Journal of Mathematical Extension
Vol. 14, No. 2, (2020), 171-188
ISSN: 1735-8299
URL: http://www.ijmex.com

On Developing an Adaptive Free-Derivative Kung and Traub's Method with Memory

M. J. Lalehchini
Hamedan Branch, Islamic Azad University

T. Lotfi*
Hamedan Branch, Islamic Azad University

K. Mahdiani
Hamedan Branch, Islamic Azad University

Abstract. In this paper, there has been developed an adaptive method with memory based on the first two steps of the general optimal method without memory by Kung and Traub. Included is the convergence analysis along with numerical implementations. The proposed adaptive method with memory can increase the efficiency index of the optimal method without memory as high as possible while reusing all the available information of the functional evaluations in the previous iteration for this purpose. Also, we discuss the numerical stability from the point of basins of attraction.

AMS Subject Classification: 65H05; 49M15; 65J15
Keywords and Phrases: Nonlinear equation, method with and without memory, adaptive method, efficiency index, basins of attraction

1. Introduction

Of the most important problems in science and engineering , one is solving a nonlinear equation. It is well-known that solving such a problem

Received: December 2018; Accepted: July 2019
*Corresponding author

V generally not have analytical solution; hence,it is numerical methods which are preferred [14, 16]. To construct a numerical method and specially iterative methods, we should consider some computational criteria such as functional evaluations and convergence order in a way that minimizing functional evaluations one could achieve the maximal convergence order [14, 16]. For this purpose, the efficiency index is defined by $E(n,p) = p^{\frac{1}{n}}$, where p and n stand for convergence order and functional evaluations, respectively [14, 16]. It is worth noting that Kung and Traub conjectured that any n-step method without memory is optimal if it uses $n+1$ functional evaluations with convergence order 2^n [11]. According to this conjecture, there have been many attempts to construct such optimal methods without memory [1-20]. By methods without memory we mean those methods that only use available information of functional evaluations in the current iteration [16]. The convergence orders of the optimal n-point methods without memory cannot exceed 2^n using $n+1$ functional evaluations. However, it is still possible to increase the convergence order if the idea of introducing self-accelerator or with memory is applied. While methods without memory use information of the current iteration, methods with memory focus on using information not only from the current iteration, but also from the information of the previous iterations. Hence, they are of better efficiency. This advantage of method with memory motivated many researchers to pay attention to study such methods successfully [14]. Indeed, to the best of our knowledge, many researchers have developed very efficient methods with memory that use the information of the only last two iterations. However,it is, to our belief, still possible to use the available information of functional evaluations to increase the convergence order as well as efficiency index of a method without memory from all previous iterations and not only the last two iterations. This matter has not been considered in literature, so to fill this gap, in this work, our prime aim is to develop an adaptive method with memory,i.e., method that reuses the information from all the previous iterations and not only the last two iterations. Indeed, we try to extend the optimal two-step method without memory developed by Kung and Traub [11] to adaptive method with memory. This idea enables us to reuse all the previous information, and the efficiency index

of the optimal method without memory is highly increased. This work is organised as follows: Section 2 studies the idea of developing the adaptive method with memory extended from Kung and Traub optimal two-step method without memory. Also, it is attempted to discuss some special cases that have practical uses. This section includes convergence analysis,too. Section 3 is devoted to numerical implementations. We disclose the mathematica code for a special case which can easily be modified for other cases. Also, some other numerical aspects such as studying the basin of attraction are carried out. Section 4 will conclude this work.

2. Developing an Adaptive Kung and Traub's Method with Memory

This section deals with developing a new kind of methods with memory. Taken from literature, adaptive methods with memory has not been investigated yet,so this subject came into our attraction here. By the terminology "adaptive method with memory" we mean to reuse all the available data to increase the convergence order of a method without memory. We emphasize that developed methods with memory in the literature use only the information of the two last iterations. However, all the available data are possible to reuse from the first iteration to the current one iteration. This is the main principle behind the construction of adaptive method with memory. For this purpose, we consider the first two steps of the derivative-free method introduced by Kung-Traub

$$\begin{cases} y_k = x_k - \frac{f(x_k)}{f[x_k,w_k]}, w_k = x_k + \gamma f(x_k) \\ x_{k+1} = y_k - \frac{f(w_k)}{f(w_k)-f(y_k)} \frac{f(y_k)}{f[x_k,y_k]}, \qquad k = 0,1,2,\cdots, \end{cases} \tag{1}$$

where x_0 and γ are given suitably. This is an optimal method without memory since it uses three functional evaluations per iteration and has convergence order four. We need to review its error equations:

$$e_{k,w} = (1+\gamma f'(\alpha))e_k + O(e_k^2), \tag{2}$$

$$e_{k,y} = c_2\big(1+\gamma f'(\alpha)\big)e_k^2 + O(e_k^3), \tag{3}$$

and

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

where $e_k = x_k - \alpha$, $e_{k,w} = w_k - \alpha$, $e_{k,y} = y_k - \alpha$ and $c_k = \frac{f^{(k)}(\alpha)}{k!f(\alpha)}$, $k = 1, 2, ...$

