Punjab University Journal of Mathematics
(ISSN 1016-2526)
Vol. 52(6)(2020) pp. 31-44

On Family of Simultaneous Method for Finding Distinct as Well as Multiple Roots of Non-linear Equation

[a]Nazir Ahmad Mir, [b]Mudassir Shams,[c]Naila Rafiq, [d]Saima Akram,[e]Rafiq Ahmed
[a,b]Department of Mathematics and Statistics, Riphah International University I-14, Islamabad 44000, Pakistan
[c]Department of Mathematics, NUML, Islamabad, Pakistan.
[d]Centre for Advanced Studies in Pure and Applied Mathematics Bahauddin Zakariya University, Multan 60000, Pakistan.
[e]Departmnet of Mathematics, University of Sindh, Jamshoro, Pakistan.
Email: [a]nazir.ahmad@riphah.edu.pk, [b]mudassir.shams@riphah.edu.pk,
[c]nrafique@numl.edu.pk,[d]saimaakram@bzu.edu.pk, [e]ra_memon@yahoo.com

Received: 24 February, 2020 / Accepted: 07 May, 2020 / Published online: 01 June, 2020

Abstract.: We construct a family of 2-step simultaneous methods for determining all the distinct roots of single variable non-linear equations. We further extend this family of simultaneous methods to the case of multiple roots. It is proved that both the family of methods are of convergence order eight and has better computation efficiency as compared to some other simultaneous methods in the literature. At the end, numerical test examples are given to demonstrate the efficiency and performance of the newly constructed simultaneous methods.

AMS (MOS) Subject Classification Codes:

Key Words: Distinct roots, Multiple roots, Non-Linear equation, Iterative methods, Simultaneous Methods, Computational Efficiency.

1. Introduction

One of the primeral problems in mathematics is the determination of roots of non-linear equation. There are number of applications of non-linear equation in science and engineering. Newton's method is a numerical method which finds a single root at a time. The simultaneous iterative methods such as, Weierstrass method is used to find all the distinct roots. The iterative methods for finding single and multiple roots of non linear polynomial equation have been studied by Wang [35], Li [15], Osda [19], Chun and Neta [5], Homeier [8], Bi [2], Proinov and Ivanov [30], Ivanov [11] and many others. On the other hand, there are lot of numerical iterative methods devoted to approximate all roots of polynomial equation simultaneously (see, e.g. Weierstrass' [36], Kanno [12], Proinov [23], Sendov [34], Petkovi´c [20], Mir [16], Nourein [18], Aberth [1], Cholakov [4], Iliev [10], Kyncheva [13] and the references therein). The simultaneous iterative methods are popular as compared to single root finding methods due to their wider range of convergence, are more stable and can be implemented for parallel computing as well. Further details on simultaneous methods, their convergence analysis, efficiency and parallel implementations can be seen in [6, 24, 35, 25, 18, 26, 22, 7, 27, 28, 3, 29, 31, 32, 33, 9, 14] and references cited there in. The main objective of this paper is to develop simultaneous methods which have a higher convergence order and can find distinct as well as multiple roots of non-linear polynomial equation, say:

$$f(x) = 0. \qquad (1.\ 1)$$

2. CONSTRUCTIONS OF SIMULTANEOUS METHODS

We construct here a family of higher order simultaneous methods which are more efficient than the methods existing in literature.

2.1. **Construction of Family of Simultaneous Methods for Distinct Roots.** Consider 2-step method proposed by Li et al. [36]:

$$\begin{cases} y^{(k)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \\ z^{(k)} = y^{(k)} - \frac{f(y^{(k)})}{f'(y^{(k)})-\alpha f(y^{(k)})}, \end{cases} \quad (k = 0, 1, 2, ...) \qquad (2.\ 2)$$

where α is any arbitrary real parameter. The 2-step method (2. 2), is of fourth order convergence, if $\alpha = 0$, then it is well-known 2-step Newton's method for calculating single root at a time. We would like to convert it into simultaneous method for approximating all the distinct roots of (1. 1).
Method (2. 2) can be written as:

$$\begin{aligned} y^{(k)} &= x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \\ z^{(k)} &= y^{(k)} - \left(\frac{1}{\frac{f'(y^{(k)})}{f(y^{(k)})} - \alpha} \right). \end{aligned} \qquad (2.\ 3)$$

Let

$$w(x_i) = \frac{f(x_i)}{\prod\limits_{\substack{j \neq i \\ j=1}}^{n} (x_i - x_j)}, i = 1, 2, 3, ..., n, \quad \text{(Weierstrass' Correction [8])}. \qquad (2.\ 4)$$

Taking natural logarithm of (2. 4) and then differentiating, we obtaine (see [14]):

$$\frac{w'(x_i)}{w(x_i)} = \frac{f'(x_i)}{f(x_i)} - \sum_{\substack{j \neq i \\ j=1}}^{n} \frac{1}{(x_i - x_j)}. \qquad (2.\ 5)$$

Let $x_1, \ldots, x_n$ be the distinct approximations to the roots $\xi_{1,\ldots,}\xi_n$ of non-linear equation (1. 1). Replacing x_j by x_j^* in (2. 5), we have:

$$\frac{w'(x_i)}{w(x_i)} = \frac{f'(x_i)}{f(x_i)} - \sum_{\substack{j \neq i \\ j=1}}^{n} \frac{1}{(x_i - x_j^*)}, \qquad (2.\ 6)$$

or equivalently

$$\frac{w(x_i)}{w'(x_i)} = \frac{1}{\frac{f'(x_i)}{f(x_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{n} \frac{1}{(x_i - x_j^*)}} = \frac{1}{\frac{1}{N(x_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{n} \frac{1}{(x_i - x_j^*)}}, \tag{2. 7}$$

$$\text{where } x_j^* = x_j - \frac{f(x_j)}{f'(x_j)} \text{ and } N(x_i) = \frac{f(x_i)}{f'(x_i)}.$$

Replacing $\frac{f(x_i)}{f'(x_i)}$ by $\frac{w(x_i)}{w'(x_i)}$ in (2. 3), we have:

$$\begin{aligned} y_i &= x_i - \frac{1}{\frac{1}{N(x_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{n} \frac{1}{(x_i - x_j^*)}}, (i, j = 1, 2, ..., n) \\ z_i &= y_i - \frac{1}{\frac{1}{N(y_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{n} \frac{1}{(y_i - y_j)} - \alpha}, \end{aligned} \tag{2. 8}$$

where α is a real parameter.

