Hindawi
Journal of Mathematics
Volume 2022, Article ID 4327913, 20 pages
https://doi.org/10.1155/2022/4327913

Research Article

A Comparative Study on Qualification Criteria of Nonlinear Solvers with Introducing Some New Ones

R. H. AL-Obaidi and M. T. Darvishi

Department of Mathematics, Faculty of Science, Razi University, Kermanshah 67149, Iran

Correspondence should be addressed to M. T. Darvishi; darvishimt@yahoo.com

Received 12 June 2022; Revised 19 August 2022; Accepted 2 September 2022; Published 21 September 2022

Academic Editor: Predrag S. Stanimirović

In order to compare different solvers for systems of nonlinear equations, some novel goodness and qualification criteria are defined in this paper. These use all parameters of a nonlinear solver such as convergence order, number of function evaluations, number of iterations, CPU time, etc. To achieve the criteria, different algorithms to solve nonlinear systems are categorised to three kinds. For any category, two criteria are defined to compare different algorithms in that category. As numerical results show, these new criteria can use to compare different algorithms which solve systems of nonlinear equations. Further, we present some corrected formulas for some classical efficiency indices and change them to be more applicable. Also, some suggestions are presented about the future works.

1. Introduction

One of the attractive problems for mathematicians, engineers, and physicians is finding a (the) root of systems of nonlinear equations. A system of nonlinear equations can be written as the following generic form:

$$F(\mathrm{x}) = 0, \tag{1}$$

where $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is a vector valued function. It is known that in general, one cannot solve (1) analytically. Hence we have to solve it numerically or more precise by an iterative method. Even though the classical Newton's method to solve (1) is a basic iterative method and widely used, but many new iterative methods are also introduced every year. It is well-known that the Newton's method is a second-order method if Jacobian of F in (1) is nonsingular in a neighborhood of the solution. To show superiority of new methods versus the Newton's method, different criteria have been used. Some authors use order of convergence, some use number of iterations to achieve convergence, some use CPU time, or a combination of some of these mentioned criteria. To remove the weakness of these criteria, for algorithms which solve nonlinear systems like (1), it is important to take into account the number of required operations, because in any iteration one must solve some linear systems. For these cases the classical efficiency index (EI) is defined by Ostrowski [1] as

$$\mathrm{EI} = p^{1/d}, \tag{2}$$

where p is the order of convergence and $d = mn^2 + rn$ wherein n is the size of the system, r is the number of functional evaluations per step, and m shows the number of Jacobian evaluations per step. Further, Traub [2] introduced the operational index as

$$I_{op} = p^{1/op}, \tag{3}$$

where op shows the number of required products or quotients per iteration.

Except these criteria, another ones were defined by some researchers for example Grau A and Grau N [3] defined the computational cost per iteration as $\mathscr{C}(\alpha, \beta, n) =$

$\alpha kn + \beta rn^2 + P(n)$, where k and r represent the number of evaluations of $F(x)$ and its Jacobian matrix, respectively. Besides, α and β are the ratios between multiplications and evaluations required to express the value of $\mathscr{C}(\alpha, \beta, n)$ in terms of multiplications. Also, $P(n)$ is the number of products in any iteration. By these notations, they introduced the computational efficiency index (CEI) as

$$\mathrm{CEI} = p^{1/\mathscr{C}(\alpha,\beta,n)}, \tag{4}$$

where p is the order of convergence. It must be noted that, CEI is an extension of EI, to see this extension, it is enough that one set $\alpha = \beta = 1$, and $P(n) = 0$.

Also, the flops-like efficiency index (FLEI) is defined by Montazeri et al. [4] as

$$\mathrm{FLEI} = p^{1/c}, \tag{5}$$

where p is the order of convergence of the method, $c = A + Bn^2 + Cn$ which A shows the cost of LU factorization (based on the flops). Also, n and C, respectively, are the size of the system and the number of functional evaluations per step. Finally, B is the number of Jacobian evaluations in any step.

The above mentioned criteria are useful but for two different reasons, they cannot express superiority of an algorithm solely. The different categories of nonlinear solvers are the first reason. Existing different parameters in any algorithm is the second one. Further, computing the values of parameters are not easy and straight forward. Therefore, there are some important gaps for these existing known criteria in the literature, I_{op}, CEI, and FLEI. In this paper, we introduce some new qualification criteria to remove these lacks. For example, for algorithms which their order of convergence is known, we present a criterion which contains order of convergence, CPU time, number of iterations, number of function evaluations, etc. These parameters have two distinguished roles; the positive and the negative ones. For example, order of convergence shows a positive index for any algorithm while CPU time presents a negative one. Hence our criterion can have a fractional form as

$$\frac{\text{Order of Convergence}}{\text{A Combination of Negative Parameters}}. \tag{6}$$

Further, we categorise different algorithms and for any kind we introduce two relevant criteria to compare them altogether. Finally, we present some corrected formulas for the classical efficiency indices, I_{op}, CEI, and FLEI and change them to be more applicable.

1.1. Test Problems. To investigate qualification of nonlinear solvers in any category, we applied them to solve four systems of nonlinear equations. These are selected form different areas which need solving of a nonlinear system. These are moderately simple nonlinear systems such that any algorithms can solve them. As a matter of fact, we only want to compare the considered algorithms; therefore these test problems were selected. Besides, all of them at least have one known exact solution.

Test Problem 1. Suppose that $F(\mathrm{x}) = (f_1(\mathrm{x}), f_2(\mathrm{x}), \cdots, f_n(\mathrm{x}))^t$ with

$$\begin{aligned} f_i(\mathrm{x}) &= x_i x_{i+1} x_{i+2} - 1, \quad i = 1, 2, \cdots, n-2, \\ f_{n-1}(\mathrm{x}) &= x_{n-1} x_n x_1 - 1, \\ f_n(\mathrm{x}) &= x_n x_1 x_2 - 1. \end{aligned} \tag{7}$$

The exact zero of $F(\mathrm{x}) = 0$ is $(1, 1, \cdots, 1)^t$. To solve Test Problem 1, we set the initial guess as $(1.5,1.5, \cdots, 1.5)^t$.

Test Problem 2. Consider the following trigonometric system of n nonlinear equations:

$$F(x) = (f_1(x), f_2(x), \cdots, f_n(x))^t, \tag{8}$$

with $f_i(x) = \cos(x_i) - 1$, $i = 1, 2, \cdots, n$. One of the exact roots of system $F(x) = 0$ is $(0, 0, \cdots, 0)^t$. To solve Test Problem 2, the initial guess is selected as $(0.87,0.87, \cdots, 0.87)^t$.

