-c_2(c_4+5c_3\lambda_{n-1})-c_2^2(-4c_3+2\lambda_{n-1}^2))e_{n-1}^5+O(e_{n-1}^6), \tag{46}$$

$$x_n-y_{n-1}=e_n-e_{n-1,y}=e_n-(c_2-\lambda_{n-1})e_{n-1}^2+2(-2c_2^2+c_3+c_2\lambda_{n-1})e_{n-1}^3$$

$$-(4c_2^3-7c_2c_3+3c_4-5c_2^2\lambda_{n-1}+4c_3\lambda_{n-1})e_{n-1}^4+O(e_{n-1}^5)$$

$$=-(c_2-\lambda_{n-1})e_{n-1}^2-2(c_3-c_2^2+c_2\lambda_{n-1})e_{n-1}^3+(-3c_2^3+3c_2^2\lambda_{n-1}$$

$$-3(c_4+c_3\lambda_{n-1})+c_2(6c_3+\lambda_{n-1}^2))e_{n-1}^4+O(e_{n-1}^5), \tag{47}$$

Using (39) and (46)-(47), we have

$$\lambda_n=\frac{1}{(x_{n-1}-z_{n-1})}\left\{\frac{(x_n-z_{n-1})}{(z_{n-1}-x_{n-1})}+\frac{2\,(z_n-x_n)}{(x_n-y_{n-1})}\right\}$$

$$=\frac{1}{(e_{n-1}-e_{n-1,z})}\left\{\frac{(e_n-e_{n-1,z})}{(e_{n-1,z}-e_{n-1})}+\frac{2\,(e_{n,z}-e_n)}{(e_n-e_{n-1,y})}\right\}$$

$$=c_2+2c_2(c_2-\lambda_{n-1})e_{n-1}+(-4c_2^3+8c_2c_3-c_4+4c_2^2\lambda_{n-1}-5c_3\lambda_{n-1}-c_2\lambda_{n-1}^2)e_{n-1}^2+O(e_{n-1}^3), \tag{48}$$

$$c_2-\lambda_n\sim-2c_2(c_2-\lambda_{n-1})e_{n-1}. \tag{49}$$

Taking the $h_n=c_2-\lambda_n$ in (49), we get

$$h_n\sim-2c_2h_{n-1}e_{n-1}\sim-2c_2(-2c_2h_{n-2}e_{n-2})e_{n-1}\sim(-2c_2)^2h_{n-2}e_{n-2}e_{n-1}$$

$$\sim(-2c_2)^kh_{n-k}e_{n-1}e_{n-2}\cdots e_{n-k}$$

$$\sim(-2c_2)^nh_0\prod_{k=1}^{n}e_{n-k}. \tag{50}$$

Assume that the error of the iterative sequence $\{x_k\}$ can be expressed as follows

$$e_k\sim D_ke_0^{r_k},\ 1\le k\le n, \tag{51}$$

where $e_0=x_0-a$ and $e_k=x_k-a$.

From (11), (50) and (51), we get

$$e_k=x_k-a\sim(c_2-\lambda_{k-1})(c_2^2-c_3-\lambda_{k-1}c_2)e_{k-1}^4$$

$$\sim(-2c_2)^{k-1}h_0(c_2^2-c_3-\lambda_{k-1}c_2)e_{k-1}^4\prod_{j=1}^{k-1}e_{k-j-1}$$

$$\sim(-2c_2)^{k-1}h_0(c_2^2-c_3-\lambda_{k-1}c_2)D_{k-1}^4e_0^{4r_{k-1}}e_0^{1+r_1+r_2+\cdots+r_{k-2}}\prod_{j=0}^{k-2}D_j. \tag{52}$$

Comparing the exponents of the error e_0 in pairs of relations ((51), (52)) for $2\le k\le n$ and ((11), (51)) for $k=1$, we obtain the system of equations

$$\begin{cases} r_n = 1 + r_1 + r_2 + \cdots + r_{n-2} + 4r_{n-1}, \\ r_k = 1 + r_1 + \cdots + r_{k-2} + 4r_{k-1}, \ 3 \le k \le n-1, \\ r_2 = 1 + 4r_1, \\ r_1 = 4. \end{cases} \tag{53}$$

From the (53), we have

$$r_n = 5r_{n-1} - 3r_{n-2}. \tag{54}$$

Dividing (54) by r_{n-1}, we get

$$\frac{r_n}{r_{n-1}} = 5 - 3\frac{r_{n-2}}{r_{n-1}}. \tag{55}$$

Letting $\lim_{n\to\infty}(r_n / r_{n-1}) = \lim_{n\to\infty}(r_{n-1} / r_{n-2}) = R$, we have

$$R = 5 - 3\frac{1}{R}, \tag{56}$$

and

$$R^2 - 5R + 3 = 0. \tag{57}$$

The positive solution of the equation (57) is given by $R = \left(5 + \sqrt{13}\right)/2$. Therefore, the convergence order of the method (31) with memory is $R = \left(5 + \sqrt{13}\right)/2 \approx 4.30278$.