Thus, we have constructed a new family of 2-step simultaneous methods (2. 8), abbreviated as $MMN8D$, for extracting all the distinct roots of non-linear equation.

2.1.1. *Construction of Family of Simultaneous Methods for Multiple Roots.* Now, family of method (2. 8) for extracting all the distinct roots of non-linear equations modified for finding multiple roots of (2. 4) as given by

$$\begin{aligned} y_i &= x_i - \frac{\sigma_i}{\frac{\sigma_i}{N(x_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{\sigma_j}{(x_i - x_j^*)}}, (i, j = 1, 2, ..., s \leq n) \\ z_i &= y_i - \frac{\sigma_i}{\frac{\sigma_i}{N(y_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - y_j)} - \alpha}, \end{aligned} \tag{2. 9}$$

where

$$x_j^* = x_j - \sigma_j \frac{f(x_j)}{f'(x_j)},$$

and σ_i is the multiplicity of actual multiple roots ζ_i. It should be noted that we denote $y(x_i)$ by y_i and $z(x_i)$ by z_i.

We denote the method (2. 9) by $MMN8M$.

2.2. **Convergence Analysis.** In this section, the convergence analysis of a family of two-step simultaneous methods (2. 9) given in form of the following theorem. Obviously, convergence for the method (2. 8) will follow from the convergence of the method (2. 9) from theorem (1) when the multiplicities of the roots are simple.

Theorem 2.3. *:Let $\xi_1, ..., \xi_s$ be all distinct roots of non-linear polynomial equation (1. 1) with multiplicites $\sigma_1, ..., \sigma_s$, respectively. If $x_1^{(0)}, ..., x_s^{(0)}$ be the initial approximations of the roots respectively and sufficiently close to actual roots, the order of convergence of method (2. 9) equals eight.*

Proof. Let

$$\begin{aligned} \epsilon_i &= x_i - \xi_i, \\ \epsilon_i' &= y_i - \xi_i, \\ \text{and } \epsilon_i'' &= z_i - \xi_i \end{aligned}$$

be the errors in x_i, y_i and z_i approximations respectively. Considering the first step of (2. 9), which is

$$y_i = x_i - \frac{\sigma_i}{\frac{\sigma_i}{N(x_i)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{\sigma_j}{(x_i - x_j^*)}},$$

where

$$N(x_i) = \frac{f(x_i)}{f'(x_i)}.$$

Then, obviously for distinct roots:

$$\frac{1}{N(x_i)} = \frac{f'(x_i)}{f(x_i)} = \sum_{j=1}^{n} \frac{1}{(x_i - \xi_j)} = \frac{1}{(x_i - \xi_i)} + \sum_{\substack{j\neq i \\ j=1}}^{n} \frac{1}{(x_i - \xi_j)}.$$

Thus, for multiple roots we have from (2. 9):

$$\begin{aligned} y_i &= x_i - \frac{\sigma_i}{\frac{\sigma_i}{(x_i-\xi_i)} + \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{\sigma_j}{(x_i-\xi_j)} - \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{\sigma_j}{(x_i - x_j^*)}}, \\ y_i - \xi_i &= x_i - \xi_i - \frac{\sigma_i}{\frac{\sigma_i}{(x_i-\zeta_i)} + \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{\sigma_j (x_i - x_j^* - x_i + \zeta_j)}{(x_i-\zeta_j)(x_i - x_j^*)}}, \\ \epsilon_i' &= \epsilon_i - \frac{\sigma_i}{\frac{\sigma_i}{\epsilon_i} + \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{-\sigma_j (x_j^* - \xi_j)}{(x_i-\xi_j)(x_i - x_j^*)}}, \\ &= \epsilon_i - \frac{\sigma_i \epsilon_i}{\sigma_i + \epsilon_i \sum\limits_{\substack{j\neq i \\ j=1}}^{s} \frac{-\sigma_j (x_j^* - \xi_j)}{(x_i-\xi_j)(x_i - x_j^*)}}, \\ &= \epsilon_i - \frac{\sigma_i . \epsilon_i}{\sigma_i + \epsilon_i \sum\limits_{\substack{j\neq i \\ j=1}}^{s} E_i \epsilon_j^2}, \end{aligned}$$

where $x_j^* - \xi_j = \epsilon_j^2$ ([17]) and $E_i = \frac{-\sigma_j}{(x_i-\xi_j)(x_i-x_j^*)}$.
Thus,

$$\epsilon_i^{'} = \frac{\epsilon_i^2 \sum\limits_{\substack{j \neq i \\ j=1}}^{s} E_i \epsilon_j^2}{\sigma_i + \epsilon_i \sum\limits_{\substack{j \neq i \\ j=1}}^{s} E_i \epsilon_j^2}. \tag{2. 10}$$