In what follows, it is attempted to describe the idea of developing adaptive method with memory extracted from the optimal method without memory(1). Let us look at the error equations (2), (3) and (4). It can be clearly seen that all of them include the crucial factor $1 + \gamma f'(\alpha)$. If $1+\gamma f'(\alpha) = 0$, the coefficient of e_k^4 in (4) vanishes, so we find it possible to increase the convergence order. On the other hand, α is unknown, so this is impossible. In other words, α can be approximated effectively as the iterations proceed by the generated sequence $\{x_n\}$. We give the complete details of this procedure here. We pointed out since α is unknown, it is impossible to compute $f'(\alpha)$. It is worth noting that even if we assume that α is known, computing $f'(\alpha)$ is not a good policy since it increases the functional evaluation; hence, the optimality does not hold. Instead, it is suggested that $f'(\alpha)$ is approximated by applying interpolation concept. To give the clear picture of what this means, we have to review it briefly. This makes easy our dissuasion of developing adaptive method with memory. So, we assume that we have the information of the current and the previous iteration, say $f(x_k)$, $f(x_{k-1})$,$f(w_{k-1})$ and $f(y_{k-1})$. Therefore, $f(t)$ can be interpolated as follows:

$$f(t) = N_3(t; x_k, y_{k-1}, w_{k-1}, x_{k-1}) + E(t, x_k, y_{k-1}, w_{k-1}, x_{k-1}), \tag{5}$$

where $N_3(t)$ and $E(t)$ stand for Newton's polynomial interpolation and its error,respectively. Hence

$$f'(t) = N_3'(t) + E'(t). \tag{6}$$

Consequently, to approximate $f'(\alpha)$ in $1 + \gamma f'(\alpha) = 0$, consider

$$\gamma = -\frac{1}{f'(\alpha)} \simeq -\frac{1}{N_3'(x_k)} = \gamma_k. \tag{7}$$

If we consider the approximate $\gamma_k = -\frac{1}{N_3'(x_k)}$ in each iteration in (1), then we have the following method with memory:

$$\begin{cases} \gamma_k = -\frac{1}{N_3'(x_k)} \\ w_k = x_k + \gamma_k f(x_k) \qquad k = 0, 1, 2, \cdots, \\ y_k = x_k - \frac{f(x_k)}{f[x_k, w_k]}, \\ x_{k+1} = y_k - \frac{f(w_k)}{f(w_k)-f(y_k)} \frac{f(y_k)}{f[x_k, y_k]}, \end{cases} \tag{8}$$

where γ_0 and x_0 are given suitably. Although the method (8) has been studied well [14], we revise its error analysis again with a different but simpler approach. We need to have the following:

Lemma 2.1. *If $\gamma_k = -1/N_3'(x_k)$, then*

$$1 + \gamma_k f'(\alpha) \sim e_{k-1} e_{k-1,w} e_{k-1,y}. \tag{9}$$

Proof. From (5), we have

$$\begin{aligned} N_3'(x_k) = f'(x_k) - \frac{f^{(4)}(\xi)}{4!}(x_k - x_{k-1})(x_k - y_{k-1})(x_k - w_{k-1}) \\ \sim f'(\alpha)(1 + c_4 e_{k-1} e_{k-1,w} e_{k-1,y}). \end{aligned} \tag{10}$$

Hence

$$1 + \gamma_k f'(\alpha) = 1 - \frac{f'(\alpha)}{N_3'(x_k)} \sim e_{k-1} e_{k-1,w} e_{k-1,y}. \quad \square \tag{11}$$

The next theorem addresses the convergence order of the method with memory (8)

Theorem 2.2. *If α is a simple zero of $f(x) = 0$, and x_0 is close enough to α, then the method (8) has convergence order six.*

