2016

An analysis of a Khattri's 4th order family of methods

Neta, Beny

Elsevier Inc.

Applied Mathematics and Computation 279 (2016) 198–207
http://hdl.handle.net/10945/50403

Applied Mathematics and Computation 279 (2016) 198–207

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

An analysis of a Khattri's 4th order family of methods

Changbum Chun[a,1], Beny Neta[b,*]

[a] Department of Mathematics, Sungkyunkwan University, Suwon 440-746, Republic of Korea
[b] Naval Postgraduate School, Department of Applied Mathematics, Monterey CA, 93943, United States

ARTICLE INFO

Keywords:
Iterative methods
Order of convergence
Basin of attraction
Extraneous fixed points

ABSTRACT

In this paper we analyze an optimal fourth-order family of methods suggested by Khattri and Babajee, (2013). We analyze the family using the information on the extraneous fixed points. Two measures of closeness to the imaginary axis of the set of extraneous points are considered and applied to the members of the family to find its best performer. The results are compared to three best members of King's family of methods.

Published by Elsevier Inc.

1. Introduction

"Calculating zeros of a scalar function f ranks among the most significant problems in the theory and practice not only of applied mathematics, but also of many branches of engineering sciences, physics, computer science, finance, to mention only some fields" [2]. For example, to minimize a function $F(x)$ one has to find the points where the derivative vanishes, i.e. $F'(x) = 0$. There are many algorithms for the solution of nonlinear equations, see e.g. Traub [3], Neta [4] and the recent book by Petković et al. [2]. The methods can be classified as one step and multistep. One step methods are of the form

$$x_{n+1} = \phi(x_n).$$

The iteration function ϕ depends on the method used. For example, Newton's method is given by

$$x_{n+1} = \phi(x_n) = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{1}$$

Some one point methods allow the use of one or more previously found points, in such cases we have a one step method with memory. For example, the secant method uses one previous point and is given by

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n).$$

In order to increase the order of a one step method, one requires higher derivatives. For example, Halley's method is of third order and uses a second derivative [5]. In many cases the function is not smooth enough or the higher derivatives are too complicated. Another way to increase the order is by using multistep. The recent book by Petković et al. [2] is dedicated to multistep methods. A trivial example of a multistep method is a combination of two Newton steps, i.e.

$$y_n = x_n - \frac{f(x_n)}{f'(x_n)}, \tag{2}$$

* Corresponding author. Tel.: +1 831 656 2235; fax: +1 831 656 2355.
E-mail addresses: cbchun@skku.edu (C. Chun), bneta@nps.edu, byneta@gmail.com (B. Neta).
[1] Tel.: +82 31 299 4523; fax: +82 31 290 7033.

http://dx.doi.org/10.1016/j.amc.2016.01.025
0096-3003/Published by Elsevier Inc.

$$x_{n+1} = y_n - \frac{f(y_n)}{f'(y_n)}. \tag{2}$$

Of course this is too expensive. The cost of a method is defined by the number (ℓ) of function-evaluations per step. The method (2) requires four function-evaluations (including derivatives). The efficiency of a method is defined by

$$I = p^{1/\ell},$$

where p is the order of the method. Clearly one strives to find the most efficient methods. To this end, Kung and Traub [6] introduced the idea of optimality. They conjectured that a method using ℓ evaluations is optimal if the order is $2^{\ell-1}$. This conjecture was proved by Woźniakowski [7] in the case of Hermitian information. Kung and Traub have developed optimal multistep methods of increasing order. Newton's method (1) is optimal of order 2. King [8] has developed an optimal fourth order family of methods depending on a parameter β

$$\begin{aligned} y_n &= x_n - \frac{f(x_n)}{f'(x_n)}, \\ x_{n+1} &= y_n - \frac{f(y_n)}{f'(x_n)} \frac{f(x_n) + \beta f(y_n)}{f_n + (\beta - 2) f(y_n)}, \end{aligned} \tag{3}$$

Neta [9] has developed a family of sixth order methods based on King's method (3). Also Neta [10] has developed optimal eighth and sixteenth order methods. Wang and Liu [11] and Thukral and Petković [12] have developed optimal eighth order methods. Khattri and Babajee [1] has developed the following optimal fourth order 3 parameter family of methods

$$\begin{aligned} y_n &= x_n - \frac{f(x_n)}{f'(x_n) + \frac{\alpha\beta}{2} f(x_n)^m}, \\ x_{n+1} &= y_n - \frac{f(x_n) f(y_n)}{f(x_n) - 2f(y_n)} \left[\frac{\alpha}{f'(x_n) + \beta f(x_n)^m} - \frac{\alpha - 1}{f'(x_n) + \eta f(y_n)} \right]. \end{aligned} \tag{4}$$

There are a number of ways to compare various techniques proposed for solving nonlinear equations. Comparisons of the various algorithms are based on the number of iterations required for convergence, number of function evaluations, and/or amount of CPU time. "The primary flaw in this type of comparison is that the starting point, although it may have been chosen at random, represents only one of an infinite number of other choices" [13]. In recent years the Basin of Attraction method was introduced to visually comprehend how an algorithm behaves as a function of the various starting points. The first comparative study using basin of attraction, to the best of our knowledge, is by Vrscay and Gilbert [14]. They analyzed Schröder and König rational iteration functions. Other work was done by Stewart [15], Kalantari and Jin [16], Amat et al. [17–20], Chicharro et al. [21], Chun et al. [22,23], Cordero et al. [24], Neta et al. [25–27], Magreñán [28], Magreñán et al. [29], and Scott et al. [13]. There are also similar results for methods to find roots with multiplicity, see e.g. [30,31] and [32].

In this paper we analyze a family of optimal fourth order methods (4). We will examine the family and show how to choose the parameters involved in the family similar to Chun et al. [33].