Test Problem 3. The third test problem is $F(x) = (f_1(x), f_2(x), \cdots, f_n(x))^t$ with

$$f_i(x) = n(x_i - 3)^2 + \frac{\cos(x_i - 3)}{2} - \frac{(x_i - 2)}{\exp(x_i - 3) + \log(x_i^2 + 1)}, i = 1, 2, \cdots, n. \tag{9}$$

The exact zero of $F(x) = 0$ is $(3, 3, \cdots, 3)^t$. Here, the initial guess is $(-3,-3, \cdots,-3)^t$.

Test Problem 4. The last system is $F(x) = 0$, where $F(x) = (f_1(x), f_2(x), \cdots, f_n(x))^t$ wherein

$$f_i(\mathrm{x}) = (1 - x_i^2) + x_i(1 + x_i x_{n-2} x_{n-1} x_n) - 2, i = 1, 2, \cdots, n. \tag{10}$$

The exact zero of this system is $(1, 1, \cdots, 1)^t$. To solve Test Problem 4, we set the initial guess as $(2, 2, \cdots, 2)^t$.

2. First Category

The first category of the nonlinear solvers contains algorithms which use Jacobian evaluation, and their order of convergence is known. The parameters of these solvers are order of convergence (OC) which is an important index for the method and in the investigation of quality of the method has a positive role, number of iterations (IT), number of total function evaluations (FE), total number of Jacobian evaluations (JE), and CPU time (CPU) that have negative roles. The most common criteria for these algorithms can be written as follows:

(i) Criterion 1:

$$\mathrm{CR1} = \frac{\mathrm{OC}}{\mathrm{CPU} + \mathrm{IT} * (\mathrm{FE} + \mathrm{JE})}. \tag{11}$$

(ii) Criterion 2:

$$\mathrm{CR2} = \frac{\mathrm{OC}}{\mathrm{CPU} + \mathrm{IT} + \mathrm{FE} + \mathrm{JE}}. \tag{12}$$

2.1. Some Selected Iterative Methods. To show goodness of our criteria, we have chosen some different iterative methods which are presented in the last two decades. We have computed values of both criteria to compare qualification of these iterative methods. Bigger qualification value of any criterion shows the better qualification of the solver. These methods are selected from different categories which use Jacobian of function F. We use them to solve Test Problems 1-4.

Method 1: The classical Newton's method (NM) is a quadratic order convergent one which has the following formula:

$$x^{(k+1)} = x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \tag{13}$$

where $J_F(\cdot)$ shows the Jacobian matrix of function F.

Method 2 (M2): Darvishi and Barati [5] introduced the following third order convergent method in (2007):

$$\begin{aligned} y^{(k)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ x^{(k+1)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1}\left(F\left(x^{(k)}\right) + F\left(y^{(k)}\right)\right). \end{aligned} \tag{14}$$

Method 3 (M3): The following super cubic convergent method is introduced by Darvishi and Barati [6] in (2007). Later, Babajee et al. [7] proved that the method is in fact a fourth order convergent method.

$$\begin{aligned} y^{(k)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ x^{(k+1)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right) - J_F\left(y^{(k)}\right)^{-1} F\left(y^{(k)}\right). \end{aligned} \tag{15}$$

Method 4 (M4): Further, a fourth order convergent method is introduced by Darvishi and Barati [8] in (2007) as

$$\begin{aligned} y^{(k)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ z^{(k)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1}\left(F\left(x^{(k)}\right) + F\left(y^{(k)}\right)\right),\\ x^{(k+1)} &= x^{(k)} - \left[\frac{1}{6}J_F\left(x^{(k)}\right) + \frac{2}{3}J_F\left(\frac{x^{(k)} + z^{(k)}}{2}\right) + \frac{1}{6}J_F\left(z^{(k)}\right)\right]^{-1} F\left(x^{(k)}\right). \end{aligned} \tag{16}$$

Method 5 (M5): Soleymani et al. [9] defined the following fourth order convergent method in (2013):

$$\begin{aligned} y^{(k)} &= x^{(k)} - \frac{2}{3}J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ x^{(k+1)} &= x^{(k)} - \frac{1}{2}\left(3J_F\left(y^{(k)}\right) - J_F\left(x^{(k)}\right)\right)^{-1}\\ &\quad \cdot \left(3J_F\left(y^{(k)}\right) + J_F\left(x^{(k)}\right)\right) \times J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right). \end{aligned} \tag{17}$$

Method 6 (M6): The following cubic convergent method is introduced by Frontini and Sormani [10] in (2004):

$$\begin{aligned} y^{(k)} &= x^{(k)} - \frac{1}{2}J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ x^{(k+1)} &= x^{(k)} - J_F\left(y^{(k)}\right)^{-1} F\left(x^{(k)}\right). \end{aligned} \tag{18}$$

Method 7(M7): Cordero et al. [11] presented the following fifth order three-step method in (2012):

$$\begin{aligned} \mathrm{y}^{(k)} &= \mathrm{x}^{(k)} - J_F\left(\mathrm{x}^{(k)}\right)^{-1} F\left(\mathrm{x}^{(k)}\right),\\ \mathrm{z}^{(k)} &= \mathrm{x}^{(k)} - 2\left(J_F\left(\mathrm{x}^{(k)}\right) + J_F\left(\mathrm{y}^{(k)}\right)\right)^{-1} F\left(\mathrm{x}^{(k)}\right),\\ \mathrm{x}^{(k+1)} &= \mathrm{z}^{(k)} - J_F\left(\mathrm{y}^{(k)}\right)^{-1} F\left(\mathrm{z}^{(k)}\right). \end{aligned} \tag{19}$$

Method 8(M8): In (2016) Xiao and Yin [12] introduced a three-step, fifth order convergent method as

$$\begin{aligned} y^{(k)} &= x^{(k)} + J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ z^{(k)} &= y^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right),\\ x^{(k+1)} &= z^{(k)} + \left(J_F\left(y^{(k)}\right)^{-1} - 2J_F\left(x^{(k)}\right)^{-1}\right) F\left(z^{(k)}\right). \end{aligned} \tag{20}$$

Method 9 (M9): Soleymani et al. [9] in (2014) presented the following three-step method which is a sixth order convergent one:

$$\begin{aligned} y^{(k)} &= x^{(k)} - \frac{2}{3}J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ z^{(k)} &= x^{(k)} - \frac{1}{2}\left(3J_F\left(y^{(k)}\right) - J_F\left(x^{(k)}\right)\right)^{-1}\\ &\quad \cdot \left(3J_F\left(y^{(k)}\right) + J_F\left(x^{(k)}\right)\right) \times J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right),\\ x^{(k+1)} &= z^{(k)} - \left(\frac{1}{2}\left(3J_F\left(y^{(k)}\right) - J_F\left(x^{(k)}\right)\right)^{-1}\right.\\ &\quad \left.\cdot \left(3J_F\left(y^{(k)}\right) + J_F\left(x^{(k)}\right)\right)\right)^{2} \times J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right). \end{aligned} \tag{21}$$