If it is assumed that absolute values of all errors ϵ_j $(j = 1, 2, 3, ...)$ are of the same order as, say $|\epsilon_j| = O\,|\epsilon|$, then from (2. 10), we have:

$$\epsilon_i^{'} = O(\epsilon)^4. \tag{2. 11}$$

From second equation of (2. 9),

$$\begin{aligned} z_i &= y_i - \frac{\sigma_i}{\frac{\sigma_i}{N(y_i)} - \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - y_j)} - \alpha}, \\ z_i - \xi_i &= y_i - \xi_i - \frac{\sigma_i}{\frac{\sigma_i}{y_i - \xi_i} + \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - \xi_j)} - \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - y_j)} - \alpha}. \end{aligned}$$

This implies,

$$\begin{aligned} \epsilon_i^{''} &= \epsilon_i^{'} - \frac{\sigma_i}{\frac{\sigma_i}{\epsilon_i^{'}} + \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - \xi_j)} - \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - y_j)} - \alpha} \\ &= \epsilon_i^{'} - \frac{\sigma_i . \epsilon_i^{'}}{\sigma_i + \epsilon_i^{'} \left(\sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j . (y_i - y_j - y_i + \xi_j)}{(y_i - \xi_j)(y_i - y_j)} \right) - \epsilon_i^{'} \alpha}, \\ &= \epsilon_i^{'} - \frac{\sigma_i . \epsilon_i^{'}}{\sigma_i + \epsilon_i^{'} \left(\sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{-\sigma_j . (y_j - \xi_j)}{(y_i - \xi_j)(y_i - y_j)} \right) - \epsilon_i^{'} \alpha}, \\ &= \epsilon_i^{'} - \frac{\sigma_i \epsilon_i^{'}}{\sigma_i + \epsilon_i^{'} \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \epsilon_j^{'} F_i - \epsilon_i^{'} \alpha}, \text{where } F_i = \frac{-\sigma_j}{(y_i - \xi_j)(y_i - y_j)}. \end{aligned}$$

This implies,

$$\begin{aligned} \epsilon_i^{''} &= \epsilon_i^{'} - \frac{\sigma_i.\epsilon_i^{'}}{\sigma_i + \epsilon_i^{'}\left(\sum\limits_{\substack{j\neq i\\ j=1}}^{s} \epsilon_j^{'} F_i - \alpha\right)}, \\ &= (\epsilon_i^{'})^2 \left(\frac{\sum\limits_{\substack{j\neq i\\ j=1}}^{s} \epsilon_j^{'} F_i - \alpha}{\sigma_i + \epsilon_i^{'}\left(\sum\limits_{\substack{j\neq i\\ j=1}}^{s} \epsilon_j^{'} F_i - \alpha\right)} \right). \end{aligned}$$

Since from (2. 11) $\epsilon_i^{'} = O(\epsilon)^4$, thus,

$$\begin{aligned} \epsilon_i^{''} &= O((\epsilon)^4)^2, \\ \epsilon_i^{''} &= O(\epsilon)^8, \end{aligned}$$

which shows convergence order of method (2. 9) is eight. Hence, it proves the theorem. ∎

3. COMPUTATIONAL ASPECT

Here we compare the computational efficiency and convergence behavior of the M. S. Petkovi´c, L. D. Petkovi´c, J. D. Zunic ,methods [10, 3] and the new method(2. 9). As presented in [3], the efficiency of an iterative method can be estimated using the efficiency index given by:

$$EL(m) = \frac{\log \mathbf{r}}{\mathbf{D}}, \tag{3. 12}$$

where $\mathbf{D}$ is the computational cost and $\mathbf{r}$ is the order of convergence of the iterative method. Using arithmetic operation per iteration with certain weight depending on the execuation time of operation to evaluate the computational cost $\mathbf{D}$. The weights used for division, multiplication and addition plus subtraction are w_d, w_m, w_{AS} respectively. For a given polynomial of degree m and n roots, the number of division, multiplication addition and subtraction per iteration for all roots are denoted by D_m, M_m and AS_m. The cost of computation can be calculated as:

$$\mathbf{D} = \mathbf{D}(m) = w_{as}AS_m + w_m M_m + w_d D_m, \tag{3. 13}$$

thus (3. 12) becomes:

$$EL(m) = \frac{\log \mathbf{r}}{w_{as}AS_m + w_m M_m + w_d D_m} \tag{3. 14}$$

Considering the number of operations of a complex polynomial with real and complex roots reduce to operation of real arithmetic, given in Table 3.1 as polynomial degree m

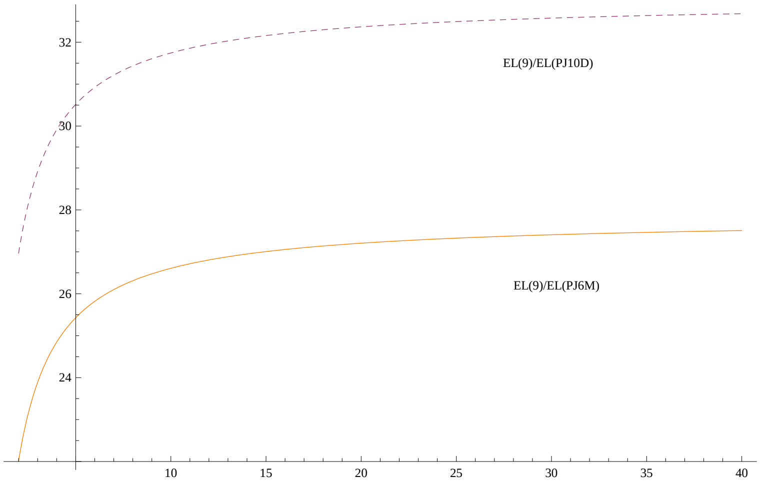

FIGURE 1

taking the dominant term of order (m^2). Apply (3. 14) and data given in Table 3.1, we calculate the percentage ratio $\rho((2.9),(X))$ [3] given by:

$$\rho((2.9),(X)) = \left(\frac{EL(2.9)}{EL(X)} - 1\right) \times 100 \text{ (in percent)}, \tag{3. 15}$$

where X is Petkovi´c methods. Figure 1 graphically illustrates these percentage ratios. It is evident from Figure 1 that the newly constructed simultaneous method 2.9 is more efficient as compared to the M. S. Petkovi´c, L. D. Petkovi´c, J. D. Zunic methods [10, 3].