Proof. Let $\{x_k\}$,$\{w_k\}$, and $\{y_k\}$ have convergence orders r,p and q, respectively. Then

$$e_{k+1} \sim e_k^r \sim (e_{k-1}^r)^r \sim e_{k-1}^{r^2}, \tag{12}$$

$$e_{k,w} \sim e_k^p \sim (e_{k-1}^p)^r \sim e_{k-1}^{pr}, \tag{13}$$

and

$$e_{k,y} \sim e_k^q \sim (e_{k-1}^q)^r \sim e_{k-1}^{qr}. \tag{14}$$

Also, by Lemma (2.1), we have

$$1 + \gamma_k f'(\alpha) \sim e_{k-1}^{p+q+1}. \tag{15}$$

Substituting (15) in relations (2),(3) and (4), we have

$$e_{k,w} \sim e_{k-1}^{1+p+q+r}, \tag{16}$$

$$e_{k,y} \sim e_{k-1}^{1+p+q+2r}, \tag{17}$$

and

$$e_{k+1} \sim e_{k-1}^{2+2p+2q+4r}. \tag{18}$$

To reach the conclusion,we match the right-hand side of (12)-(18), (13)-(16) and (14)-(17). Thus

$$\begin{cases} 2 + 2p + 2q + 4r = r^2 \\ 1 + p + q + r = pr \\ 1 + p + q + 2r = qr. \end{cases} \tag{19}$$

Solving this nonlinear of system gives $p = 2$, $q = 3$ and $r = 6$ □

Up to now, we have discussed reusing the available information from the last two iterations and the convergence order from four to six without any new computation of the functions. In other words, with the same complexity of the basic optimal method without memory (1), we are able to improve its efficiency index defined by $E(p, n) = p^{\frac{1}{n}}$ introduced by Ostrowski([16]). The efficiency index of method (1) is $E(1) = 4^{\frac{1}{3}} \simeq 1.5874$ while the efficiency index of method (8) is $E(8) = 6^{\frac{1}{3}} \simeq 1.81712$. Now,it is our aim to consider the available information in the last three iterations to approximate the self-accelerator γ_k.

To this end, we approximate $f(t)$ by Newton's interpolation going through the node $x_k, y_{k-1}, w_{k-1}, x_{k-1}, y_{k-2}, w_{k-2}, x_{k-2}$

$$f(t) = N_6(t) + E(t). \tag{20}$$

Hence

$$\gamma = -\frac{1}{f'(\alpha)} \simeq -\frac{1}{N_6'(x_k)} = \gamma_k. \tag{21}$$

We suggest the following new method with memory

$$\begin{cases} \gamma_k = -\frac{1}{N_6'(x_k)} \\ w_k = x_k + \gamma_k f(x_k) \\ y_k = x_k - \frac{f(x_k)}{f[x_k, w_k]}, \qquad k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(w_k)}{f(w_k) - f(y_k)} \frac{f(y_k)}{f[x_k, y_k]} \end{cases} \tag{22}$$

where γ_0 and x_0 are given suitably. To discuss its convergence analysis, we need to have the following:

Lemma 2.3. *If $\gamma_k = -1/N_6'(x_k)$, then*

$$\begin{aligned} 1 + \gamma_k f'(\alpha) &\sim e_{k-1} e_{k-2} e_{k-1,w} e_{k-2,w} e_{k-1,y} e_{k-2,y} \\ &= \prod_{s=1}^{2} e_{k-s} e_{k-s,w} e_{k-s,y}. \end{aligned} \tag{23}$$

Proof. By differentiating (20)and setting $t = x_k$, we have

$$\begin{aligned} N_6'(x_k) = f'(x_k) - \frac{f^{(7)}(\xi)}{7!} \prod_{s=1}^{2} (x_k - x_{k-s})(x_k - y_{k-s})(x_k - w_{k-s}) \\ \sim f'(\alpha)(1 + c_7 \prod_{s=1}^{2} e_{k-s} e_{k-s,w} e_{k-s,y}). \end{aligned} \tag{24}$$

Consequently,

$$\begin{aligned} 1 + \gamma_k f'(\alpha) = 1 - \frac{f'(\alpha)}{N_6'(x_k)} = 1 - \frac{1}{1 + c_7 \prod_{s=1}^{2} e_{k-s} e_{k-s,w} e_{k-s,y}} \\ \sim \prod_{s=1}^{2} e_{k-s} e_{k-s,w} e_{k-s,y}. \quad \square \end{aligned} \tag{25}$$

The following theorem addresses the convergence order of the new method with memory (22).