Table 1: Values of CR1 for methods 1-10 to solve all test problems, and $n = 100$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.1907	0.1192	0.0636	0.1364
M2	0.3135	0.1564	0.0852	0.1880
M3	0.3181	0.2386	0.1194	0.2387
M4	0.2440	0.1831	0.0916	0.1832
M5	0.5635	0.2833	0.1410	0.2825
M6	0.3121	0.1885	0.0941	0.1883
M7	0.5704	0.2865	0.1637	0.2866
M8	0.3820	0.2292	0.1433	0.2860
M9	0.6619	0.3288	0.2198	0.4400
M10	0.6333	0.3176	0.1818	0.3169

Method 10 (M10): As the last method, we investigate qualification criteria for the following sixth order convergent iterative method which is introduced by Lotfi et al. [13] in (2015):

$$\begin{aligned} y^{(k)} &= x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ z^{(k)} &= x^{(k)} - 2\left(J_F\left(x^{(k)}\right) + J_F\left(y^{(k)}\right)\right)^{-1} F\left(x^{(k)}\right), \\ x^{(k+1)} &= z^{(k)} - \left[\frac{7}{2}I - 4J_F\left(x^{(k)}\right)^{-1} J_F\left(y^{(k)}\right) + \frac{3}{2}\left(J_F\left(x^{(k)}\right)^{-1} J_F\left(y^{(k)}\right)\right)^2\right] \\ &\quad \times J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right). \end{aligned} \tag{22}$$

2.2. Numerical Results. As a matter of fact, our aim is showing superiority of different methods on the classical Newton's one. Hence any method which has better results for the criteria will be better than the Newton's method. In fact, we would like to test the mentioned methods M1-M10 to pass or fail our examinations from the criteria. Any method which has small values for a criterion in respect with the Newton's method is failed from this evaluation. Also, we can compare the methods altogether. All test problems are solved for two different values of n, namely, $n = 100{,}500$. The numerical results are reported for these cases as follows.

Case $n = 100$. Tables 1 and 2, respectively, show the values of $CR1$ and $CR2$ to solve all test problems by iterative Methods 1-10. From Tables 1 and 2, M9 has bigger values in comparing with the other methods. Hence for case $n = 100$, M9 acts better than the other ones.

Case $n = 500$. Values of our criteria, for $n = 500$ are reported in Tables 3 and 4 for test problems 1-4, respectively. We can see from Tables 3, and 4 that M3 is better than the other methods for solving all test problems in both criteria CR1 and CR2, clearly. From these tables, we conclude that by increasing size of the system, M3 acts better than the other methods.

2.3. Discussion on Numerical Results for the First Category. As we can see from Tables 1–4 four methods, namely, M3, M7, M9, and M10 have better results among all methods. Hence, we compared those methods altogether for different sizes of our problems. As a matter of fact, we investigated the qualification of these methods by increasing n. The results are reported only for Test Problem 1. Plots of Figure 1 shows the results graphically. As one can see from these plots, even though methods M3, M7, and M9 have better results for small n but by increasing the size of the problem, the third order method M3 has better results. It must be

Table 2: Values of CR2 for methods 1-10 to solve all test problems, and $n = 100$.

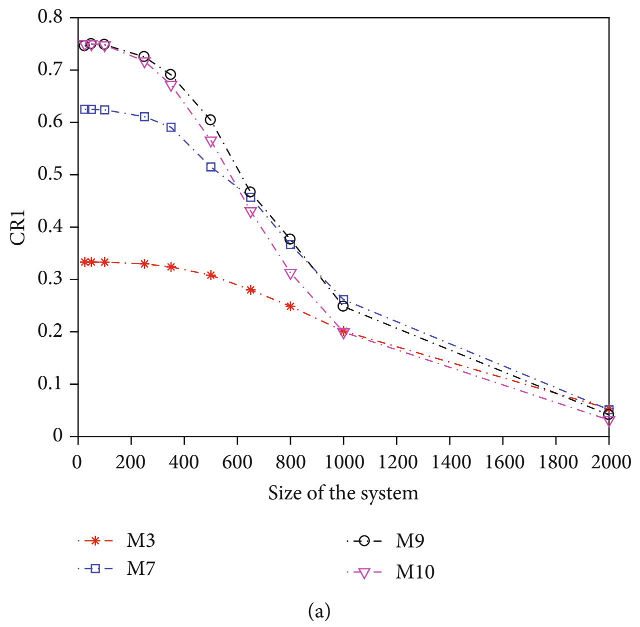
Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.2670	0.1856	0.1084	0.2070
M2	0.4567	0.2947	0.1848	0.3350
M3	0.5281	0.4562	0.2960	0.4566
M4	0.4257	0.3688	0.2397	0.3692
M5	0.6558	0.4387	0.2601	0.4367
M6	0.4537	0.3366	0.2015	0.3360
M7	0.7390	0.5291	0.3692	0.5293
M8	0.6182	0.4623	0.3358	0.5272
M9	0.8492	0.5853	0.4514	0.6948
M10	0.8028	0.5509	0.3748	0.5486

Table 3: Values of CR1 for methods 1-10 to solve all test problems, and $n = 500$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.0341	0.0188	0.0098	0.0243
M2	0.0446	0.0222	0.0111	0.0262
M3	0.0566	0.0339	0.0190	0.0418
M4	0.0251	0.0150	0.0083	0.0183
M5	0.0217	0.0131	0.0073	0.0164
M6	0.0446	0.0224	0.0122	0.0268
M7	0.0452	0.0286	0.0145	0.0292
M8	0.0389	0.0228	0.0116	0.0255
M9	0.0484	0.0234	0.0137	0.0293
M10	0.0365	0.0179	0.0091	0.0183

Table 4: Values of CR2 for methods 1-10 to solve all test problems, and $n = 500$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.0360	0.0201	0.0106	0.0258
M2	0.0467	0.0238	0.0120	0.0279
M3	0.0609	0.0374	0.0213	0.0456
M4	0.0263	0.0159	0.0088	0.0192
M5	0.0221	0.0134	0.0075	0.0167
M6	0.0467	0.0240	0.0132	0.0285
M7	0.0461	0.0300	0.0154	0.0307
M8	0.0405	0.0240	0.0123	0.0266
M9	0.0492	0.0242	0.0143	0.0301
M10	0.0370	0.0183	0.0094	0.0188