Table 3.1: The number of basic operations

Methods	AS_m	M_m	D_m
Petkovic Method (PJ8M)	$15m^2 + O(m)$	$13m^2 + O(m)$	$2m^2 + O(m)$
Petkovic Method(PJ10D)	$22m^2 + O(m)$	$18m^2 + O(m)$	$2m^2 + O(m)$
New Method(2. 9)	$8m^2 + O(m)$	$10m^2 + O(m)$	$2m^2 + O(m)$

We also calculate the CPU execuation time, as all the calculations are done using maple 18 on (Processor Intel(R) Core(TM) i3-3110m CPU@2.4GHz with 64-bit Operating System. We observe that CPU time of the method MMN8M is less than M. S. Petkovic methods [10, 3], showing the dominance efficiency of our method (2. 9) as compared to them.

4. NUMERICAL RESULTS

Here, some numerical test examples are considered in order to demonstrate the performance of our family of two-step eighth order simultaneous methods, namely $MMN8D$ (2. 8) and $MMN8M$ (2. 9). We compare our family of methods with M. S. Petkovi´c, L. D. Petkovi´c, J. D. Zunic [10] method of order six for multiple roots (abbreviated as PJ6M method) and M. S. Petkovi´c, L. D. Petkovi´c, J. D. Zunic [3] method of order ten for multiple roots (abbreviated as PJ10D method). All the computations are performed using $Maple$ 18 with 64 digits floating point arithmetic. We take $\in = 10^{-30}$ as a tolerance and approximating the roots the following stopping criteria are used:

$$(i) \qquad e_i = \left| f\left(x_i^{(k+1)}\right)\right| < \in,$$

$$(ii) \qquad e_i = \left\| x_i^{(k+1)} - \xi_i \right\|_2 = \left(\sum_{i=1}^{n} \left| x_i^{(k+1)} - \xi_i \right| \right)^{\frac{1}{2}}, (k = 0, 1, ...),$$

where e_i represents the absolute error of function values in (i) and norm-2 in (ii) [10].

In all the examples for $MMN8M$, we have taken $\alpha = 0.001$. Numerical tests examples from [15, 17, 4] are provided in Tables $4.1(a)$, $4.1(b)$, $4.2(a)$, $4.2(b)$, $4.3(a)$, $4.3(b)$, $4.4(a)$ and $4.4(b)$ In Table $4.1(a)$, $4.2(a)$, $4.3(a)$, and Table $4.4(a)$ the stopping criteria (i) is used while in Table $4.1(b)$, $4.2(b)$, $4.3(b)$ and Table $4.4(b)$, stopping criteria (ii) is used. In all Tables CO represents the convergence order, n represents the number of iterations, γ=1 represents for multiplicity is equal to one , $\gamma \neq$1 represents for multiplicity not equal to one and CPU represents computational time in seconds. For multiplicity unity,we get numerical results for distinct roots. We observe that numerical results of the methods MMN8M and MMN8D are better than PJ6M and PJ10D methods on second iteration.The Figure 2 ,3, 4, 5 shows the residual fall of different methods for the examples 4.1,4.2,4.3,4.4, shows that MMN8M is more efficient as compared to the other methods.

Algorithm of simultaneous iterative methods (MMN8M)

Step 1: For given $x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, ..., x_s^{(0)}$, calculate $x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, ..., x_s^{(1)}$ such that

$$y_i = x_i - \frac{\sigma_i}{\frac{\sigma_i}{N(x_i)} - \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(x_i - x_j^*)}}, (i, j = 1, 2, ..., s \leq n),$$

$$z_i = y_i - \frac{\sigma_i}{\frac{\sigma_i}{N(y_i)} - \sum\limits_{\substack{j \neq i \\ j=1}}^{s} \frac{\sigma_j}{(y_i - y_j)} - \alpha},$$

where $x_j^* = x_j - \sigma_j \frac{f(x_j)}{f'(x_j)}$,and σ_i is the multiplicity of actual multiple roots ζ_i.

Step 2: For a given $\in > 0$, $\left| f\left(x_i^{(k+1)}\right)\right| < \in$ or $\left(\sum\limits_{i=1}^{n} \left| x_i^{(k+1)} - \xi_i \right| \right)^{\frac{1}{2}} < \in$, then stop.

Step 3: Set $k = k + 1$ and go to step 1.

Example 4.1[15]**:** Consider

$$f(x) = (x+1)^2\,(x+3)^3\,(x^2-2x+2)^2\,(x-1)^3\,(x^2-4x+5)^2\,(x^2+4x+5)^2,$$

with multiple exact roots ($\gamma \neq$1):

$$\xi_1 = -1,\ \xi_2 = -3,\ \xi_{3,4} = 1 \pm i,\ \xi_5 = 1,\ \xi_{6,7} = -2 \pm I,\ \xi_{8,9} = 2 \pm I.$$

The initial approximations have been taken as:

$$\overset{(0)}{x}_1 = -1.3 + .2i,\ \overset{(0)}{x}_2 = -2.8 - .2i,\ \overset{(0)}{x}_3 = 1.2 + 1.3i,\ \overset{(0)}{x}_4 = 0.8 - 1.2i,\ \overset{(0)}{x}_5 = 0.8 - 0.3i,$$

$$\overset{(0)}{x}_{6,7} = -1.8 \pm 1.2i,\ \overset{(0)}{x}_{8,9} = 1.8 \pm .8i.$$