Scheme 4, λ_n is calculated by (29):

From (36), we obtain

$$x_n - x_{n-1} = e_n - e_{n-1} = -e_{n-1} + (c_2 - \lambda_{n-1})(c_2^2 - c_3 - \lambda_{n-1}c_2)e_{n-1}^4 + O(e_{n-1}^5). \tag{58}$$

According to (37), (38), (43), (46), (47) and (58), we have

$$\lambda_n = \frac{1}{3(x_{n-1} - z_{n-1})}\left\{\frac{6(z_n - x_n)}{(x_n - y_{n-1})} + \frac{5(x_n - z_{n-1})}{(z_{n-1} - x_{n-1})} - \frac{4(x_n - z_{n-1})}{(y_{n-1} - x_{n-1})}\right\} - \frac{2(x_n - z_{n-1})(y_{n-1} - x_{n-1})}{3(x_n - x_{n-1})^3}$$

$$= c_2 - (-2c_2^3 + c_4 + c_3\lambda_{n-1} + \frac{7c_2\lambda_{n-1}^2}{3})e_{n-1}^2 + O(e_{n-1}^3), \tag{59}$$

$$c_2 - \lambda_n \sim (c_4 - 2c_2^3 + c_3\lambda_{n-1} + \frac{7c_2\lambda_{n-1}^2}{3})e_{n-1}^2. \tag{60}$$

Using (36) and (60), we get

$$\begin{aligned} e_{n+1} &\sim (c_4 - 2c_2^3 + c_3\lambda_{n-1} + \frac{7}{3}c_2\lambda_{n-1}^2)e_{n-1}^2(c_2^2 - c_3 - \lambda_n c_2)e_n^4 \\ &\sim (c_4 - 2c_2^3 + c_3\lambda_{n-1} + \frac{7}{3}c_2\lambda_{n-1}^2)e_{n-1}^2(D_{n-1,r}e_{n-1}^r)^4 \\ &\sim (c_4 - 2c_2^3 + c_3\lambda_{n-1} + \frac{7}{3}c_2\lambda_{n-1}^2)D_{n-1,r}^4 e_{n-1}^{4r+2}. \end{aligned} \tag{61}$$

By comparing exponents of e_{n-1} appearing in equations (33) and (61), we get the following equation

$$4r + 2 = r^2 \tag{62}$$

Positive solution of the equation (62) is given by $r = 2 + \sqrt{6} \approx 4.4495$. Therefore, the *R*-order of the methods with memory (31), when λ_n is calculated by (29), is at least 4.4495.

The proof is completed.

4. Numerical results

The new methods (3) and (31) are employed to solve nonlinear equations $f_i(x)(i = 1,2,3)$ and compared with Ostrowski's method (OM, (2)), Petković-Džunić-Neta's method (PDN, (63)), Petković-Ilić-Džunić's method (PID, (64)) and Wang-Zhang's methods (WZ1, (65) and (WZ2, (66)). The iterative methods used in the numerical experiment are as follows:

Petković-Džunić-Neta's method (PDN, see [3])

$$\begin{cases} y_{-1} = N(x_0), N(x_n) = x_n - \dfrac{f(x_n)}{f'(x_n)},\ \phi(t) = \dfrac{1}{f(t) - f(x_n)}\left[\dfrac{t - x_n}{f(t) - f(x_n)} - \dfrac{1}{f'(x_n)}\right], \\ y_n = N(x_n) + f(x_n)^2 \phi(y_{n-1}), \\ x_{n+1} = N(x_n) + f(x_n)^2 \phi(y_n), \end{cases} \tag{63}$$

Petković-Ilić-Džunić's method with memory (PID, see [6])

$$\begin{cases} y_n = x_n - \dfrac{f(x_n)}{f[x_n, z_n]}, \\ x_{n+1} = y_n - \dfrac{f(y_n)}{f[x_n, z_n]}\left(1 + \dfrac{f(y_n)}{f(x_n)} + \dfrac{f(y_n)}{f(z_n)}\right), \end{cases} \tag{64}$$

where $z_n = x_n - \gamma_n f(x_n)$ and $\gamma_n = (x_n - x_{n-1}) / (f(x_n) - f(x_{n-1}))$.