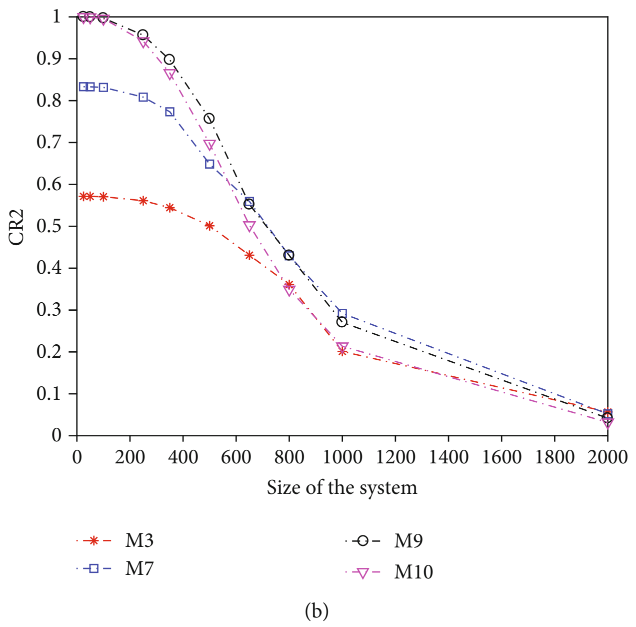


Figure 1: Values of CR1 (a) and CR2 (b) for solving Test Problem 1 by algorithms M3, M7, M9, and M10.

noted that, CPU time to compute these results is computed in hours while the other results were obtained for CPU time in minutes. As a conclusion on the first category, we claim that the order of convergence cannot show the quality of a nonlinear solver. In fact as our results show, even though for small values of n, high order methods can be used to solve nonlinear systems, but for large values of the size of the problem, the moderate convergence method, namely M3 acts better than the high order convergence methods. Therefore for the first category, we can apply high order method M9 for small values of n. But for large values of n, the cubic order method M3 is the best solver. Finally, any new method in this category must be compared with M9 for small n while for large n any comparison can be done with M3.

2.4. Some Other Criteria for the First Category. It must be noted that in our investigation, to introduce a new criterion, we tested many different formulas, except CR1 and CR2. Some other criteria which can be useful in future researches are as follows:

(i) Criterion 3: The third criterion is defined as

$$\mathrm{CR3} = \frac{\mathrm{OC}}{\mathrm{CPU} * \mathrm{IT} * (\mathrm{FE} + \mathrm{JE})}. \tag{23}$$

(ii) Criterion 4: To give a big weight to the order of convergence, the next criterion receives the following formula:

$$\mathrm{CR4} = \frac{2^{\mathrm{OC}}}{\mathrm{CPU} * \mathrm{IT} * (\mathrm{FE} + \mathrm{JE})}. \tag{24}$$

There are similar interpretations for the other criteria.

(iii) Criterion 5:

$$\mathrm{CR5} = \frac{2^{\mathrm{OC}}}{\mathrm{CPU} + \mathrm{IT} + (\mathrm{FE} + \mathrm{JE})}. \tag{25}$$

(iv) Criterion 6:

$$\mathrm{CR6} = \frac{\mathrm{OC}}{(\mathrm{CPU}/2) + \mathrm{FE} + (\mathrm{IT} * \mathrm{JE})}. \tag{26}$$

(v) Criterion 7:

$$\mathrm{CR7} = \frac{\mathrm{OC}}{(\mathrm{CPU}/OC) + \mathrm{FE} + (\mathrm{IT} * \mathrm{JE})}. \tag{27}$$

(vi) Criterion 8: To decrease influence of the order of convergence, we use the following:

$$\mathrm{CR8}=\frac{\ln(\mathrm{OC})}{\mathrm{CPU}+\mathrm{FE}+\mathrm{IT}+\mathrm{JE}}\cdot \tag{28}$$

(vii) Criterion 9:

$$\mathrm{CR9}=\frac{\mathrm{OC}}{\mathrm{CPU}+\mathrm{FE}+(\mathrm{IT}*\mathrm{JE})}\cdot \tag{29}$$

3. Second Category: Jacobian Free Methods

In this category, we consider Jacobian free methods which apply to solve systems of nonlinear equations and compare them by introducing two qualification criteria. In this category, our parameters are: order of convergence (OC) or computational order of convergence (COC), number of iteration(IT), total number of function evaluations(FE), total number of linear operations evaluations (LOE), number of LU-decomposition (LU), and CPU time(CPU). It must be noted that for some iterative methods, we cannot compute their order of convergence, instead we use the computational order of convergence [14] which is computed as follows:

$$\mathrm{COC}=\frac{\ln\left(\left\|x^{(k+1)}-x^{(k)}\right\|_\infty/\left\|x^{(k)}-x^{(k-1)}\right\|_\infty\right)}{\ln\left(\left\|x^{(k)}-x^{(k-1)}\right\|_\infty/\left\|x^{(k-1)}-x^{(k-2)}\right\|_\infty\right)}. \tag{30}$$

Hence for the second category, we define our criteria as follows:

(i) Jacobian Free Criterion 1 (JFC1): The first criterion is defined as

$$\mathrm{JFC1}=\frac{\mathrm{COC}}{\mathrm{CPU}+\mathrm{IT}+\mathrm{LU}+\mathrm{FE}+\mathrm{LOE}}\cdot \tag{31}$$

(ii) Jacobian Free Criterion 2 (JFC2): The second criterion is defined as

$$\mathrm{JFC2}=\frac{\mathrm{COC}}{\mathrm{CPU}+\mathrm{IT}*(\mathrm{LU}+\mathrm{FE}+\mathrm{LOE})}\cdot \tag{32}$$

3.1. Some Selected Iterative Methods

3.1. Some Selected Iterative Methods. To show usefulness of our criteria, we have chosen some different Jacobian free methods which are presented in different years. We have applied both criteria (31) and (32) to compare qualification of the following iterative methods.

Jacobian Free Newton Method (JFNM): The Jacobian free version of the classical Newton's method (with $OC = 2$) has the following form:

$$x^{(k+1)}=x^{(k)}-[w,x;F]^{-1}F\left(x^{(k)}\right), \tag{33}$$

where the Jacobian matrix $J_F(x^{(k)})$ has replaced by the linear operator $[w,x;F]$ *where* $x=x^{(k)}$ *and* $w=x^{(k)}+F(x^{(k)})$ satisfy in

$$[w,x;F](w-x)=F(w)-F(x). \tag{34}$$

To compute the elements of (34), we can use the classical first-order divided difference operator (cf. [15])

$$[w,x;F]_{i,j}=\frac{\left(F_i\left(w_1,w_2,\cdots,w_{j-1},w_j,x_{j+1},\cdots,x_m\right)-F_i\left(w_1,w_2,\cdots,w_{j-1},x_j,x_{j+1},\cdots,x_m\right)\right)}{\left(w_j-x_j\right)}, \tag{35}$$

where $x=(x_1,\cdots,x_{j-1},x_j,x_{j+1},\cdots,x_m)$, $w=(w_1,\cdots,w_{j-1},w_j,w_{j+1},\cdots,w_m)$, and $1\le i\le j\le m$.