2. Extraneous fixed points

In solving a nonlinear equation iteratively we are looking for fixed points which are zeros of the given nonlinear function. Many multipoint iterative methods have fixed points that are not zeros of the function of interest. Thus, it is necessary to investigate the number of extraneous fixed points, their location and their properties. In order to find the extraneous fixed points, we rewrite the family of methods in the form

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} H_f(x_n, y_n), \tag{5}$$

where the function H_f for method (4) is given by

$$H_f(x_n, y_n) = \frac{f'(x_n)}{f'(x_n) + \frac{\alpha\beta}{2} f(x_n)^m} + \frac{f'(x_n) f(y_n)}{f(x_n) - 2f(y_n)} \left[\frac{\alpha}{f'(x_n) + \beta f(x_n)^m} - \frac{\alpha - 1}{f'(x_n) + \eta f(y_n)} \right]. \tag{6}$$

Clearly, if x_n is the root then from (5) we have $x_{n+1} = x_n$ and the iterative process converged. But we can have $x_{n+1} = x_n$ even if x_n is not the root but $H_f(x_n, y_n) = 0$. Those latter points are called extraneous fixed points. It is best to have the extraneous fixed points on the imaginary axis or close to it. For example, in the case of King's method (3) we found that the best performance is when the parameter $\beta = 3 - 2\sqrt{2}$ or $\beta = 0$ since then the extraneous fixed points are closest to the imaginary axis.

We have searched the parameter space (α, β, η) in the case of $m = 1$ and found that the extraneous fixed points are not on the imaginary axis except in the case that any two of the parameters are zero, which is Ostrwoski's fourth order method [3]. As it can be seen in the next section, the cases of m greater than 1 (i.e. methods KB1 and KB2) gave worse performance than $m = 1$. We have tried to get several measures of closeness to the imaginary axis and experimented with those members from the parameter space.

Table 1
The eleven cases for experimentation.

Case	Method	α	β	η	m
1	KB1	6	−1	2.5	2
2	KB2	5.5	1	2.5	2
3	KB3	2	1	2.5	1
4	KB4	−2	0	0.01	1
5	KB5	2.1	−0.7	−1.1	1
6	KB6	−2	0	0.1	1
7	KB7	−2	0	0.001	1
8	KB8	−2	0	−4	1
9	King0	–	–	–	–
10	King	–	–	–	–
11	King01	–	–	–	–

Let $E = \{z_1, z_2, \ldots, z_{n_{\alpha,\beta,\eta}}\}$ be the set of the extraneous fixed points corresponding to the values given to α, β, and η. We define

$$d(\alpha, \beta, \eta) = \max_{z_i \in E} |Re(z_i)|. \tag{7}$$

To choose the parameters α, β, and η we set $m = 1$. The minimum of $d(\alpha, \beta, \eta)$ occurs at $\alpha = -2, \beta = 0$ and $\eta = 0.1$ for the grid spacing of 0.1 in the α, β and η directions. We observed that the minimum of $d(\alpha, \beta, \eta)$ occurs also at $\alpha = -2, \beta = 0$ when we decrease the value of η from 0.1 to 0.01. To further look for where the minimum of d occurs for the grid spacing of 0.001 in the α, β and η directions, we set at $\alpha = -2, \beta = 0$ and found that it occurs at $\eta = 0.001$

Another method to choose the parameters is by considering the stability of $z \in E$ defined by

$$dq(z) = \frac{dq}{dz}(z), \tag{8}$$

where q is the iteration function of (5). We define a function, $A(\alpha, \beta, \eta)$, the averaged stability value of the set E by

$$A(\alpha, \beta, \eta) = \frac{\sum_{z_i \in E} |dq(z_i)|}{n_{\alpha,\beta,\eta}}. \tag{9}$$

The smaller A becomes, the less chaotic the basin of attraction tends to. We also set $m = 1$ to choose the parameters. The minimum of $A(\alpha, \beta, \eta)$ occurs at $\alpha = 2.1, \beta = -0.7$ and $\eta = -1.1$ for the grid spacing $dx = 0.1$. To choose another parameters set for the grid spacing $dx = 0.001$, we set $\alpha = -2, \beta = 0$, and found that the minimum of A occurs at $\eta = -4$.

In the next section we plot the basins of attraction for these cases along with the basins for several members of King's family of methods and the cases presented in [1] to find the best performer.

3. Numerical experiments

In this section, we give the results of using the 11 cases described in Table 1 on six different polynomial equations.

Khattri et al. [1] suggested three cases (i) $\alpha = 6, \beta = -1, \eta = 2.5, m = 2$, (ii) $\alpha = 5.5, \beta = 1, \eta = 2.5, m = 2$, and (iii) $\alpha = 2, \beta = 1, \eta = 2.5, m = 1$ of their proposed family. Here, we consider these cases and call them KB1, KB2, and KB3, respectively. We also compare the results to 3 other members of King's fourth-order family (3). In King's family (3) we have chosen the parameters $\beta = 3 - 2\sqrt{2}$ as suggested by the analysis in [26], $\beta = 0$ and $\beta = \frac{1}{10}$. We call them King, King0, and King01, respectively.

We have ran our code for each case and each example on a 6 by 6 square centered at the origin. We have taken 360,000 equally spaced points in the square as initial points for the algorithms. We have recorded the root the method converged to and the number of iterations it took. We chose a color for each root and the intensity of the color gives information on the number of iterations. The slower the convergence the darker the shade. If the scheme did not converge in 40 iterations to one of the roots, we color the point black.

Example 1. In the first case we have taken the cubic polynomial

$$p_1(z) = z^3 + 4z^2 - 10 \tag{10}$$

Clearly, one root is real (1.365230013) and the other two are complex conjugate. The basins are plotted in Fig. 1. In the top row we have KB1 (left) and KB2 (right). Clearly these two methods have many initial points in the square leading to a non converging sequence within 40 iterations. In the second row we have KB3 (left), KB4 (center) and KB5 (right). In the third row we view KB6 (left), KB7 (center) and KB8 (right) and the bottom row shows King method with $\beta = 0$ (left), $\beta = 3 - 2\sqrt{2}$ (center) and $\beta = 0.1$ (right). It is clear that the only ones not having black points are KB4, KB6, KB7 and King with $\beta = 3 - 2\sqrt{2}$. The worst are KB1, KB2, KB8 and KB3. In order to have a more quantitative comparison, we have listed the average number of iterations per point for each method and each example in Table 2 and the standard deviation in Table 3.