For distinct roots ($\gamma = 1$):

$$f(x) = (x+1)\,(x+3)(x^2-2x+2)(x-1)(x^2-4x+5)(x^2+4x+5).$$

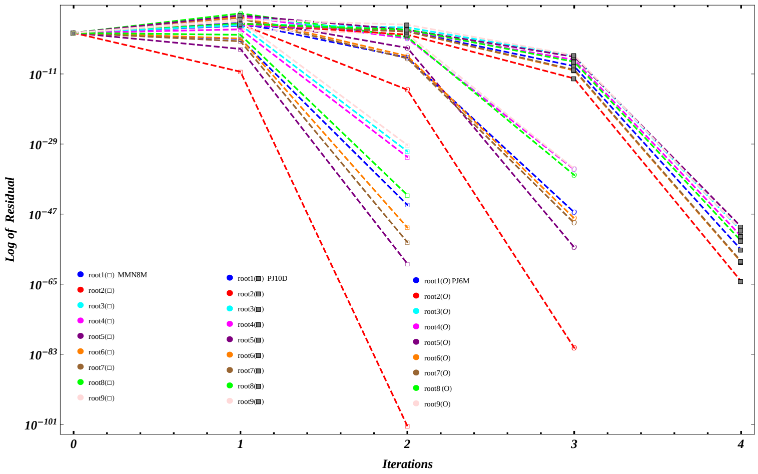

FIGURE 2. Residual fall for Example 4.1.

Table.4.1(a)

Method	CO	CPU	γ	n	e1	e2	e3	e4	e5	e6	e7	e8	e9
PJ6M	6	1.547	$\gamma \neq 1$	2	0.2e-46	0.4e-81	0.3 e-35	0.2e-35	0.2 e-55	0.8 e-48	0.5 e-49	0.1 e-36	0.4e-35
PJ6M	6	1.156	$\gamma = 1$	2	3.6e-1	4.5e-1	4.3 e-0	8.8 e-0	2.9 e-0	5.0 e-1	7.2 e-1	1.5 e-0	4.3e-0
PJ10D	10	0.797	γ=1	2	1.0e-9	6.9e-13	4.0 e-7	4.2 e-8	2.6 e-7	8.0 e-11	1.2 e-10	1.3 e-8	6.0e-7
MMN8M	8	0.203	$\gamma \neq 1$	2	0.2e-44	0.3e-101	0.1 e-30	0.4 e-32	0.1 e-59	0.3 e-50	0.6 e-54	0.7 e-42	0.5e-29
MMN8D	8	0.234	γ=1	2	3.8e-28	1.0 e-33	2.2 e-21	5.5 e-24	3.3 e-20	3.5e-35	1.8e-31	7.3 e-24	1.4e-21

Table.4.1(b)

Method	CO	CPU	γ	n	e1	e2	e3	e4	e5	e6	e7	e8	e9
PJ6M	6	1.641	$\gamma \neq 1$	2	0.6e-26	0.1 e-29	0.2 e-20	0.1e-20	0.6e-20	0.1 e-27	0.4 e-28	0.6 e-22	0.4e-21
PJ6M	6	1.172	$\gamma = 1$	2	3.6e-1	6.4 e-6	9.5 e-3	1.9e-2	1.8e-2	2.0 e-5	2.9 e-5	8.6 e-3	2.5e-2
PJ10D	10	0.766	$\gamma = 1$	2	0.6e-26	0.1 e-29	0.2 e-20	0.1e-20	0.6e-20	0.1 e-27	0.4 e-28	0.6 e-22	0.4e-21
MMN8M	8	0.234	$\gamma \neq 1$	2	0.6e-25	0.2 e-36	0.4 e-18	0.7e-19	0.2e-21	0.1 e-28	0.1 e-30	0.1 e-24	0.4e-18
MMN8D	8	0.156	$\gamma = 1$	2	9.7e-31	1.5 e-37	5.12 e-24	1.2e-26	2.0e-22	1.4 e-38	7.6 e-35	4.0 e-27	8.1e-21

Example4.2[15]**:** Consider

$$f(x) = (x+1)^{2} (x+2)^{3} (x^{2}-2x+2)^{2} (x^{2}+1)^{2} (x-2)^{3} (x+2i)^{2},$$

with multiple exact roots ($\gamma \neq 1$):

$$\xi_1 = -1,\ \xi_2 = -2,\ \xi_{3,4} = -2 \pm i,\ \xi_{5,6} = \pm i,\ \xi_7 = 2,\ \xi_8 = -2 + i.$$

The initial approximations valve have been taken as:

$\overset{(0)}{x}_1 = -1.3 + .2i,\ \overset{(0)}{x}_2 = -2.2 - .3i,\ \overset{(0)}{x}_3 = 1.3 + 1.2i,\ \overset{(0)}{x}_4 = 0.7 - 1.2i,\ \overset{(0)}{x}_5 = -0.2 + .8i,$
$\overset{(0)}{x}_6 = 0.2 - 1.3i,\ \overset{(0)}{x}_7 = 2.2 - .3i,\ \overset{(0)}{x}_8 = -2.2 + 0.7i.$