Jacobian Free Method 2 (JFM2): This method is presented by Amiri et al. [16] in 2018, with $OC/COC = 3$ and has the following form:

$$\begin{aligned} y^{(k)}&=x^{(k)}-[w,x;F]^{-1}F\left(x^{(k)}\right),\\ x^{(k+1)}&=y^{(k)}-\left[2[x,y;F]-[w,x;F]\right]^{-1}F\left(y^{(k)}\right). \end{aligned} \tag{36}$$

Jacobian Free Method 3 (JFM3): Authors: Chicharro et al. [17]; Published year: 2020; OC/COC = 3:

$$\begin{aligned} y^{(k)}&=x^{(k)}-[w,x;F]^{-1}F\left(x^{(k)}\right),\\ x^{(k+1)}&=y^{(k)}-[w,y;F]^{-1}F\left(y^{(k)}\right). \end{aligned} \tag{37}$$

Jacobian Free Method 4(JFM4): Authors: Sharma and Arora [18]; Published year: 2013; OC/COC = 4:

$$\begin{aligned} y^{(k)}&=x^{(k)}-[w,x;F]^{-1}F\left(x^{(k)}\right),\\ x^{(k+1)}&=y^{(k)}-\left(3I-[w,x;F]^{-1}([y,x;F]+[y,w;F])\right)\\ &\quad\times[w,x;F]^{-1}F\left(y^{(k)}\right). \end{aligned} \tag{38}$$

Jacobian Free Method 5 (JFM5): Authors: Cordero et al. [19]; Published year: 2019; OC/COC = 5:

$$\begin{aligned} y^{(k)}&=x^{(k)}-[a,b;F]^{-1}F\left(x^{(k)}\right),\\ z^{(k)}&=y^{(k)}-\alpha[a,b;F]^{-1}F\left(y^{(k)}\right),\\ t^{(k)}&=z^{(k)}-\beta[a,b;F]^{-1}F\left(y^{(k)}\right),\\ x^{(k+1)}&=z^{(k)}-\gamma[a,b;F]^{-1}F\left(t^{(k)}\right), \end{aligned} \tag{39}$$

where $a=x^{(k)}+F(x^{(k)}), b=x^{(k)}-F(x^{(k)})$ and $\alpha=(2-\gamma)$, $\beta=(\gamma-1)^2/\gamma, \gamma=1/5$.

Jacobian Free Method 6 (JFM6): Authors: Wang and Fan [20]; Published year: 2016; OC/COC = 6:

$$y^{(k)} = x^{(k)} - [a, b\,; F]^{-1} F\left(x^{(k)}\right),$$
$$z^{(k)} = y^{(k)} - \left[3I - 2[a, b\,; F]^{-1}[y, x\,; F]\right][a, b\,; F]^{-1} F\left(y^{(k)}\right),$$
$$x^{(k+1)} = z^{(k)} - \left[3I - 2[a, b\,; F]^{-1}[y, x\,; F]\right][a, b\,; F]^{-1} F\left(z^{(k)}\right), \tag{40}$$

where $a = x^{(k)} + F(x^{(k)})$, $b = x^{(k)} - F(x^{(k)})$.

Jacobian Free Method 7 (JFM7): Authors: Wang and Zhang [21]; Published year: 2013; OC/COC = 7:

$$y^{(k)} = x^{(k)} - [w, x\,; F]^{-1} F\left(x^{(k)}\right),$$
$$z^{(k)} = y^{(k)} - [y, x\,; F]^{-1}[[y, x\,; F] - [w, y\,; F] + [w, x\,; F]] \times [y, x\,; F]^{-1} F\left(y^{(k)}\right),$$
$$x^{(k+1)} = z^{(k)} - [[z, x\,; F] + [z, y\,; F] - [y, x\,; F]]^{-1} \times F\left(z^{(k)}\right). \tag{41}$$

Jacobian Free Method 8 (JFM8): Authors: Wang and Zhang [21]; Published year: 2013; OC/COC = 7:

$$y^{(k)} = x^{(k)} - [w, x\,; F]^{-1} F\left(x^{(k)}\right),$$
$$z^{(k)} = y^{(k)} - [[y, x\,; F] + [y, w\,; F] - [w, x\,; F]]^{-1} \times F\left(y^{(k)}\right),$$
$$x^{(k+1)} = z^{(k)} - [[z, x\,; F] + [z, y\,; F] - [y, x\,; F]]^{-1} \times F\left(z^{(k)}\right). \tag{42}$$

Jacobian Free Method 9 (JFM9): Authors: Wang et al. [22]; Published year: 2015; OC/COC = 7:

$$y^{(k)} = x^{(k)} - A^{-1} F\left(x^{(k)}\right),$$
$$z^{(k)} = y^{(k)} - \left[3I - 2A^{-1}[y, x\,; F]\right] \times A^{-1} F\left(y^{(k)}\right),$$
$$x^{(k+1)} = z^{(k)} - \left[\frac{13}{4}I - A^{-1}[z, y\,; F]\left(\frac{7}{2}I - \frac{5}{4}A^{-1}[z, y\,; F]\right)\right] \times A^{-1} F\left(z^{(k)}\right), \tag{43}$$

where I is the identity matrix, $A = [a, b\,; F]$ is the first order central difference operator, $a = x^{(k)} + F(x^{(k)})$ and $b = x^{(k)} - F(x^{(k)})$.

3.2. Numerical Results. Case $n = 25$. Tables 5 and 6 show the results of criteria JFC1 and JFC2 for Test Problems 1-4. As we can see from these tables, JFM6 is better among all the Jacobian free methods which are mentioned in this paper only for Test Problems 1-3; this method fails in solving the fourth test problem. Even though just for Test Problems 2 and 3, JFM5 is better, but this method fails to solve Test Problem 4 and diverges. Thus, we can claim that for case $n = 25$, JFM8 is a suitable method.

Table 5: Values of JFC1 for case $n = 25$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
JFNM	0.1228	0.0938	0.0478	0.0842
JFM2	0.1800	0.1798	0.0918	0.1208
JFM3	0.1769	0.1479	0.0815	0.1334
JFM4	0.1995	0.1701	0.0900	0.1474
JFM5	0.4654	0.4003	0.2757	div.
JFM6	0.5118	0.3499	0.2317	div.
JFM7	0.2640	0.2210	0.1218	0.2161
JFM8	0.2851	0.2205	0.1246	0.2188
JFM9	0.3851	0.3330	0.2486	div.