Wang-Zhang's method (WZ1, see [9])

$$\begin{cases} y_n = x_n - \dfrac{f(x_n)}{\lambda_n f(x_n) + f'(x_n)},\ \lambda_n = -\dfrac{f[x_n, y_{n-1}, x_{n-1}]}{f'(x_n)}, \\ x_{n+1} = y_n - \dfrac{f(y_n)}{2\lambda_n f(x_n) + f'(x_n)}\left(1 + 2\dfrac{f(y_n)}{f(x_n)} + \left(\dfrac{f(y_n)}{f(x_n)}\right)^2\right), \end{cases} \tag{65}$$

Wang-Zhang's method (WZ2, see [17])

$$\begin{cases} y_n = x_n - \dfrac{f(x_n)}{f'(x_n) - \gamma_n f(x_n)}, \gamma_n = \dfrac{H_4''(x_n)}{2f'(x_n)}, \beta_n = -\dfrac{H_4'''(x_n)}{6f'(x_n)}, \\ x_{n+1} = y_n - \dfrac{f(y_n)}{2f[x_n, y_n] - f'(x_n)}\left(1 + \beta_n \left(\dfrac{f(y_n)}{f(x_n)}\right)^2\right). \end{cases} \tag{66}$$

where $H_4''(x_n) = 2f[x_n, x_n, y_{n-1}] + (4f[x_n, x_n, y_{n-1}, x_{n-1}] - 2f[x_n, y_{n-1}, x_{n-1}, x_{n-1}])(x_n - y_{n-1})$ and $H_4'''(x_n)$ $= 6f[x_n, x_n, y_{n-1}, x_{n-1}] + 6f[x_n, x_n, y_{n-1}, x_{n-1}, x_{n-1}](2x_n - x_{n-1} - y_{n-1})$.

The absolute errors $|x_k - a|$ in the first four iterations are given in Tables 1-3, where a is the exact root computed with 1200 significant digits. The parameters $\lambda = 0.1$, $\lambda_0 = 0.1$ and $\gamma_0 = 0.1$ are used in the first iteration. The computational order of convergence ρ is defined by [27]:

$$\rho \approx \frac{\ln(|x_{n+1} - x_n| / |x_n - x_{n-1}|)}{\ln(|x_n - x_{n-1}| / |x_{n-1} - x_{n-2}|)}. \tag{67}$$

Following test functions are used:

$$f_1(x) = xe^{x^2} - \sin^2(x) + 3\cos(x) + 5, \quad a \approx -1.2076478271309189, \quad x_0 = -1.3,$$
$$f_2(x) = x^5 + x^4 + 4x^2 - 15, \quad a \approx 1.3474280989683050, \quad x_0 = 1.6,$$
$$f_3(x) = \arcsin(x^2 - 1) - 0.5x + 1, \quad a \approx 0.59481096839836918, \quad x_0 = 1.$$

Table 1 Numerical results for $f_1(x)$ by the methods with and without memory

Methods	$\lvert x_1 - a\rvert$	$\lvert x_2 - a\rvert$	$\lvert x_3 - a\rvert$	$\lvert x_4 - a\rvert$	ρ
OM	0.35032e-4	0.57502e-18	0.41734e-73	0.11580e-293	4.0000000
(3)	0.36743e-4	0.74187e-18	0.12328e-72	0.94016e-292	4.0000000
PID	0.47398e-4	0.30448e-17	0.20742e-73	0.28696e-311	4.2348771
WZ1	0.51371e-3	0.47887e-13	0.10861e-55	0.26710e-236	4.2352409
(31),(16)	0.52829e-4	0.90363e-17	0.77056e-72	0.11150e-304	4.2281273
(31),(21)	0.52829e-4	0.90363e-17	0.77056e-72	0.11150e-304	4.2281273
(31),(25)	0.52829e-4	0.11356e-16	0.27652e-71	0.33168e-306	4.3015188
(31),(29)	0.52829e-4	0.77707e-18	0.14271e-79	0.32057e-354	4.4487581