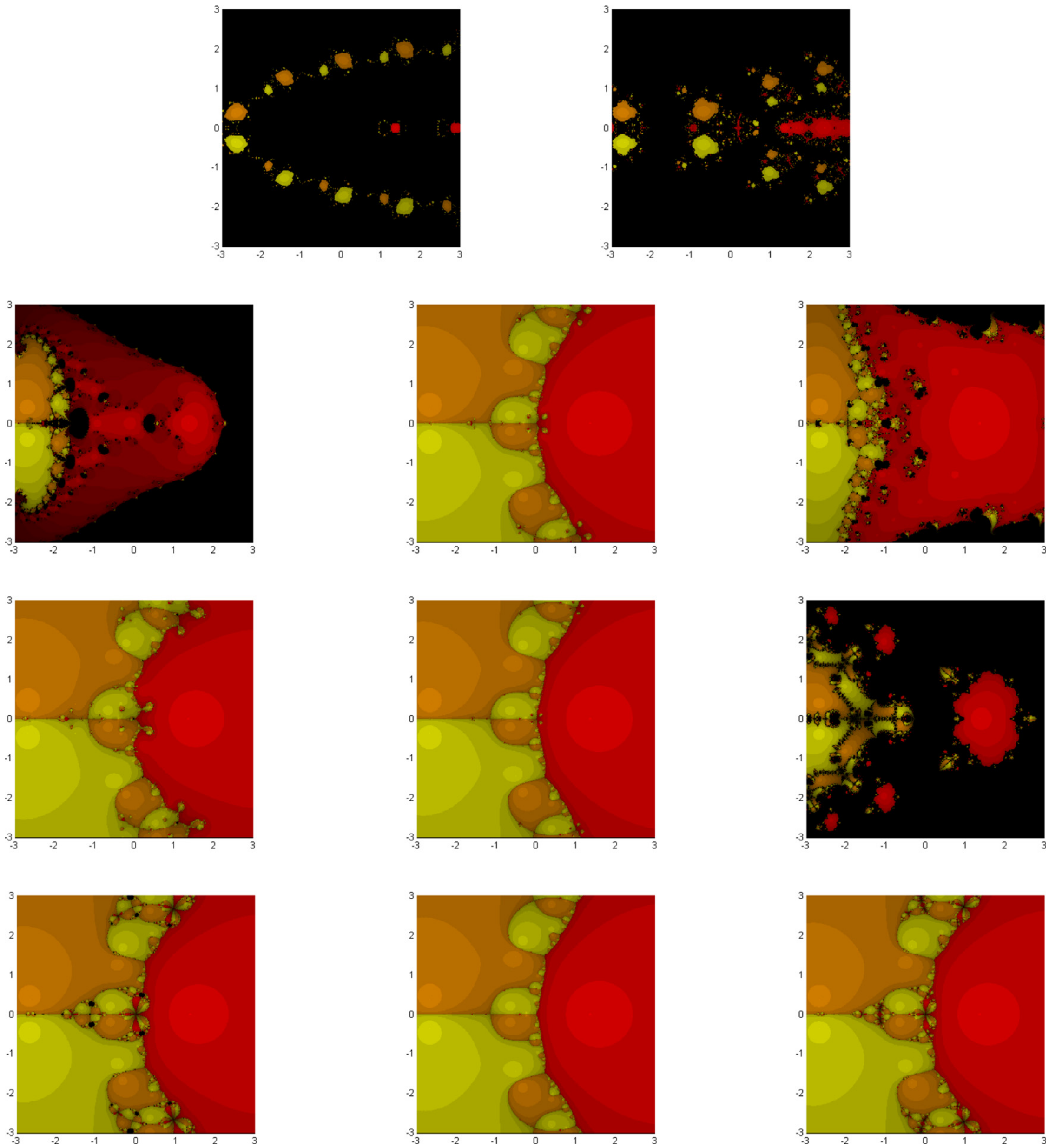

Fig. 1. Top row for KB1 (left) and KB2 (right), second row for KB3 (left), KB4 (center), and KB5 (right), third row for KB6 (left), KB7 (center) and KB8 (right), bottom row for King with $\beta = 3 - 2\sqrt{2}$ (left), $\beta = 0$ (center) and $\beta = 0.1$ (right) for the roots of the polynomial $z^3 + 4z^2 - 10$.

Consulting these tables for the first example, we note that King with $\beta = 3 - 2\sqrt{2}$, KB7 and KB4 are requiring about the same number followed by KB6. The worst are KB1, KB2 and KB8, followed by KB3. Another measure for comparison is the CPU time to run the method on all 360,000 points. This is listed in Table 4 for a Samsung Premium Ultrabook NT900X4C. Now it shows that King with $\beta = 3 - 2\sqrt{2}$ and $\beta = 0.1$ are the fastest followed by King with $\beta = 0$, KB7 and KB6.

Example 2. In the second example we have taken a quintic polynomial with real simple roots

$$p_2(z) = z^5 - 5z^3 + 4z. \tag{11}$$

The results are plotted in Fig. 2. The order of the subplots is as before. Again the worst are KB1 and KB2. Therefore these methods will not be shown in the rest of the examples. The best methods seem to be King with $\beta = 3 - 2\sqrt{2}$, followed by KB7, KB4 and KB6. The results in Tables 2 and 3 confirm these qualitative conclusions. The CPU time for King method for any value of β we have was the lowest, followed by KB4, KB7 and KB6 in that order.