Theorem 2.4. *If α is a simple zero of $f(x) = 0$, and x_0 is close enough to α, then the sequence $\{x_n\}$ generated by method (22) has the convergence order 6.31.*

Proof. Let $\{x_k\}$,$\{w_k\}$ and $\{y_k\}$ have convergence orders r,p, and q, respectively, i.e.,

$$e_{k+1} \sim e_{k-2}^{r^3}, \tag{26}$$

$$e_{k-1,w} \sim e_{k-2}^{pr}, \tag{27}$$

and

$$e_{k-1,y} \sim e_{k-2}^{qr}. \tag{28}$$

By Lemma 2.3, we have

$$\begin{aligned} 1 + \gamma_k f'(\alpha) &\sim \prod_{s=1}^{2} e_{k-s} e_{k-s,w} e_{k-s,y} = e_{k-1} e_{k-2} e_{k-1,w} e_{k-2,w} e_{k-1,y} e_{k-2,y} \\ &= e_{k-2}^{r} e_{k-2} e_{k-2}^{pr} e_{k-2}^{p} e_{k-2}^{qr} e_{k-2}^{q} = e_{k-2}^{1+r} e_{k-2}^{p(1+r)} e_{k-2}^{q(1+r)} = e_{k-2}^{(1+p+q)(1+r)}, \end{aligned} \tag{29}$$

or

$$1 + \gamma_k f'(\alpha) \sim e_{k-2}^{(1+p+q)(1+r)}. \tag{30}$$

substituting (30) in relations (2),(3) and (4)gives

$$e_{k+1} \sim e_{k-2}^{2(p+q+1)(r+1)+4r^2}, \tag{31}$$

$$e_{k,w} \sim e_{k-2}^{(p+q+1)(r+1)+r^2}, \tag{32}$$

and

$$e_{k,y} \sim e_{k-2}^{(1+p+q)(1+r)+2r^2}. \tag{33}$$

To obtain the desired result, it is enough to match the right-hand side of (26)-(31), (27)-(32) and (28)-(33), giving

$$\begin{cases} (p+q+1)(r+1) + r^2 = pr^2 \\ (p+q+1)(r+1) + 2r^2 = qr^2 \\ 2(p+q+1)(r+1) + 4r^2 = r^3. \end{cases} \tag{34}$$

Solving this nonlinear system with the command `Solve` in Mathematica software gives the solution $p = \frac{1}{2}(1+\sqrt{11})$, $q = \frac{1}{2}(3+\sqrt{11})$, and $r = 3+\sqrt{11} \simeq 6.31$ $\square$

Methods with memory (8) and(22) use the available information of the last two and three iterations, respectively. We recall method (8) has been studied in [14], while method (22) is new. Now, it is attempted to discus the general case, i.e., adaptive method with memory in which it uses all the available information from the first iteration to the current iteration. We suppose that we are at the iteration k. In each iteration three functional evaluations are at hand; therefore, we have computed the function $f(t)$ at the following points $\bigcup_{s=0}^{k-1}\{x_s, w_s, y_s\}\bigcup\{x_k\}$. We have $3k+1$ points, so the best Newton's interpolation which goes through them has degree $3k$. As with the two methods (8) and(22), now, we can apply the following Newton's interpolation polynomial for approximation of the accelerator γ_k,

$$f(t)-N_{3k}(t) = E_{3k+1}(t), \quad N_{3k}(t) = N_{3k}(t; x_k, y_{k-1}, w_{k-1}, x_{k-1}, ..., y_0, w_0, x_0). \tag{35}$$

If we use $\gamma_k = -\frac{1}{N'_{3k}(x_k)}$, then the following adaptive method with memory is proposed

$$\begin{cases} \gamma_k = -\frac{1}{N'_{3k}(x_k)} \\ w_k = x_k + \gamma_k f(x_k), \\ y_k = x_k - \frac{f(x_k)}{f[x_k, w_k]}, \qquad k = 0, 1, 2, \cdots, \\ x_{k+1} = y_k - \frac{f(w_k)}{f(w_k)-f(y_k)} \frac{f(y_k)}{f[x_k, y_k]}, \end{cases} \tag{36}$$

where x_0 and γ_0 are given suitably.

To discuss the error analysis of this method, we need the following lemma.

Lemma 2.5. *If $\gamma_k = -1/N'_{3k}(x_k)$, then*

$$1 + \gamma_k f'(\alpha) \sim \prod_{s=1}^{k} e_{k-s} e_{k-s,w} e_{k-s,y}. \tag{37}$$

Proof. By differentiating (35) and setting $t = x_k$, we have

$$N'_{3k}(x_k) = f'(x_k) - \frac{f^{(3k+1)}(\xi)}{(3k+1)!}\prod_{s=1}^{k}(x_k - x_{k-s})(x_k - y_{k-s})(x_k - w_{k-s}) \sim f'(\alpha)(1 + c_{3k-1}\prod_{s=1}^{2} e_{k-s}e_{k-s,w}e_{k-s,y}). \tag{38}$$

Therefore,

$$1 + \gamma_k f'(\alpha) = 1 - \frac{f'(\alpha)}{N'_{3k}(x_k)} = 1 - \frac{1}{1 + c_{3k+1}\prod_{s=1}^{k} e_{k-s}e_{k-s,w}e_{k-s,y}} \sim \prod_{s=1}^{k} e_{k-s}e_{k-s,w}e_{k-s,y}. \tag{39}$$

This completes the proof. □

Now, we deal with the convergence analysis of the adaptive method with memory (36).