Table 2 Numerical results for $f_2(x)$ by the methods with and without memory

Methods	$\lvert x_1-a\rvert$	$\lvert x_2-a\rvert$	$\lvert x_3-a\rvert$	$\lvert x_4-a\rvert$	ρ
OM	0.15615e-2	0.31086e-11	0.48918e-46	0.29995e-185	4.0000000
(3)	0.15226e-2	0.27243e-11	0.27965e-46	0.31051e-186	4.0000000
PID	0.34012e-1	0.56555e-6	0.47462e-26	0.36556e-111	4.2395330
WZ1	0.80255e-2	0.10174e-8	0.35663e-38	0.66482e-163	4.2345381
(31),(16)	0.11891e-2	0.54502e-12	0.84967e-52	0.22950e-220	4.2346273
(31),(21)	0.11891e-2	0.48085e-12	0.55357e-53	0.40945e-226	4.2290146
(31),(25)	0.11891e-2	0.12774e-11	0.22808e-50	0.63052e-217	4.2984748
(31),(29)	0.11891e-2	0.17703e-12	0.17730e-56	0.25267e-252	4.4511139

Table 3 Numerical results for $f_3(x)$ by the methods with and without memory

Methods	$\lvert x_1-a\rvert$	$\lvert x_2-a\rvert$	$\lvert x_3-a\rvert$	$\lvert x_4-a\rvert$	ρ
OM	0.32137e-2	0.49177e-11	0.27132e-46	0.25139e-187	4.0000000
(3)	0.32174e-2	0.48262e-11	0.24591e-46	0.16574e-187	4.0000000
PID	0.28807e-2	0.17703e-12	0.14180e-55	0.35820e-238	4.2369602
WZ1	0.48052e-2	0.32661e-10	0.33069e-45	0.23474e-193	4.2334759
(31),(16)	0.31760e-2	0.14779e-10	0.18138e-46	0.19226e-198	4.2319727
(31),(21)	0.31760e-2	0.13767e-10	0.88209e-47	0.85788e-200	4.2276333
(31),(25)	0.31760e-2	0.31404e-11	0.44428e-50	0.29725e-217	4.3031516
(31),(29)	0.31760e-2	0.13010e-11	0.14489e-53	0.43173e-240	4.4460397

On the other hand, in Tables 4-5 the mean CPU time (in seconds), after 50 performances of the programs, appears. The stopping criterion are $|x_{k+1}-x_k|<10^{-200}$ and $|x_{k+1}-x_k|<10^{-300}$ in Tables 4 and 5, respectively.

Table 4 Mean CPU time for the stopping criterion $|x_{k+1}-x_k|<10^{-200}$

f	PDN	PID	WZ1	WZ2	(31),(16)	(31),(21)	(31),(25)	(31),(29)
f_1	1.7759	1.7038	1.5996	3.3034	**1.4083**	1.4155	1.4735	1.6907
f_2	0.9731	1.0062	1.2545	2.8123	1.1335	**0.9154**	0.9949	1.1303
f_3	1.2417	1.0682	1.3628	2.5802	1.1587	**0.9328**	1.0395	1.1506

Table 5 Mean CPU time for the stopping criterion $|x_{k+1} - x_k| < 10^{-300}$

f	PDN	PID	WZ1	WZ2	(31),(16)	(31),(21)	(31),(25)	(31),(29)
f_1	1.9949	1.7799	2.0486	3.5611	**1.4136**	1.4717	1.5375	1.7281
f_2	1.1640	1.1874	1.2845	2.6317	**1.0386**	1.0969	1.1462	1.3568
f_3	1.0473	1.0667	1.2573	2.4152	**1.0146**	1.0189	1.1157	1.3079

From the Tables 1-3, we can see that methods PID, WZ1, ((31),(16)) and ((31),(21)) have the similar convergence behaves, since these methods have the same convergence order. Tables 4-5 show that compared with other methods, our methods ((31),(16)) and ((31),(21)) use less computing time. It is noteworthy that methods WZ2 and PDN have higher convergence order, but they cost more computing time. Method WZ2 costs maximum computing time. The main reason is that the accelerating parameter of method WZ2 is too complex.

5. Conclusions

In this paper, we propose a novel method to construct the self-accelerating parameter of Ostrowski-type method. The new self-accelerating parameter has the properties of simple structure and easy calculation, which do not increase the computational cost of the iterative method. The new Ostrowski-type methods with and without memory are compared in performance with the existing methods by numerical examples. The numerical examples confirm the theoretical results and show that the new methods have better performance. We can conclude that complicated accelerating parameter can improve the convergence order and calculation precision of iterative methods, which also increase the computing time. Therefore, simple accelerating parameter is important for the iterative method with memory, which is the main research work of this paper.

Acknowledgments

The project supported by the National Natural Science Foundation of China (Nos. 11547005 and 61572082), Doctor Startup Foundation of Liaoning Province of China (No. 201501196) and the Educational Commission Foundation of Liaoning Province of China (No. L2015012).