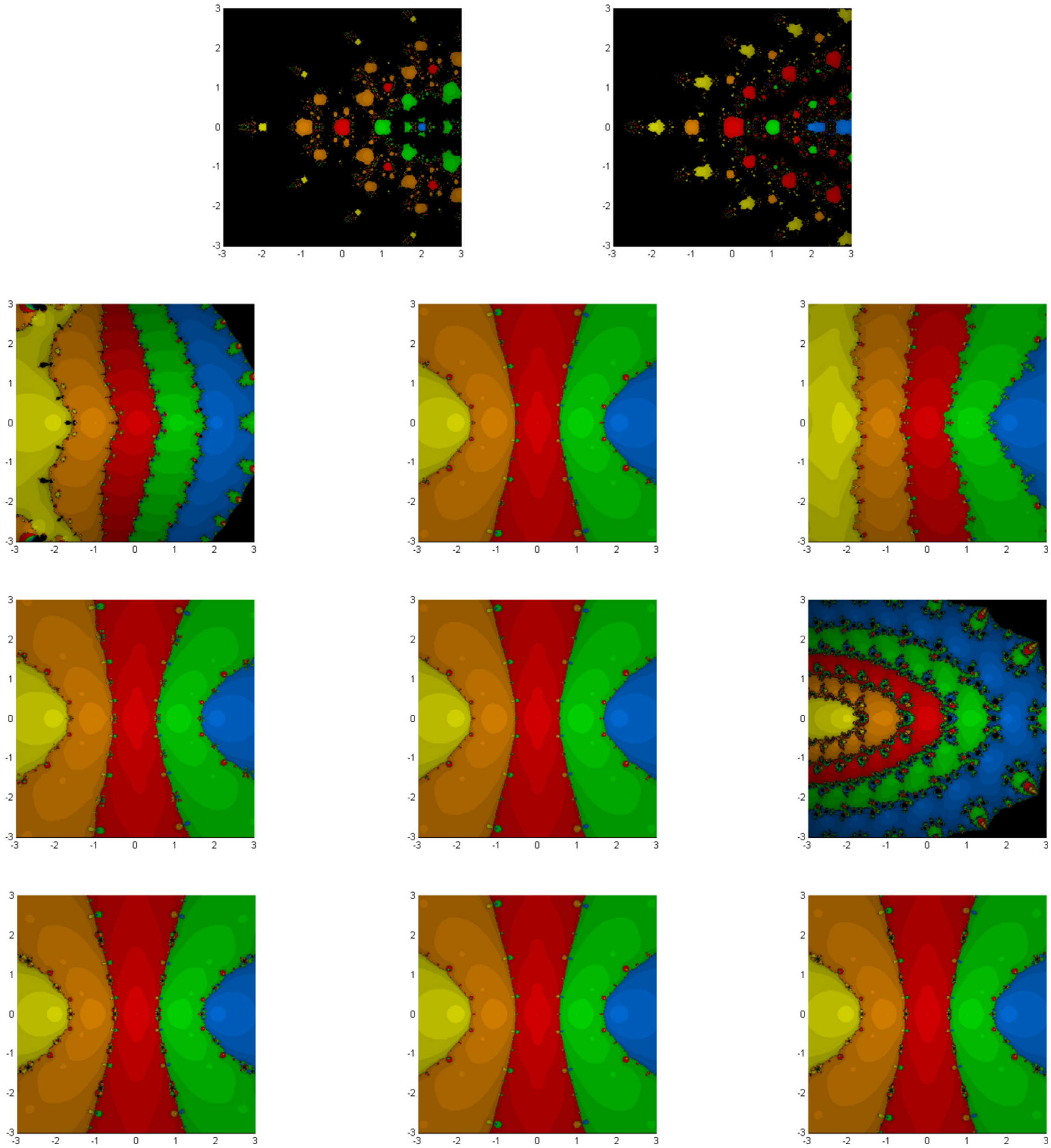

Fig. 2. Top row for KB1 (left) and KB2 (right), second row for KB3 (left), KB4 (center), and KB5 (right), third row for KB6 (left), KB7 (center) and KB8 (right), bottom row for King with $\beta = 3 - 2\sqrt{2}$ (left), $\beta = 0$ (center) and $\beta = 0.1$ (right) for the roots of the polynomial $z^5 - 5z^3 + 4z$.

Table 2
Average number of iterations per point for each example (1–6) and each of the 11 methods.

Example	1	2	3	4	5	6	Average
KB1	38.0463	36.0777	35.2893	31.5894	36.5235	37.6280	35.8590
KB2	36.7347	36.0863	31.6800	26.8284	37.7211	36.1124	34.1938
KB3	21.7486	7.2352	11.9838	8.1814	25.2988	9.8164	15.044
KB4	3.7810	4.1420	5.0251	6.1719	4.0078	4.6179	4.6242
KB5	8.8653	4.4799	8.2300	7.1327	10.2061	5.7878	7.4503
KB6	3.9215	4.1695	5.2388	6.4170	4.0466	4.7026	4.7430
KB7	3.7478	4.1320	4.9423	6.0892	3.9979	4.5666	4.5793
KB8	31.0994	11.3453	18.8762	12.8613	26.1077	16.8813	19.5285
King0	4.2024	4.3839	6.7290	8.5668	4.2425	5.5509	5.6126
King	3.7381	4.1246	4.7468	5.7538	3.9934	4.4682	4.4708
King01	3.9736	4.2820	6.4665	8.4667	4.1228	5.2308	5.4237