Table 6: Values of JFC2 for case $n = 25$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
JFNM	0.0733	0.0522	0.0254	0.0468
JFM2	0.0978	0.0978	0.0391	0.0557
JFM3	0.0969	0.0764	0.0330	0.0645
JFM4	0.1175	0.0941	0.0428	0.0782
JFM5	0.2818	0.2128	0.1215	div.
JFM6	0.3816	0.1926	0.1093	div.
JFM7	0.1538	0.1154	0.0553	0.1140
JFM8	0.1607	0.1152	0.0559	0.1148
JFM9	0.2399	0.1841	0.1225	div.

Case $n = 80$. Tables 7 and 8, respectively, show the results of criteria JFC1 and JFC2 for Test Problems 1-4. From the numerical results, JFM8 is the best method among the other Jacobian free methods for case $n = 80$.

3.3. Discussion on Numerical Results for Jacobian Free Methods. As computation of Jacobian matrix in solving nonlinear systems is the most consuming part of CPU time, hence applying Jacobian free solvers is very attractive. In this section, we selected just ten Jacobian free methods to compare them altogether. As our numerical results show, we can claim that high order convergence for a method can show qualification of a Jacobian free method. Therefore, our criteria can be useful to show qualification of new Jacobian free methods in the future. Meanwhile, JFM8 can be a good method to compare any new method with it.

4. The Third Category: Frozen Jacobian Methods

One of the attractive schemes in multistep iterative methods for solving nonlinear systems is the frozen Jacobian method. In any iteration of multistep frozen Jacobian method, the Jacobian matrix is same in all steps. Hence, one computes the inverse of the Jacobian matrix only once. For example, this inversion is performed in *LU* decomposition only once in any iteration. In this paper, we compare the frozen Jacobian iterative methods which are used to solve nonlinear systems by introducing two qualification criteria. The

Table 7: Values of JFC1 for case $n = 80$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
JFNM	0.0111	0.0077	0.0038	0.0065
JFM2	0.0137	0.0117	0.0046	0.0082
JFM3	0.0136	0.0072	0.0037	0.0080
JFM4	0.0133	0.0105	0.0036	0.0062
JFM5	0.0588	0.0449	0.0225	div.
JFM6	0.0363	0.0272	0.0158	div.
JFM7	0.0176	0.0122	0.0050	0.0133
JFM8	0.0184	0.0127	0.0052	0.0137
JFM9	0.0305	0.0227	0.0114	div.

Table 8: Values of JFC2 for case $n = 80$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
JFNM	0.0105	0.0072	0.0035	0.0062
JFM2	0.0129	0.0109	0.0042	0.0076
JFM3	0.0128	0.0068	0.0035	0.0074
JFM4	0.0127	0.0100	0.0034	0.0060
JFM5	0.0544	0.0408	0.0201	div.
JFM6	0.0345	0.0256	0.0147	div.
JFM7	0.0168	0.0116	0.0048	0.0126
JFM8	0.0175	0.0121	0.0049	0.0129
JFM9	0.0291	0.0215	0.0108	div.

parameters for this category are: order of convergence (OC), number of iterations (IT), number of function evaluations (FE), total numbers of Jacobian matrix evaluations (JME), CPU time (CPU), total numbers of steps of the iterative method (NOS), and number of LU-factors (LU.F). Hence we define our criteria as follows:

(i) Frozen Jacobian Criterion 1(FJC1): The first criterion is

$$\text{FJC1} = \frac{\text{OC}}{\text{CPU} + \text{IT} + \text{LU.F} + \text{FE} + \text{JME} + \text{NOS}}. \tag{44}$$

(ii) Frozen Jacobian Criterion 2(FJC2): The second criterion is defined as

$$\text{FJC2} = \frac{\text{OC}}{\text{CPU} + \text{IT} * (\text{LU.F} + \text{FE} + \text{JME} + \text{NOS})}. \tag{45}$$

4.1. Some Selected Frozen Jacobian Methods. To show goodness of our criteria, we have chosen some different frozen Jacobian methods which are presented in the last two decades. We have applied our proposed criteria (44) and (45) to compare qualification of these iterative methods.

Newton Method (NM): As our goal is the comparing iterative methods with the classical Newton's algorithm, for this category we consider the Newton's method as a one-step frozen Jacobian method. Therefore, the Newton's method is the first frozen Jacobian method with the following information:

$$\begin{cases} \text{NOS} = 1 \\ \text{OC} = 1 \\ \text{JME} = 1 \ \ \text{NM}: x^{(k+1)} = x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ \text{FE} = 1 \\ \text{LU.F} = 1 \end{cases} \tag{46}$$

where $J_F(\cdot)^{-1}$ is the inverse of Jacobian matrix.

Frozen Jacobian Method 2 (FJM2): A two-step, second order frozen Jacobian method is presented by Darvishi and Barati [6] in 2007 with the following information:

$$\begin{cases} \text{NOS} = 2 \\ \text{OC} = 3 \\ \text{JME} = 1 \ \ \text{FJM2}: \begin{cases} y^{(k)} = x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ x^{(k+1)} = x^{(k)} - J_F\left(x^{(k)}\right)^{-1} \left(F\left(x^{(k)}\right) + F\left(y^{(k)}\right)\right). \end{cases} \\ \text{FE} = 2 \\ \text{LU.F} = 1 \end{cases} \tag{47}$$

Frozen Jacobian Method 3 (FJM3): The third order frozen Jacobian method is presented by Qasim et al. [23] in 2016 with the following information:

$$\begin{cases} \text{NOS} = 2 \\ \text{OC} = 4 \\ \text{JME} = 2 \ \ \text{FJM3}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(y^{(k)}\right)\phi_2\right), \\ x^{(k+1)} = y^{(k)} - 2\phi_2 + \phi_3. \end{cases} \\ \text{FE} = 2 \\ \text{LU.F} = 1 \end{cases} \tag{48}$$

Frozen Jacobian Method 4 (FJM4): The fourth order frozen Jacobian method is presented by Ahmad et al. [24] in 2016 which has the following information:

$$\begin{cases} \text{NOS} = 3 \\ \text{OC} = 4 \\ \text{JME} = 1 \ \ \text{FJM4}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)} - \phi_1\right), \\ z^{(k)} = y^{(k)} - \phi_2, \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right), \\ x^{(k+1)} = z^{(k)} - \phi_3. \end{cases} \\ \text{FE} = 3 \\ \text{LU.F} = 1 \end{cases} \tag{49}$$

Frozen Jacobian Method 5 (FJM5): The fifth order frozen Jacobian method is presented by Qasim et al. [23] in 2016 as follows:

$$\begin{cases} \text{NOS} = 2 \\ \text{OC} = 5 \\ \text{JME} = 2 \\ \text{FE} = 2 \\ \text{LU.F} = 1 \end{cases} \text{FJM5}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(y^{(k)}\right)\phi_2\right), \\ \phi_4 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(y^{(k)}\right)\phi_3\right), \\ x^{(k+1)} = y^{(k)} - \dfrac{13}{4}\phi_2 + \dfrac{7}{2}\phi_3 - \dfrac{5}{4}\phi_4. \end{cases} \tag{50}$$