Theorem 2.6. *If α is a simple zero of $f(x) = 0$, and x_0 is close enough to α, then the convergence order of the method ((36)) is given by*

$$\begin{cases} (p+q+1)\sum_{s=0}^{k-1} r^s + r^k = r^k p, \\ (p+q+1)\sum_{s=0}^{k-1} r^s + 2r^k = r^k q, \\ (p+q+1)\sum_{s=0}^{k-1} r^s + 4r^k = r^{k+1}, \end{cases} \tag{40}$$

where r,p and q are the convergence orders of the sequences $\{x_k\}$,$\{w_k\}$, and $\{y_k\}$, respectively.

Proof. We have

$$e_1 \sim e_0^r, e_2 \sim e_0^{r^2}, ..., e_{k+1} \sim e_0^{r^{k+1}}, \tag{41}$$

$$e_{0,w} \sim e_0^p, e_{1,w} \sim e_0^{rp}, ..., e_{k,w} \sim e_0^{r^k p}, \tag{42}$$

and

$$e_{0,y} \sim e_0^q, e_{1,y} \sim e_0^{rq}, ..., e_{k,y} \sim e_0^{r^k q}. \tag{43}$$

By applying relations (41)-(42)to Lemma **??**, we have

$$\begin{aligned} 1 + \gamma_k f'(\alpha) \sim e_0^{r^{k-1}} ... e_0^r e_0 e_0^{r^{k-1}p} ... e_0^{rp} e_0^p e_0^{r^{k-1}q} ... e_0^{rq} e_0^q \\ \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s}. \end{aligned} \tag{44}$$

Now, considering this result in the error equations (2)-(4)), we have

$$\begin{aligned} e_{k,w} \sim (1 + \gamma_k f'(\alpha)) e_k \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s} e_0^{r^k} \\ \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s + r^k}, \end{aligned} \tag{45}$$

$$\begin{aligned} e_{k,y} \sim (1 + \gamma_k f'(\alpha)) e_k^2 \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s} e_0^{2r^k} \\ \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s + 2r^k}, \end{aligned} \tag{46}$$

and

$$\begin{aligned} e_{k+1} \sim (1 + \gamma_k f'(\alpha))^2 e_k^4 \sim e_0^{2(p+q+1)\sum_{s=0}^{k-1} r^s} e_0^{4r^k} \\ \sim e_0^{2(p+q+1)\sum_{s=0}^{k-1} r^s + 4r^k}. \end{aligned} \tag{47}$$

Now, to achieve the devised result it is enough to match the right-hand side of (41)-(47),(42)-(46) and (43)-(45), respectively. Then

$$\begin{cases} e_0^{r^k p} \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s + r^k} \\ e_0^{r^k q} \sim e_0^{(p+q+1)\sum_{s=0}^{k-1} r^s + 2r^k} \\ e_0^{r^{k+1}} \sim e_0^{2(p+q+1)\sum_{s=0}^{k-1} r^s + 4r^k}. \end{cases} \tag{48}$$

Hence

$$\begin{cases} (p+q+1)\sum_{s=0}^{k-1} r^s + r^k = r^k p \\ (p+q+1)\sum_{s=0}^{k-1} r^s + 2r^k = r^k q \\ 2(p+q+1)\sum_{s=0}^{k-1} r^s + 4r^k = r^{k+1}. \end{cases} \tag{49}$$

This completes the proof. □

3. Numerical Test Problems and Discussions

In this section, we try to implement the adaptive method with memory (23) to show its practical feature. To this end, among many test problems, three test functions are considered and reported. Also, we disclose the mathematica code for this section. Table 1 shows the test functions, and Table 2 presents the numerical results. The basic optimal method without memory (1) has the convergence order four. However, its extension adaptive method with memory (36) has convergence orders 6, 6.31, and 6.36 for k = 1, 2, 3, respectively. Indeed, let us look at them closely. As can be seen in the first row of Table 1, the error exponents are decreasing according to the structure of the adaptive method with memory (36). In its first iteration, the error exponent is -6, corresponding to $k = 1$, while for the second iteration it is -39. The last column shows the approximate COC which is in accordance with the claimed goal in Theorem 2.6. Similarly, the next row of the reported result is in accordance with the given theory in Theorem 2.6. The last example shows that it is possible that for some test functions, the desired results are not supported. However, it produces an acceptable COC, say 6.2. It should be noted this issue is probably solved if we change the initial data. Indeed, in implementation, to achieve a good performance, one should pay enough attention to other aspects of the provided algorithms, not only to its direct running.