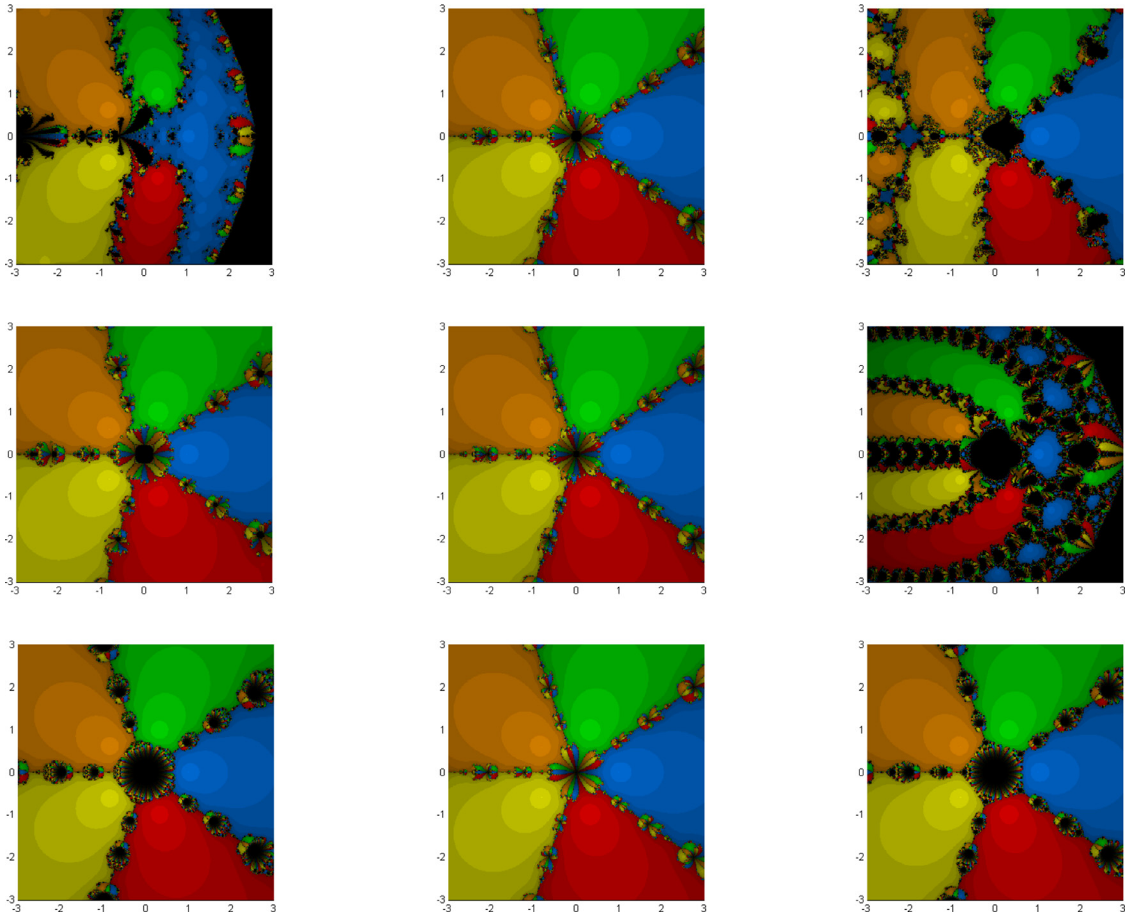

Fig. 3. Top row for KB3 (left), KB4 (center), and KB5 (right), second row for KB6 (left), KB7 (center) and KB8 (right), bottom row for King with $\beta = 3 - 2\sqrt{2}$ (left), $\beta = 0$ (center) and $\beta = 0.1$ (right) for the roots of the polynomial $z^5 - 1$.

Table 3
Standard deviation for each example (1–6) and each of the 11 methods.

Example	1	2	3	4	5	6
KB1	8.0621	11.0025	11.8690	14.2144	10.4785	8.7399
KB2	10.1726	10.9618	14.3347	16.4513	8.7597	10.7734
KB3	16.2666	8.3862	13.6347	9.1354	17.1542	11.4326
KB4	1.4320	1.1531	3.1224	4.5166	1.0597	1.8245
KB5	11.9018	1.9052	9.6221	6.9423	12.7737	4.0665
KB6	1.5607	1.2182	4.2277	5.3978	1.1616	1.9386
KB7	1.3565	1.1242	2.7293	4.2101	1.0316	1.7361
KB8	14.9765	11.1584	15.5711	12.5419	16.4313	14.6135
King0	2.8940	2.0927	7.2217	9.2797	2.0176	4.3342
King	1.3468	1.0950	2.0903	3.3499	1.0138	1.4578
King01	1.6315	1.5827	7.0440	9.5377	1.2501	3.7457

Example 3. In the third example we have taken a polynomial of degree 5 with the 5 roots of unity, i.e.

$$p_3(z) = z^5 - 1. \tag{12}$$

The basins for all methods except KB1 and KB2 are given in Fig. 3. Qualitatively, the results are exactly the same as before. The same is true when consulting Tables 2 and 3. As for the CPU time (see Table 4), we find that the fastest is King with $\beta = 3 - 2\sqrt{2}$, followed by KB7, King with $\beta = 0.1$, KB4 and KB6. In all these examples so far we see that the special cases we have for KB are very competitive. On the other hand the two cases suggested by Khattri and Babajee[1] are not.

Example 4. In the next example we have taken a polynomial of degree 7

$$p_4(z) = z^7 - 1. \tag{13}$$

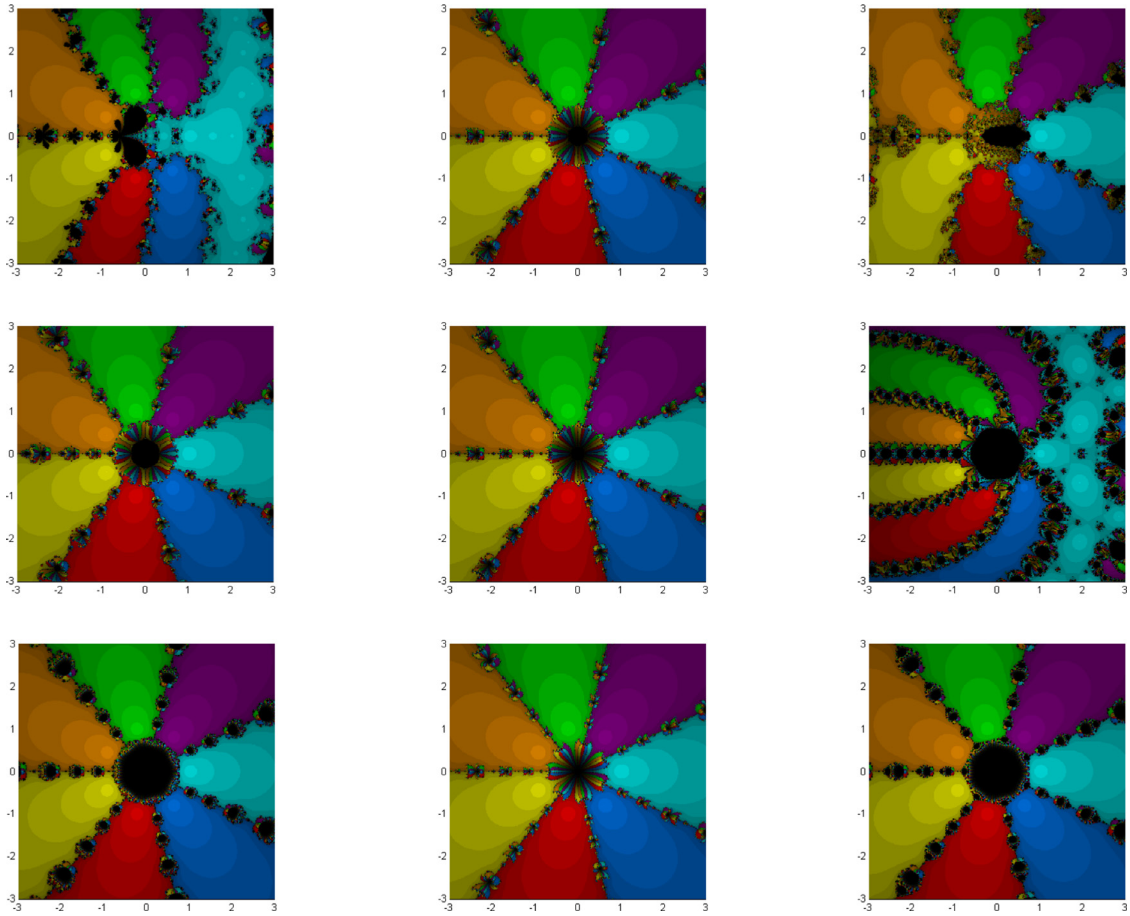

Fig. 4. Top row for KB3 (left), KB4 (center), and KB5 (right), second row for KB6 (left), KB7 (center) and KB8 (right), bottom row for King with $\beta = 3 - 2\sqrt{2}$ (left), $\beta = 0$ (center) and $\beta = 0.1$ (right) for the roots of the polynomial $z^7 - 1$.