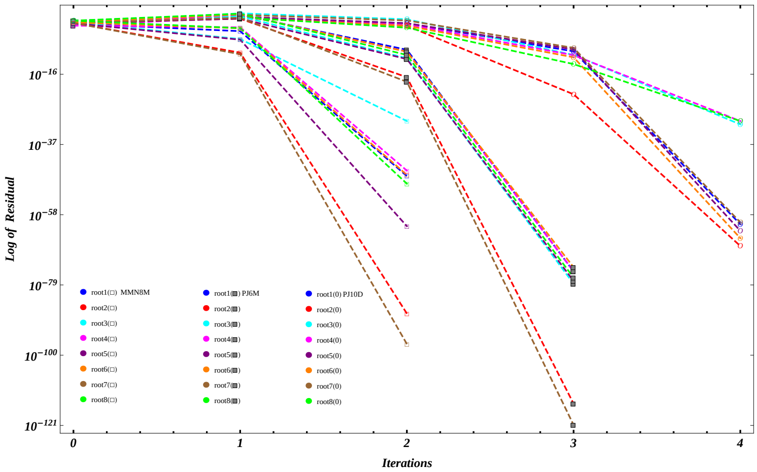

FIGURE 3. Residual fall for Example 4.2

For distinct roots ($\gamma = 1$):

$$f(x) = (x+1)\,(x+2)\,(x^{2}-2\,x+2)\,(x^2+1)\,(x-2)\,(x+2i).$$

Table.4.2(a)

Method	CO	CPU	γ	n	e1	e2	e3	e4	e5	e6	e7	e8
PJ6M	6	0.922	$\gamma \neq 1$	2	2.6e-7	1.8e-7	5.5e-0	1.1e-7	8.4e-9	8.0e-8	1.2e-7	1.0e-7
PJ6M	6	1.078	$\gamma=1$	2	2.0e-9	1.2e-7	6.3e-12	5.7e-10	3.4e-12	5.4e-10	5.4e-19	5.5e-11
PJ10D	10	0.594	$\gamma=1$	2	1.2e-1	3.0e-2	2.7e-1	7.9e-2	1.8e-1	3.1e-2	1.2e-1	8.2e-3
MMN8M	8	0.250	$\gamma \neq 1$	2	3.5e-47	2.7e-88	0.0	1.4e-45	3.7e-62	7.7e-47	2.2e-97	1.8e-49
MMN8D	8	0.156	$\gamma = 1$	2	2.6e-35	3.8e-36	0.0	0.0	5.1e-38	4.6e-32	1.9e-37	0.0

Table.4.2(b)

Method	CO	CPU	γ	n	e1	e2	e3	e4	e5	e6	e7	e8
PJ6M	6	1.047	$\gamma \neq 1$	2	6.2e-7	4.3e-8	8.9e-9	7.0e-8	1.3e-8	1.1e-7	8.2e-9	1.0e-8
PJ6M	6	0.906	$\gamma=1$	2	6.2e-9	9.4e-10	4.1e-12	7.4e-10	1.3e-10	8.9e-10	2.5e-10	2.9e-10
PJ10D	10	0.578	$\gamma=1$	2	3.9e-3	1.5e-4	2.1e-1	4.9e-4	2.8e-3	3.5e-4	2.4e-3	2.3e-4
MMN8M	8	0.203	$\gamma \neq 1$	2	0.8e-25	0.2e-30	0.1e-32	0.1e-24	0.2e-32	0.4e-25	0.5e-33	0.6e-27
MMN8D	8	0.156	$\gamma = 1$	2	6.6e-37	2.0e-38	1.1e-39	3.6e-34	1.6e-39	5.2e-34	4.2e-42	5.4e-37

Example4.3[4]**:** Consider

$$f(x) = (e^{\,x\,(x-1)\,(x-2)\,(x-3)} - 1)^{\,4},$$

with multiple exact roots ($\gamma \neq 1$):

$$\xi_1 = 0,\ \xi_2 = 1,\ \xi_3 = 2,\ \xi_4 = 3.$$

The initial approximations have been taken as:

$$\overset{(0)}{x_1} = 0.1,\ \ \overset{(0)}{x_2} = 0.9,\ \ \overset{(0)}{x}_{\,3} = 1.8,\ \ \overset{(0)}{x}_{\,4} = 2.9,$$

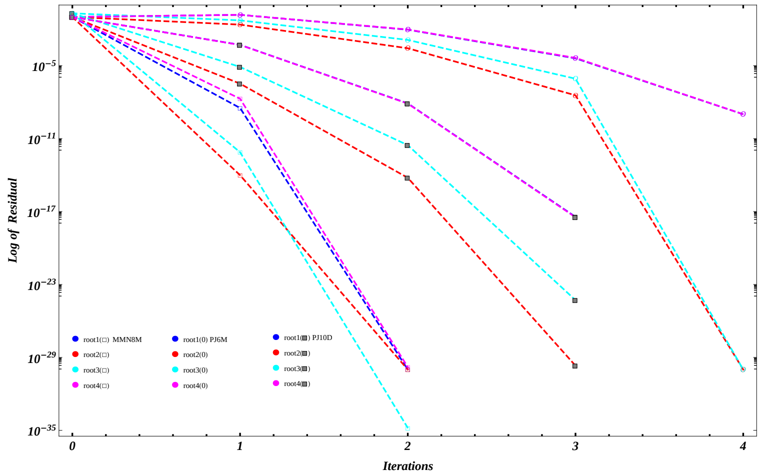

FIGURE 4. Residual fall for Example 4.3

For distinct roots ($\gamma = 1$):

$$f(x) = (e^{x\,(x-1)\,(x-2)\,(x-3)} - 1).$$

Table.4.3(a)

Method	CO	CPU	γ	n	e1	e2	e3	e4
PJ6M	6	0.234	$\gamma \neq 1$	2	7.7e-9	2.6e-4	1.1e-3	9.3e-3
PJ6M	6	0.156	γ=1	2	9.3e-3	2.5e-4	2.0e-3	9.3e-3
PJ10D	10	0.125	γ=1	2	9.3e-3	2.7e-4	1.2e-3	9.3e-3
MMN8M	8	0.062	$\gamma \neq 1$	2	0.0	0.0	0.0	0.0
MMN8D	8	0.062	γ=1	2	1.1e-9	0.0	0.0	1.1e-9

Table.4.3(b)