Table 1: Test functions for $\gamma = 0.1, \lambda = 0.1$

Example	x_0	α
$f_1(t) = e^{(t^3-t)} - \cos(t^2-1) + t^3 + 1$	-1.00	-1.65
$f_2(t) = e^t \sin(t) + \log(t^4 - 3t + 1)$	0.00	0.30
$f_3(t) = \frac{1}{t^4} - t^2 - \frac{1}{t} + 1$	1.00	2.00

Table 2: Results of (36) for different test functions

Funs.	$\lvert x_1 - \alpha\rvert$	$\lvert x_2 - \alpha\rvert$	$\lvert x_3 - \alpha\rvert$	COC
f_1	0.6588(-1)	0.4012(-6)	0.4181(-39)	6.3239
f_2	0.1157(-1)	0.1492(-8)	0.8962(-53)	6.4185
f_3	0.9741(-5)	0.1095(-24)	0.9272(-149)	6.2195

We have drawn the basins of attractions for both methods without and with memory (1) and (36) in Figures 1, 2, and 3. The aim of these drawings is to determine which one of the mentioned methods is better from the point of numerical stability. The test function is $p(z) = z^3 - 1$. As can be seen, at first glance it seems the optimal method without memory (1) is more stable than the adaptive method with memory (36). Indeed, Figure 1 corresponds to the initial fixed parameter $\gamma = -0.01$ while Figure 2 corresponds to the approximate $\gamma_3 = -0.3332$ computed by (21). If we change the fixed parameter of the optimal method without memory (1), say $\gamma = 1$, Figure 3 is generated which is lousy. To conclude, the stability behaviour of the optimal method without memory (1) varies as its parameter varies. For some parameters it is more stable than the adaptive method with memory (36) and for some it is not. Therefore, we cannot say which method is generally more stable. To study some other aspects of the stability, one can consult [1, 6].