Table 4
CPU time (in seconds) required for each example (1–6) and each of the 11 methods using a Samsung Premium Ultrabook NT900X4C.

Example	1	2	3	4	5	6	Average
KB1	5609.75	7061.65	4837.03	5176.18	7086.34	19856.07	8226.17
KB2	4991.67	9374.46	4275	4355.73	7281.07	18890.17	8194.68
KB3	2763.92	1352.06	1547.04	1304.5	4666.57	4972.54	2767.77
KB4	517.06	744.03	646.56	970.84	756.53	2335.75	995.13
KB5	1118.65	864.35	1094.48	1218.62	1910.98	2881.53	1514.77
KB6	499.29	750.15	683.15	1034.46	754.39	2365.25	1014.45
KB7	477.09	745.37	631.5	960.04	739.67	2262.71	969.40
KB8	3886.6	2020.34	2564.96	1984.89	4903	8518.82	3979.77
King0	466.73	644.28	729.64	1123.56	645.35	2218.21	971.30
King	356.23	556.93	465.7	703.85	558.68	1711.2	725.43
King01	376.92	647.67	639.54	1090.06	587.57	2001.98	890.62

The basins are plotted in Fig. 4. The qualitative results are identical to the previous example. The average number of iterations per point confirms that conclusion. The CPU time is the lowest for King with $\beta = 3 - 2\sqrt{2}$, followed by KB7, KB4 and KB6 in that order. This matches perfectly the qualitative results.

In the last two examples, we consider polynomials with complex coefficients. We find that these are more challenging problems.

Example 5. We have considered a cubic polynomial with complex coefficients

$$p_5(z) = z^3 + 2z^2 - 3iz^2 - \frac{3}{4}z - \frac{9}{2}iz - \frac{7}{4} - \frac{3}{2}i \tag{14}$$

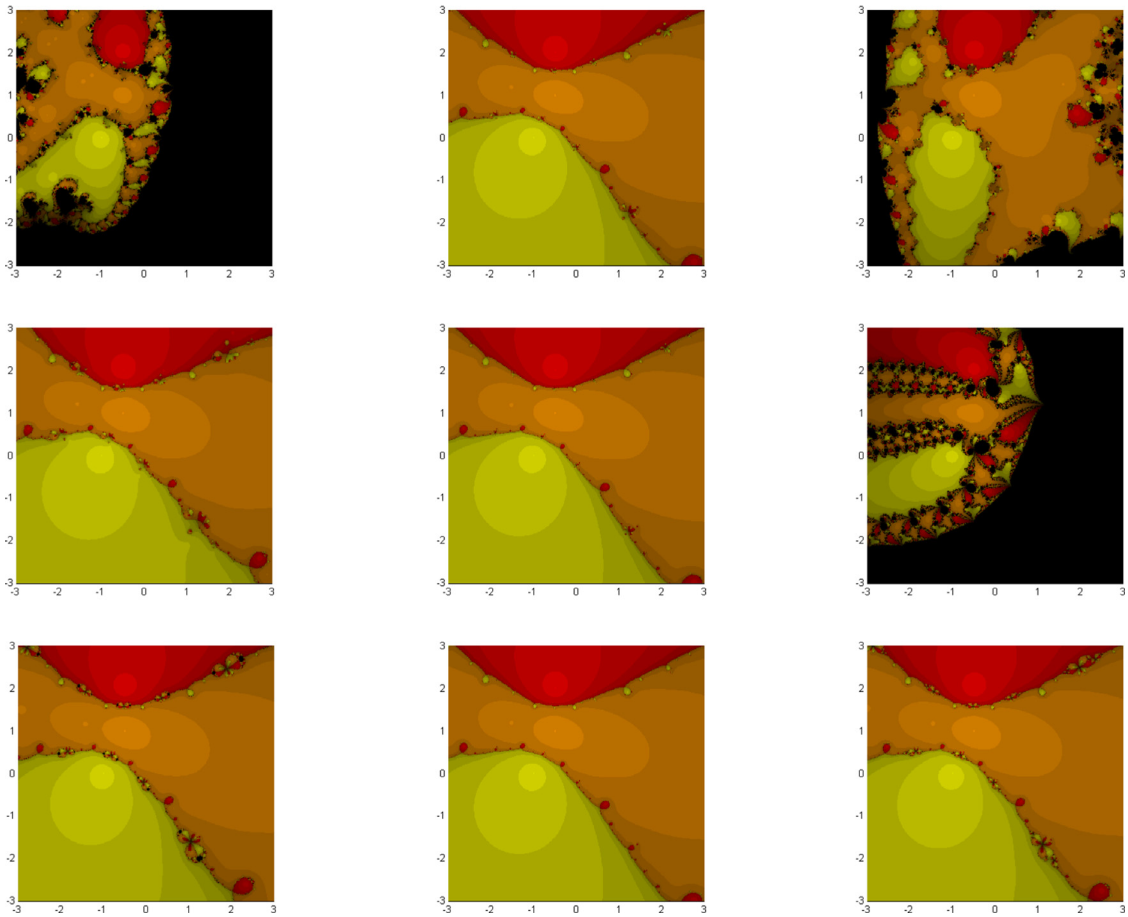

Fig. 5. Top row for KB3 (left), KB4 (center), and KB5 (right), second row for KB6 (left), KB7 (center) and KB8 (right), bottom row for King with $\beta = 3 - 2\sqrt{2}$ (left), $\beta = 0$ (center) and $\beta = 0.1$ (right) for the roots of the polynomial $p_5(z)$.

Table 5
The number of points requiring 40 iterations for each method and each example.