Method	CO	CPU	γ	n	e1	e2	e3	e4
PJ6M	6	0.176	$\gamma \neq 1$	2	1.5e-3	1.3e-4	6.5e-4	1.5e-3
PJ6M	6	0.110	γ=1	2	1.5e-3	1.3e-4	6.3e-4	1.5e-3
PJ10D	10	0.094	γ=1	2	1.5e-3	1.3e-4	6.5e-4	1.5e-3
MMN8M	8	0.063	$\gamma \neq 1$	2	6.6e-11	6.0e-16	4.7e-14	3.8e-10
MMN8D	8	0.062	γ=1	2	1.7e-10	5.0e-16	2.4e-13	1.5e-10

Example 4.4[17]**:** Consider

$$f(x) = (x^3+5x^2-4x-20+\cos(x^3+5x^2-4x-20)-1)^5,$$

with multiple exact roots ($\gamma \neq 1$):

$$\xi_1 = -5,\ \xi_2 = -2,\ \xi_3 = 2,$$

The initial approximations have been taken as:

$$x_1^{(0)} = -5.1,\ \ x_2^{(0)} = -1.8,\ \ x_3^{(0)} = 1.9.$$

For distinct roots ($\gamma = 1$):

$$f(x) = (x^3+5x^2-4x-20+\cos(x^3+5x^2-4x-20)-1).$$

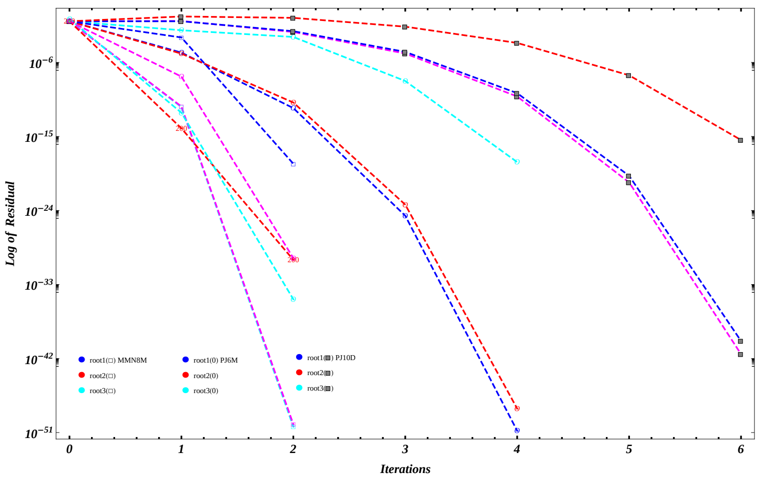

FIGURE 5. Residual fall for Example 4.4.

Table.4.4(a)							
Method	CO	CPU	γ	n	e1	e2	e3
PJ6M	6	0.125	$\gamma \neq 1$	2	2.7e-12	1.3e-11	1.2e-3
PJ6M	6	0.140	γ=1	2	4.9e-3	6.1e-3	2.6e-1
PJ10D	10	0.250	γ=1	2	4.9e-3	6.0e-3	2.5e-1
MMN8M	8	0.141	$\gamma \neq 1$	2	7.2e-11	1.7e-10	2.3e-4
MMN8D	8	0.094	γ=1	2	5.1e-51	1.0e-50	4.8e-19
Table.4.4(b)							
Method	CO	CPU	γ	n	e1	e2	e3
PJ6M	6	0.031	$\gamma \neq 1$	2	2.3e-4	5.5e-4	8.4e-3
PJ6M	6	0.047	$\gamma = 1$	2	2.3e-4	5.1e-4	8.4e-3
PJ10D	10	0.062	$\gamma = 1$	2	2.3e-4	5.5e-4	8.4e-3
MMN8M	8	0.047	$\gamma \neq 1$	2	4.7e-12	8.3e-12	8.3e-6
MMN8D	8	0.016	$\gamma = 1$	2	4.7e-12	8.3e-12	7.7e-6

5. CONCLUSION

In this article, a new two-step eighth order family of iterative methods for the simultaneous approximations of all roots of a polynomial equation was introduced and discussed. Our method MMN8M, determines all the multiple roots and as a special case for multiplicity unity all the distinct roots of polynomial equation. The results of numerical test examples, CPU time, residual error, computational efficiency corroborate theoretical analysis, illustrate the effectiveness and rapid convergence of our proposed family of iterative method as compared to the methods PJ6M and PJ10D [21, 22].

REFERENCES

[1] O. Aberth, *Iteration methods for finding all zeros of a polynomial simultaneously*, Math. Comp. 27(1973) 339-344.

[2] W. Bi, H. Ren and Q. Wu, *Convergence of the modified Halley's method for multiple zeros under Hölder continuous derivative*, Numer. Alg. 58 (2011) 497–512.

[3] S. I. Cholakov and M. T. Vasileva, *A convergence analysis of a fourth-order method for computing all zeros of a polynomial simultaneously*, J. Comput. Appl. Math. 321 (2017) 270–283.

[4] S. I. Cholakov, *Local and semilocal convergence of Wang-Zheng method for simultaneous finding polynomial zeros*, Symmetry, 2019 (2019), Art. 736, 15pp.

[5] C. Chun and B. Neta, *A third-order modification of Newton's method for multiple roots*, Appl. Math. Comput. 211 (2009) 474–479.

[6] M. Cosnard and P. Fraigniaud, *Finding the roots of a polynomial on an MIMD multicomputer,* Parallel Computing, **15,** No.1–3 (1990), 75–85.

[7] M.R. Farmer, *Computing the zeros of polynomials using the Divide and Conquer approach*, Ph.D Thesis, Department of Computer Science and Information Systems, Birkbeck, University of London, July 2014.

[8] H. H. H. Homeier, *On Newton-type methods for multiple roots with cubic convergence*, J. Comput. Appl. Math. 231 (2009) 249–254.