Frozen Jacobian Method 6 (FJM6): The sixth order frozen Jacobian method is presented by Cordero et al. [25] in 2016 as:

$$\begin{cases} \text{NOS} = 3 \\ \text{OC} = 5 \\ \text{JME} = 1 \\ \text{FE} = 3 \\ \text{LU.F} = 1 \end{cases} \text{FJM6}: \begin{cases} y^{(k)} = x^{(k)} - J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ z^{(k)} = y^{(k)} - 5J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ x^{(k+1)} = z^{(k)} - J_F\left(x^{(k)}\right)^{-1} \left(\dfrac{-16}{5} F\left(y^{(k)}\right) + \dfrac{1}{5} F\left(z^{(k)}\right)\right). \end{cases} \tag{51}$$

Frozen Jacobian Method 7 (FJM7): The following seventh order frozen Jacobian method is presented by Montazeri et al. [4] in 2012:

$$\begin{cases} \text{NOS} = 3 \\ \text{OC} = 6 \\ \text{JME} = 2 \\ \text{FE} = 2 \\ \text{LU.F} = 1 \end{cases} \text{FJM7}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \dfrac{2}{3}\phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(y^{(k)}\right)\phi_1\right), \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(y^{(k)}\right)\phi_2\right), \\ z^{(k)} = x^{(k)} - \dfrac{23}{8}\phi_1 + 3\phi_2 - \dfrac{9}{8}\phi_3, \\ \phi_4 = J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right), \\ \phi_5 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(y^{(k)}\right)\phi_4\right), \\ x^{(k+1)} = z^{(k)} - \dfrac{5}{2}\phi_4 + \dfrac{3}{2}\phi_5. \end{cases} \tag{52}$$

Frozen Jacobian Method 8 (FJM8): The following eighth order frozen Jacobian method is presented by Kouser et al. [26] in 2018:

$$\begin{cases} \text{NOS} = 5 \\ \text{OC} = 7 \\ \text{JME} = 1 \\ \text{FE} = 5 \\ \text{LU.F} = 1 \end{cases} \text{FJM8}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ z^{(k)} = x^{(k)} - \phi_1 - 5\phi_2, \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right), \\ w^{(k)} = x^{(k)} - \phi_1 - \dfrac{9}{5}\phi_2 - \dfrac{1}{5}\phi_3, \\ \phi_4 = J_F\left(x^{(k)}\right)^{-1} F\left(w^{(k)}\right), \\ t^{(k)} = w^{(k)} - \phi_4, \\ \phi_5 = J_F\left(x^{(k)}\right)^{-1} F\left(t^{(k)}\right), \\ x^{(k+1)} = t^{(k)} - \phi_5. \end{cases} \tag{53}$$

Frozen Jacobian Method 9 (FJM9): The following ninth order frozen Jacobian method is presented by Alzahrani et al. [27] in 2016:

$$\begin{cases} \text{NOS} = 4 \\ \text{OC} = 9 \\ \text{JME} = 2 \\ \text{FE} = 3 \\ \text{LU.F} = 1 \end{cases} \text{FJM9}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ z^{(k)} = y^{(k)} - \dfrac{1}{2}\phi_2, \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(z^{(k)}\right)\phi_2\right), \\ \phi_4 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(z^{(k)}\right)\phi_3\right), \\ \phi_5 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(z^{(k)}\right)\phi_4\right), \\ w^{(k)} = y^{(k)} - \dfrac{17}{4}\phi_2 + \dfrac{27}{4}\phi_3 - \dfrac{19}{4}\phi_4 + \dfrac{5}{4}\phi_5, \\ \phi_6 = J_F\left(x^{(k)}\right)^{-1} F\left(w^{(k)}\right), \\ \phi_7 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(z^{(k)}\right)\phi_6\right), \\ \phi_8 = J_F\left(x^{(k)}\right)^{-1} \left(J_F\left(z^{(k)}\right)\phi_7\right), \\ x^{(k+1)} = w^{(k)} - \dfrac{13}{4}\phi_6 + \dfrac{7}{2}\phi_7 - \dfrac{5}{4}\phi_8. \end{cases} \tag{54}$$

Frozen Jacobian Method 10 (FJM10): The following tenth order frozen Jacobian method is presented by Ahmad et al. [28] in 2016 with the following information:

$$\begin{cases} \text{NOS} = 4 \\ \text{OC} = 9 \\ \text{JME} = 2 \\ \text{FE} = 4 \\ \text{LU.F} = 1 \end{cases} \text{FJM10}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ z^{(k)} = y^{(k)} - \phi_2, \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right), \\ w^{(k)} = z^{(k)} - \phi_3, \\ \phi_4 = J_F\left(x^{(k)}\right)^{-1} F\left(w^{(k)}\right), \\ \phi_5 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_4\right), \\ \phi_6 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_5\right), \\ \phi_7 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_6\right), \\ \phi_8 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_7\right), \\ x^{(k+1)} = w^{(k)} - \frac{21}{4}\phi_4 + 11\phi_5 - \frac{23}{2}\phi_6 + 6\phi_7 - \frac{5}{4}\phi_8. \end{cases} \tag{55}$$

Frozen Jacobian Method 11 (FJM11): The following eleventh order frozen Jacobian method is presented by Qasim et al. [29] in 2016:

$$\begin{cases} \text{NOS} = 4 \\ \text{OC} = 10 \\ \text{JME} = 2 \\ \text{FE} = 4 \\ \text{LU.F} = 1 \end{cases} \text{FJM11}: \begin{cases} \phi_1 = J_F\left(x^{(k)}\right)^{-1} F\left(x^{(k)}\right), \\ y^{(k)} = x^{(k)} - \phi_1, \\ \phi_2 = J_F\left(x^{(k)}\right)^{-1} F\left(y^{(k)}\right), \\ z^{(k)} = x^{(k)} - \phi_1 - 5\phi_2, \\ \phi_3 = J_F\left(x^{(k)}\right)^{-1} F\left(z^{(k)}\right), \\ w^{(k)} = x^{(k)} - \phi_1 - \frac{9}{5}\phi_2 - \frac{1}{5}\phi_3, \\ \phi_4 = J_F\left(x^{(k)}\right)^{-1} F\left(w^{(k)}\right), \\ \phi_5 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_4\right), \\ \phi_6 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_5\right), \\ \phi_7 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_6\right), \\ \phi_8 = J_F\left(x^{(k)}\right)^{-1}\left(J_F\left(w^{(k)}\right)\phi_7\right), \\ x^{(k+1)} = w^{(k)} - 5\phi_4 + 10\phi_5 - 10\phi_6 + 5\phi_7 - \phi_8. \end{cases} \tag{56}$$

4.2. Numerical Results. Case $n = 100$. Tables 9 and 10 show the results of criteria FJC1 and FJC2 for Test Problems 1-4. As these tables show, for case $n = 100$, FJM9-FJM11 have better results with respect to the other methods of this category.