Example	1	2	3	4	5	6	Average
KB1	341116	320272	8749	264153	325146	336237	265945.5
KB2	327317	320108	268543	218628	338255	319113	298660.7
KB3	157472	21118	67289	26055	206765	43320	87003.17
KB4	107	0	1131	2876	0	0	685.6667
KB5	44983	515	27935	12904	54882	2368	23931.17
KB6	130	5	3735	6318	5	12	1701
KB7	79	0	8749	1442	0	0	1711.667
KB8	265079	44352	123111	59293	208347	96392	132762.3
King0	93	0	18	468	0	0	96.5
King	1612	572	9279	22505	671	1342	5996.833
King01	3	12	8749	24288	0	663	5619.167

The results are plotted in Fig. 5. The best methods are almost as before King with $\beta = 0$, KB7, KB4 and KB6, but the other King methods are also good. In terms of CPU time, King method with any of the parameter values we tried were faster than KB methods.

Example 6. In the last example we took a polynomial of degree 6 with complex coefficients

$$p_6(z) = z^6 - \frac{1}{2}z^5 + \frac{11}{4}(1+i)z^4 - \left(\frac{3}{4}i + \frac{19}{4}\right)z^3 - \left(\frac{5}{4}i + \frac{11}{4}\right)z^2 - \left(\frac{1}{4}i + \frac{11}{4}\right)z + \frac{3}{2} - 3i \tag{15}$$

The basins are presented in Fig. 6. The most chaotic is KB8 followed by KB3 and KB5. The best ones seem to be as before King with $\beta = 0$, KB7, KB4 and KB6. This is confirmed by the average number of iterations per point (see Table 2). The CPU

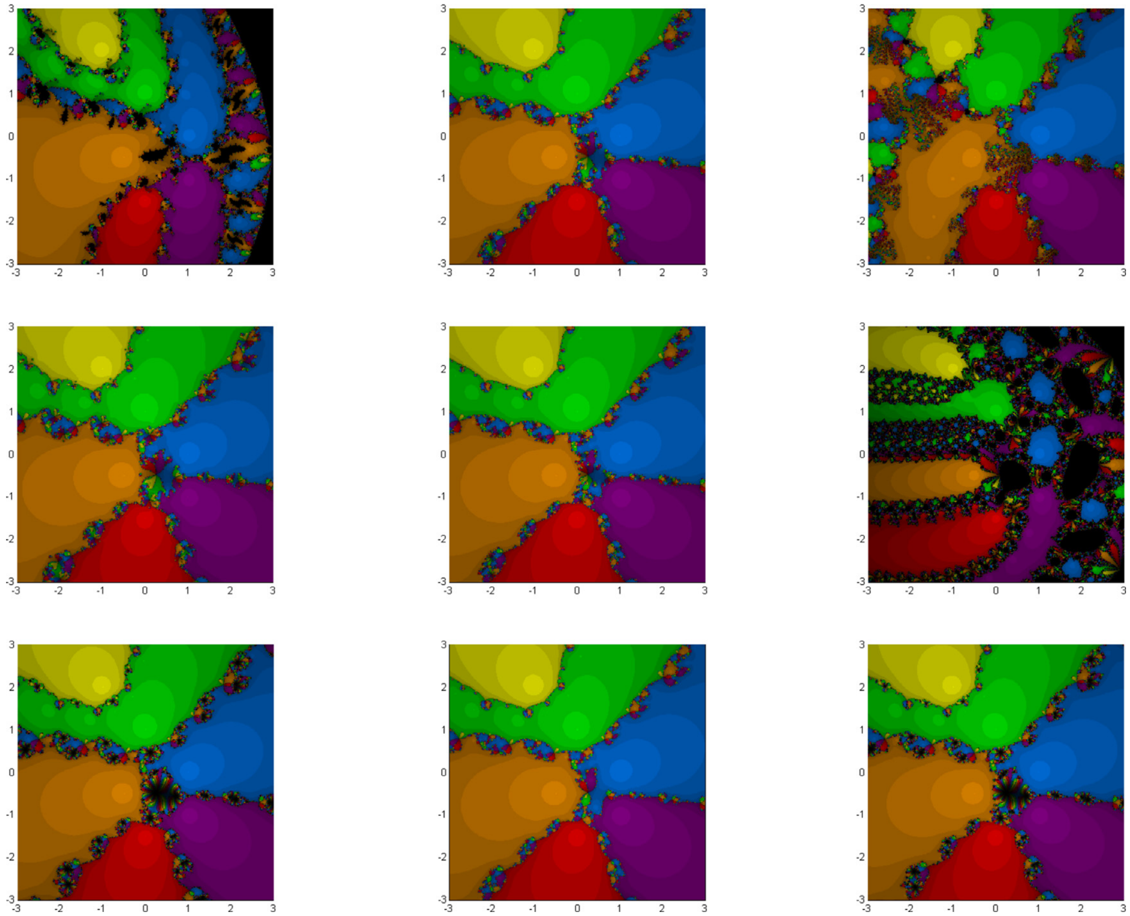

Fig. 6. Top row for KB3 (left), KB4 (center), and KB5 (right), second row for KB6 (left), KB7 (center) and KB8 (right), bottom row for King with $\beta = 3 - 2\sqrt{2}$ (left), $\beta = 0$ (center) and $\beta = 0.1$ (right) for the roots of the polynomial $p_6(z)$.

time was lowest for King method with the 3 different values of the parameter β followed by KB7, KB4 and KB6 in that order.

To summarize, we have averaged all the values in Table 2 over all 6 examples. The results show that King with $\beta = 3 - 2\sqrt{2}$ requires 4.47 iterations per point followed by KB7 (4.58), KB4 (4.62) and KB6 (4.74). All the others require more than 5.4 iterations per point (on average) with the worst being KB1 and KB2 with over 34 iterations per point. In terms of CPU time, the fastest is King with $\beta = 3 - 2\sqrt{2}$ (725 s) followed by King with $\beta = 0.1$ (890 s), KB7 (969 s), King with $\beta = 0$ (971 s) and KB4 (995 s). All the others require more than 1000 s on average over all examples with the highest being KB1 (8226 s) and KB2 (8194 s).