References

[1] J. M. Ortega, W. C. Rheinbolt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.

[2] B. Neta, A New Family of Higher Order Methods for Solving Equations, Int. J. Comput. Math. 14 (1983), 191-195.

[3] M. S. Petković, J. Džunić, B. Neta, Interpolatory multipoint methods with memory for solving nonlinear equations, Appl. Math. Comput, 218 (2011) 2533-2541.

[4] G. Fernández-Torres, Derivative free iterative methods with memory of arbitrary high convergence order, Numer. Algor. 67 (2014) 565-580.

[5] J. F. Traub, Iterative Method for the Solution of Equations, Prentice hall, New York, 1964.

[6] M. S. Petković, S. Ilić, J. Džunić, Derivative free two-point methods with and without memory for solving nonlinear equations, Appl. Math. Comput. 217 (2010) 1887-1895

[7] J. Džunić, M. S. Petković, L. D. Petković, Three-point methods with and without memory for solving nonlinear equations, Appl. Math. Comput, 218 (2012) 4917-4927.

[8] J. Džunić, M. S. Petković, On generalized multipoint root-solvers with memory, J. Comput. Appl. Math, 236 (2012) 2909-2920.

[9] T Lotfi, Á. A. Magreñán, K. Mahdiani, J.Javier Rainer, A variant of Steffensen-King's type family with accelerated sixth-order convergence and high efficiency index: Dynamic study and approach. Appl. Math. Comput. 252(2015) 347-353.

[10] R. F. King, A family of fourth order methods for nonlinear equations, SIAM J. Numer. Anal. 10 (1973) 876-879.

[11] X. Wang, T. Zhang, A new family of Newton-type iterative methods with and without memory for solving nonlinear equations, Calcolo, 51 (2014) 1-15.

[12] X. Wang, T. Zhang, Some Newton-type iterative methods with and without memory for solving nonlinear equations, Int. J. Comput. Methods, 11 (2014) 1350078.

[13] X. Wang et al. A Family of Newton Type Iterative Methods for Solving Nonlinear Equations, Algorithms, 8 (2015) 786-798.

[14] A. Cordero, T. Lotfi, P. Bakhtiari, J. R. Torregrosa, An efficient two-parametric family with memory for nonlinear equations, Numer. Algor. 68 (2015) 323-335

[15] J. Džunić, On efficient two-parameter methods for solving nonlinear equations, Numer. Algor, 63 (2013) 549-569.

[16] J. Džunić, M. S. Petković, On generalized biparametric multipoint root finding methods with memory, J.

Comput. Appl. Math, 255 (2014) 362-375.

[17] X. Wang, T. Zhang, High-order Newton-type iterative methods with memory for solving nonlinear equations, Math. Commun. 19 (2014) 91-109.

[18] F. Soleymani, T. Lotfi, E. Tavakoli, F. K. Haghani, Several iterative methods with memory using-accelerators, Appl. Math. Comput., 254 (2015) 452-458.

[19] T. Lotfi, P. Assari, New three- and four-parametric iterative with memory methods with efficiency index near 2, Appl. Math. Comp. 270 (2015) 1004-1010

[20] X. Wang, T. Zhang, Y. Qin, Efficient two-step derivative-free iterative methods with memory and their dynamics, Int. J. Comput. Math., 93 (2016)1423-1446

[21] X. Wang, T. Zhang, Efficient n-point iterative methods with memory for solving nonlinear equations, Numer. Algor., 70 (2015) 357-375.

[22] J. R. Sharma, R. K. Guha, P. Gupta, Some efficient derivative free methods with memory for solving nonlinear equations, Appl. Math. Comp. 219 (2012) 699-707.

[23] M. Kansal, V. Kanwar, S. Bhatia, Efficient derivative-free variants of Hansen-Patrick's family with memory for solving nonlinear equations, Numer. Algor., (2016) doi:10.1007/s11075-016-0127-6

[24] A. M. Ostrowski, Solution of equations in Euclidean and Banach space, Academic Press, New York, 1973.

[25] H. T. Kung, J. F. Traub, Optimal order of one-point and multipoint iterations, J. Appl. Comput. Math. 21 (1974) 643-651.

[26] G. Alefeld, J. Herzberger, Introduction to Interval Computation, Academic Press, New York,1983.

[27] A. Cordero, J. R. Torregrosa, Variants of Newton's Method using fifth-order quadrature formulas. Appl. Math. Comput. 190 (2007) 686-698.