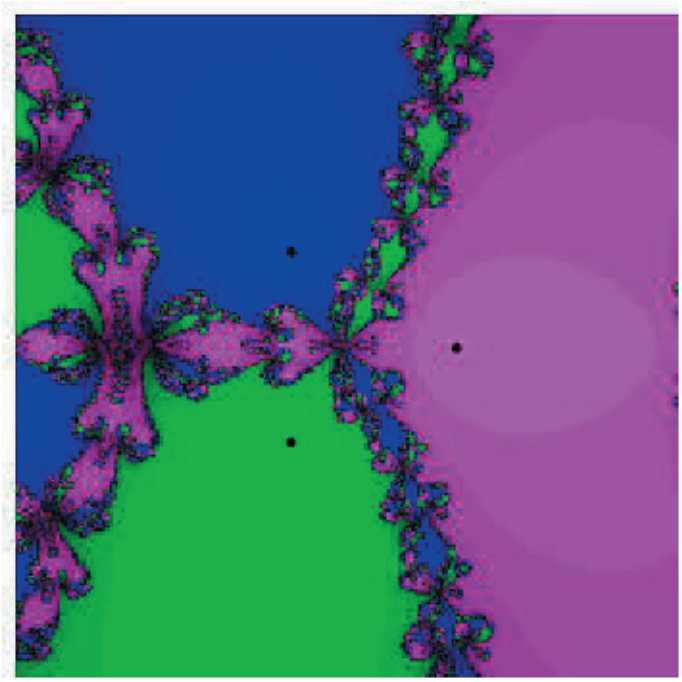

Figure 1. Dynamical Planes for (1) for $\gamma = -0.1$

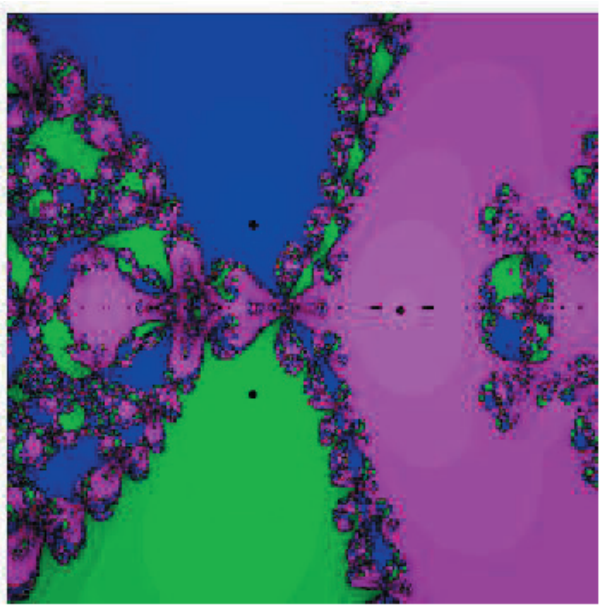

Figure 2. Dynamical Planes for (36) for $\gamma = -0.3332$

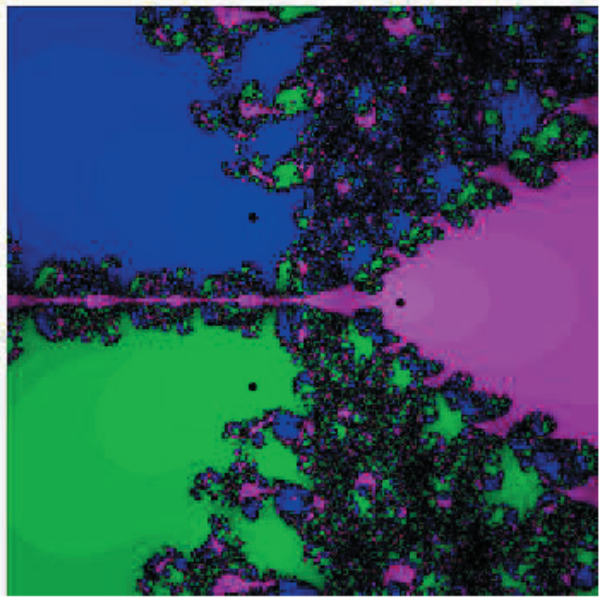

Figure 3. Dynamical Planes for (1) for $\gamma = 1$

4. Conclusion

In this work, we have tried to develop an adaptive method with memory for the first time. Compared to the previous methods with memory, this method uses all the available information from the first iteration to the current iteration, so it increases the efficiency index as high as possible. It should be noted that we considered only one accelerator, while the previous methods in literature have used one, two, or three accelerators in which increasing such accelerators may result in numerical instability and complexity. We emphasise that though it is possible to reuse the all the information from the first iteration to the current iterations at least theoretically, in the case study in this work, using the information of the last three iteration has the high impact and increases the efficiency index well enough. Also, we noted that we could not reach a conclusion about numerical stability of the proposed method with memory as apposed to its optimal method without memory from the basins of attraction point.

References

[1] S. Amat, S. Busquier, C. Bermudez, and A. A. Magrenan, On the election of the damped parameter of a two-step relaxed Newton-type method, *Nonlinear Dynamics,* 84 (1) (2016), 9-18.

[2] S. Amat, S. Busquier, and J. M. Gutierrez, On the local convergence of secant-type methods, *International Journal of Computer Mathematics,* 81 (2004), 1153-1161.

[3] I. K. Argyros, M. Kansal, V. Kanwar, and S. Bajaj, Higher-order derivativefree families of Chebyshev-Halley type methods with or without memory for solving nonlinear equations, *Applied Mathematics and Computation,* 315 (15) (2017), 224-242.

[4] I. K. Argyros, S. George, and A. A. Magrenan, Local convergence for multipoint-parametric Chebyshev-Halley-type methods of high convergence order, *Journal of Computational and Applied Mathematics,* 282 (2015), 215-224.

[5] I. K. Argyros and H. Ren, Efficient Steffensen-type algorithms for solving nonlinear equations, *International Journal of Computer Mathematics,* 90 (2013), 691-704.

[6] P. Bakhtiari, A. Cordero, T. Lotfi, K. Mahdiani, and J. R. Torregrosa, Widening basins of attraction of optimal iterative methods, *Nonlinear Dynamics,* 87 (2) (2017), 913-938.

[7] S. Aridiello, F. Chicharro, A. Cordero, and J. Torregrosa, Local convergence and dynamical analysis of a new family of optimal fourth-order iterative methods, *International Journal of Computer Mathematics,* 90 (2013), 2049-2060.

[8] S. Artidiello, A. Cordero, J. Torregrosa, and M. Vassileva, Two weighted eight-order classes of iterative root-finding methods, *International Journal of Computer Mathematics,* 92 (2015), 1790-1805.