Another measure for comparison is the number of points requiring 40 iterations. These are the points colored black. We tabulated the numbers for each method and each example in Table 5. It is clear that King with $\beta = 0$ had the lowest such number on average (96.5) followed by KB4 (685.7 points), KB6 (1701 points) and KB7 (1711.7 points). King with $\beta = 3 - 2\sqrt{2}$ had 5996.8 points on average, even though on average this was the fastest and had the lowest number of iterations per point. King with $\beta = 0$ has the lowest number of black points but came fourth in CPU time and sixth in the average number of iterations per point. It is now clear that the best methods are KB7, KB4 and KB6 in that order.

Conclusion

In this paper we have experimented with several possible parameter combinations for Khattri et al.'s family of methods and compared them to 3 members of King's family of methods. We found based on several criteria that the 3 members KB7, KB4 and KB6 are the best.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2005012).

References

[1] S.K. Khattri, D.K.R. Babajee, Fourth-order family of iterative methods with four parameters, Int. J. Math. Comput. 19 (2) (2013) 1–10.
[2] M.S. Petković, B. Neta, L.D. Petković, J. Džunić, Multipoint Methods for Solving Nonlinear Equations, Elsevier, 2013.
[3] J.F. Traub, Iterative Methods for the Solution of Equations, Chelsea Publishing Company, New York, 1977.
[4] B. Neta, Numerical Methods for the Solution of Equations, net-a-sof, Monterey, 1983.
[5] E. Halley, A new, exact and easy method of finding the roots of equations generally and that without any previous reduction, Phil. Trans. Roy. Soc. Lond. 18 (1694) 136–148.
[6] H.T. Kung, J.F. Traub, Optimal order of one-point and multipoint iterations, J. Assoc. Comput. Mach. 21 (1974) 643–651.
[7] H. Woźniakowski, Maximal order of multipoint iteration using n evaluations, in: J.F. Traub (Ed.), Analytic Computational Complexity, Academic Press, New York, 1976, pp. 75–107.
[8] R.F. King, A family of fourth-order methods for nonlinear equations, SIAM Numer. Anal. 10 (1973) 876–879.
[9] B. Neta, A sixth order family of methods for nonlinear equations, Int. J. Comput. Math. 7 (1979) 157–161.
[10] B. Neta, On a family of multipoint methods for nonlinear equations, Int. J. Computer Math. 9 (1981) 353–361.
[11] X. Wang, L. Liu, Modified ostrowski's method with eighth-order convergence and high efficiency index, Appl. Math. Lett. 23 (2010) 549–554.
[12] R. Thukral, M.S. Petković, Family of three-point methods of optimal order for solving nonlinear equations, J. Comput. Appl. Math. 233 (2010) 2278–2284.
[13] M. Scott, B. Neta, C. Chun, Basin attractors for various methods, Appl. Math. Comput. 218 (2011) 2584–2599.
[14] E.R. Vrscay, W.J. Gilbert, Extraneous fixed points, basin boundaries and chaotic dynamics for Schröder and Önig rational iteration functions, Numer. Math. 52 (1988) 1–16.
[15] B.D. Stewart, Attractor Basins of Various Root-Finding Methods, Naval Postgraduate School, Department of Applied Mathematics, Monterey, CA, 2001 M.S. thesis.
[16] B. Kalantari, Y. Jin, On extraneous fixed-points of the basic family of iteration functions, BIT Numer. Math. 43 (2003) 453–458.
[17] S. Amat, S. Busquier, S. Plaza, Iterative root-finding methods, 2004. Unpublished report.
[18] S. Amat, S. Busquier, S. Plaza, Review of some iterative root-finding methods from a dynamical point of view, Sci. Ser. A Math. Sci. 10 (2004) 3–35.
[19] S. Amat, S. Busquier, S. Plaza, Dynamics of a family of third-order iterative methods that do not require using second derivatives, Appl. Math. Comput. 154 (2004) 735–746.
[20] S. Amat, S. Busquier, S. Plaza, Dynamics of the King and Jarratt iterations, Aeq. Math 69 (2005) 212–223.
[21] F. Chicharro, A. Cordero, J.M. Gutiérrez, J.R. Torregrosa, Complex dynamics of derivative-free methods for nonlinear equations, Appl. Math. Comput. 219 (2013) 7023–7035.
[22] C. Chun, M.Y. Lee, B. Neta, J. Džunić, On optimal fourth-order iterative methods free from second derivative and their dynamics, Appl. Math. Comput. 218 (2012) 6427–6438.
[23] C. Chun, B. Neta, S. Kim, On Jarratt's family of optimal fourth-order iterative methods and their dynamics, Fractals 22 (2014) 1450013.
[24] A. Cordero, J. García-Maimó, J.R. Torregrosa, M.P. Vassileva, P. Vindel, Chaos in King's iterative family, Appl. Math. Lett. 26 (2013) 842–848.
[25] B. Neta, C. Chun, M. Scott, Basins of attraction for optimal eighth order methods to find simple roots of nonlinear equations, Appl. Math. Comput. 227 (2014) 567–592.
[26] B. Neta, M. Scott, C. Chun, Basin of attractions for several methods to find simple roots of nonlinear equations, Appl. Math. Comput. 218 (2012) 10548–10556.
[27] B. Neta, M. Scott, C. Chun, Basin attractors for various methods for multiple roots, Appl. Math. Comput. 218 (2012) 5043–5066.
[28] A.A.M. nán, Different anomalies in a Jarratt family of iterative root-finding methods, Appl. Math. Comput. 233 (2014) 29–38.
[29] A.A.M. nán, A. Cordero, J.M. Gutiérrez, J.R. Torregrosa, Real qualitative behavior of a fourth-order family of iterative methods by using the convergence plane, Math. Comput. Simulat. 105 (2014) 49–61.
[30] B. Neta, C. Chun, Basins of attraction for Zhou-Chen-Song fourth order family of methods for multiple roots, Math. Comput. Simulat. 109 (2015) 74–91.
[31] B. Neta, C. Chun, Basins of attraction for several optimal fourth order methods for multiple roots, Math. Comput. Simulat. 103 (2014) 39–59.
[32] B. Neta, C. Chun, On a family of Laguerre methods to find multiple roots of nonlinear equations, Appl. Math. Comput. 219 (2013) 10987–11004.
[33] C. Chun, B. Neta, J. Kozdon, M. Scott, Choosing weight functions in iterative methods for simple roots, Appl. Math. Comput. 227 (2014) 788–800.