[9] S. I. Ivanov, *A unified semilocal convergence analysis of a family of iterative algorithms for computing all zeros of a polynomial simultaneously*, Numer. Algor., 75 (2017) 1193–1204.

[10] A. I. Iliev and Kh. I. *Semerdzhiev, Some generalizations of the Chebyshev method for simultaneous determination of all roots of polynomial equations,* Comput. Math. Math. Phy. **39**, No. 9 (1999) 1384-1391.

[11] S. I. Ivanov, O*n the convergence of Chebyshev's method for multiple polynomial zeros*, Results. Math., **69** No.1 (2016), 93–103.

[12] S. Kanno N, Kjurkchiev and T. Yamamoto, *On some methods for the simultaneous determination of polynomial zeros*, Japan J. Apple. Math. 13 (1995) 267-288.

[13] V. K. Kyncheva, V. V. Yotov and S. I. Ivanov. *Convergence of Newton, Halley and Chebyshev iterative methods as methods for simultaneous determination of multiple polynomial zeros*, Appl. Numer. Math.,112 (2017) 146-154.

[14] V. K. Kyncheva, V. V. Yotov and S. I. Ivanov. *On the convergence of Sch¨rder's method for the simultaneous computation of polynomial zeros of unknown multiplicity*, Calcolo, **54,** No.4 (2017) 1199–1212,

[15] T.-F. Li, D.-S. Li, Z.-D.Xu and Y.-I. Fang, *New iterative methods for non-linear equations*, Appl. Math. and Comput. (2008) 755-759.

[16] N. A. Mir, R. Muneer and I. Jabeen, *Some families of two-step simultaneous methods for determining zeros of non-linear equations*, ISRN Applied Mathematics, (2011) 1-11.

[17] G. H. Nedzhibov, *Iterative methods for simultaneous computing arbitrary number of multiple zeros of non-linear equations*, Inrern. J. Cornput. Math,, (2013) 994-1007.

[18] A. W. M. Nourein, *An Improvement on two iteration methods for simultaneous determination of the zeros of a polynomial*, Inrern. J. Cornput. Math, (1977) 241-252.

[19] N. Osada, *Asymptotic error constants of cubically convergent zero finding methods*, J. Comput. Appl. Math. 196, (2006), 347–357.

[20] M. P. Petkovic, *Point Estimation of Root Finding Methods, Lecture Notes in Mathematics*, Springer, 1933 Berlin (2008).

[21] M. S. Petkovic, L. D. Petkovic,c and J. D. Zunic, *On an efficient method for simultaneous approximation of polynomial multiple roots*, Appl. Anal. Discrete Math, 8 (2014) 73–94.

[22] M. S. Petkovic and J. Zunic, *On an efficient simultaneous method for finding polynomial zeros*, Appl. Math. Lett, 28 (2014) 60-65.

[23] P. D. Proinov, *On the local convergence of Ehrlich method for numerical computation of polynomial zeros*, Calcolo, **53,** No. 3 (2016) 413–426.

[24] P. D. Proinov and S. I. Cholakov, *Semilocal convergence of Chebyshev-like root-finding method for simultaneous approximation of polynomial zeros*, Appl. Math. Comput. 236(2014) 669-682.

[25] P. D. Proinov and M.T. Vasileva, *On the convergance of family of Weierstrass-type root-finding methods*, C. R. Acad. Bulg. Sci, 68 (2015) 697-704.

[26] P. D. Proinov, *A general semilocal convergence theorem for simultaneous methods for polynomial zeros and its applications to Ehrlich and Dochev-Byrnevs methods*, Appl. Math. Comput. 284 (2016) 102–114.

[27] P. D. Proinov, *Unified convergence analysis for Picard iteration in n-dimensional vector spaces*, Calcolo 55 (2018) Art. 6. 21 pp.

[28] P. D. Proinov, and M. D. Petkova, *A new semilocal convergence theorem for the Weierstrass method for finding zeros of a polynomial simultaneously*, J. Complexity, 30 (2014) 366–380.

[29] P. D. Proinov and M. T. Vasileva, *On the convergence of high-order Gargantini-Farmer-Loizou type iterative methods for simultaneous approximation of polynomial zeros*, Appl. Math. Comput. 361 (2019) 202–214.

[30] P. D. Proinov and S. I. Ivanov. *On the convergence of Halley's method for multiple polynomial zeros*, Mediterranean J. Math., 12 (2015) 555–572.

[31] P. D. Proinov and S. I. Ivanov. *On the convergence of Halley's method for simultaneous computation of polynomial zeros*, J. Numer. Math., **23** No.4 (2015) 379–394.

[32] P. D. Proinov and S. I. Ivanov, M. S. Petkovic. *On the convergence of Gander's type family of iterative methods for simultaneous approxima-tion of polynomial zeros*, Appl. Math. Comput., 349(2019) 168–183.

[33] P. D. Proinov, and S. I. Ivanov, *Convergence analysis of Sakurai–Torii–Sugiura iterative method for simultaneous approximation of polynomial zeros*, J. Comp. Appl. Math., 357 (2019) 56–70.

[34] Bl. Sendov, A. Andereev and N. Kjurkchiev, *Numerical Solutions of Polynomial Equations*, Elsevier science, New York, 1994.

[35] X. Wang and L. Liu, *Modified Ostrowski method with eight order convergence and high efficiency index,* Appl. Math. Lett., 23 (2010) 549-554.

[36] Weierstrass and K Neuer Beweis des Satzes, *dass jede ganze rationale Function einer Veränderlichen dargestellt werden kann als ein Product aus linearen Functionen derselben Veränderlichen*, Sitzungsber. Königl. Preuss. Akad. Wiss. Berlinn II, (1891) 1085–1101.