[9] A. Cordero, T. Lotfi, P. Bakhtiari, and J. R. Torregrosa, An efficient twoparametric family with memory for nonlinear equations, *Numerical Algorithms,* 68 (2015), 323-335.

[10] Y. H. Geum, Y. I. Kim, and B. Neta, On developing a higher-order family of double-Newton methods with a bivariate weighting function, *Appl. Math. Comput.,* 254 (2015), 277-290.

[11] H. T. Kung and J. F. Traub , Optimal order of one-point and multipoint iteration, *J. ACM.,* 21 (1974), 643-651.

[12] T. Lotfi, P. Bakhtiari, A. Cordero, K. Mahdiani, and J. R. Torregrosa, Some new efficient multipoint iterative methods for solving nonlinear systems of equations, *International Journal of Computer Mathematics,* 92 (2015), 1921-1930.

[13] T. Lotfi, F. Soleymany, M. Ghorbanzadeh, and P. Assari, On the construction of some tri-parametric iterative methods with memory, *Numerical Algorithms,* 70 (2015), 835-845.

[14] M. S. Petkovic, B. Neta, L. D. Petkovic, and J. Dzunic, Multipoint methods for solving nonlinear equations: a survey, *Appl. Math. Comput.,* 226 (2014), 635-660.

[15] M. Salimi, T. Lotfi, S. Sharifi, and S. Siegmund, Optimal NewtonSecant like methods without memory for solving nonlinear equations with its dynamics, *International Journal of Computer Mathematics,* 94 (2017), 1759-1777.

[16] J. F. Traub, *Iterative Methods for the Solution of Equations,* Prentice Hall, New York, 1964.

[17] H. Veiseh, T. Lotfi, and T. Allahviranloo, *A study on the local convergence and dynamics of the two-step and derivative-free KungTraubs method,* (2017). https://doi.org/10.1007/s40314-017-0458-5.

[18] X. Wang, T. Zhang, and Y. Qin, Efficient two-step derivative-free iterative methods with memory and their dynamics, *International Journal of Computer Mathematics,* 93 (2016), 1423-1446.

[19] M. Zaka Ullah, S. Kosari, F. Soleymani, F. Khaksar Haghani, and A. S. Al-Fhaid, A super-fast tri-parametric iterative method with memory, *Applied Mathematics and Computation,* 289 (2016), 486-491.

[20] S. Qasim, Z. Ali, F. Ahmad, S. Serra-Capizzano, M. Z. Ullah, and A. Mahmood, solving systems of nonlinear equations when the nonlinearity is expensive, *Comput. Math. Appl.,* 71 (2016), 1464-1478.

[21] A. R. Sohaila and F. Soleymani, Iterative methods for nonlinear systems associated with finite difference approach in stochastic differential equations, *Numer. Algorithms,* 71 (2016), 89-102.

[22] F. Soleymani, D. K. R. Babajee, S. Shateyi, and S. S. Motsa, Construction of optimal derivative-free techniques without memory, *J. Appl. Math.,* 2012 (2012), 24. Art.ID497023.

[23] J. F. Stephenson, Remarks on iteration, *Skand. Aktuarietidskr,* 16 (1933), 64-72.

[24] J. F. Traub, *Iterative Methods for the Solution of Equations*, Prentice Hall, New York, 1964.

[25] S. Wagon, *Mathematica in Action, third ed., Springer,* New York, NY, USA, 2010.

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

[27] J. Duni and M. S. Petkovi, A Cubically Convergent Steffensen-Like Method for Solving Nonlinear Equations, *Applied Mathematics Letters,* 25 (2012), 1881-1886.

[28] J. Dunic, On efficient two-parameter methods for solving nonlinear equations, *Numer Algor,* 63 (2013), 549-569. DOI 10.1007/s11075-012-9641-3

Mohammad Javad Lalehchini
Ph.D Student of Mathematics
Department of Applied Mathematics
Hamedan Branch, Islamic Azad University
Hamedan, Iran
E-mail: mj lalehchini@yahoo.com
E-mail: lalehchini@iauh.ac.ir

Taher Lotfi
Associate Professor of Mathematics
Department of Applied Mathematics
Hamedan Branch, Islamic Azad University
Hamedan, Iran
E-mail: lotfitaher@yahoo.com
E-mail: lotfi@iauh.ac.ir

Katayoun Mahdiani
Associate Professor of Mathematics
Department of Applied Mathematics
Hamedan Branch, Islamic Azad University
Hamedan, Iran
E-mail: mahdiani@iauh.ac.ir