Table 9: Values of FJC1 for case $n = 100$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.2108	0.1566	0.0978	0.1715
FJM2	0.3122	0.2281	0.1555	0.2512
FJM3	0.3591	0.2855	0.2066	0.3212
JFM4	0.3361	0.2770	0.2040	0.3049
FJM5	0.4280	0.3772	0.2541	0.3798
FJM6	0.4213	0.3791	0.2699	0.3812
FJM7	0.5210	0.3971	0.2724	0.4020
FJM8	0.4651	0.3865	0.3125	0.4271
FJM9	0.6220	0.4828	0.3731	0.5431
FJM10	0.5858	0.5952	0.3660	0.5173
FJM11	0.6466	0.5826	0.4063	0.5750

Table 10: Values of FJC2 for case $n = 100$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.0976	0.0610	0.0325	0.0698
FJM2	0.1612	0.0807	0.0439	0.0970
FJM3	0.1807	0.1081	0.0603	0.1358
JFM4	0.1606	0.0965	0.0536	0.1208
FJM5	0.2204	0.1653	0.0824	0.1658
FJM6	0.2011	0.1507	0.0752	0.1510
FJM7	0.3425	0.1709	0.0857	0.1718
FJM8	0.2794	0.1397	0.0801	0.1872
FJM9	0.4005	0.2016	0.1321	0.2681
FJM10	0.3694	0.3731	0.1223	0.2473
FJM11	0.4087	0.2765	0.1358	0.2748

Table 11: Values of FJC1 for case $n = 500$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.0347	0.0197	0.0105	0.0252
FJM2	0.0367	0.0138	0.0081	0.0249
FJM3	0.0288	0.0135	0.0085	0.0172
FJM4	0.0375	0.0232	0.0085	0.0293
FJM5	0.0183	0.0139	0.0128	0.0197
FJM6	0.0295	0.0301	0.0084	0.0324
FJM7	0.0334	0.0149	0.0069	0.0133
FJM8	0.0588	0.0252	0.0159	0.0408
FJM9	0.0319	0.0159	0.0082	0.0209
FJM10	0.0194	0.0281	0.0090	0.0236
FJM11	0.0326	0.0153	0.0108	0.0193

Case $n = 500$. Tables 11 and 12 show the results of criteria FJC1 and FJC2 for Test Problems 1-4. As Table 11 for FJC1 shows, FJM8 acts better than the other methods for Test Problems 1, 3, and 4 while for the other problem, FJM10 acts better. For FJC2 as Table 12 shows, six methods FJM6-FJM11 are better than the others.

Table 12: Values of FJC2 for case $n = 500$.

Method	Problem 1	Problem 2	Problem 3	Problem 4
NM	0.0292	0.0161	0.0084	0.0207
FJM2	0.0331	0.0125	0.0070	0.0215
FJM3	0.0267	0.0125	0.0077	0.0160
FJM4	0.0334	0.0201	0.0075	0.0255
FJM5	0.0176	0.0131	0.0113	0.0185
FJM6	0.0274	0.0259	0.0077	0.0287
FJM7	0.0323	0.0142	0.0065	0.0127
FJM8	0.0543	0.0226	0.0135	0.0364
FJM9	0.0311	0.0152	0.0078	0.0201
FJM10	0.0191	0.0273	0.0085	0.0225
FJM11	0.0317	0.0146	0.0102	0.0187

4.3. Discussion on Numerical Results for the Frozen Jacobian Methods. Among ten frozen Jacobian methods FJM2-FJM11, four methods namely, FJM8-FJM11 have better values. Hence, we considered these methods separately and solved Test Problem 1 by them for some different values of n. Figure 2 shows that, FJM8 demonstrates brilliant results by increasing n. It must be noted that to obtain these results, the CPU time is computed in minutes. From these figures, we can conclude that order of convergence, solely, cannot show an advantage for an algorithm in this category. It must be noted that, order of convergence for FJM8 is 7 while for FJM9, FJM10, and FJM11 are 9, 9, and 10, respectively.

5. A Real Test Problem

In this part, we consider a real problem as our last test problem. This is the well-known Van der Pol nonlinear differential equation.

Test Problem 5. The Van der Pol equation governs the flow of current in a vacuum tube, and it has the following form [30].

$$u'' - \beta(u^2 - 1)u' + u = 0, \beta > 0, \tag{57}$$

with boundary conditions

$$u(0) = 0, u(2) = 1. \tag{58}$$

To solve the boundary value problem (57), numerically, first we use the following second-order finite difference approximations:

$$\begin{aligned} u_i' &\approx \frac{u_{i+1} - u_{i-1}}{2h}, i = 1, 2, 3, \cdots, n-1, \\ u_i'' &\approx \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2}, i = 1, 2, 3, \cdots, n-1. \end{aligned} \tag{59}$$

This leads to the following system of nonlinear equations:

$$2(h^2 - 1)u_i - h\beta(u_i^2 - 1)(u_{i+1} - u_{i-1}) + 2(u_{i-1} + u_{i+1}) = 0, i = 1, 2, 3, \cdots, n-1. \tag{60}$$

In system (60), $h = 2/n$ is the step size and $u_1, u_2, u_3, \cdots$ are the approximations of the unknown $u(x_i), i = 1, 2, 3, \cdots$. To solve (60), we set the initial guess as $(1, 1, \cdots, 1)^t$. The stopping criterion is selected as $\|f(\mathbf{x}^{(k)})\| \leq 10^{-6}$, and β is considered as $\beta = 1/2$.

Results for the first category. Tables 13 and 14 show the values of parameters of the algorithms in the first category for cases $n = 100$ and $n = 250$, respectively. Table 15 shows the values of CR1 and CR2 for the algorithms of this category for $n = 100$. As this table shows, algorithm M9 is the best algorithm among the others. By increasing the number of equations in (60) to $n = 250$, M3 has the best values for criteria CR1 and CR2 as Table 16 shows. These confirm our previous discussion on Test Problems 1-4.

Results for the second category. As one may see from Table 17, in this category and for $n = 10$, JFM5 works better than the other algorithms. Because our criteria JFC1 and JFC2 have better values for this method. By increasing n to $n = 25$, again JFM5 is the best algorithm among the others. We can see this superiority from the reported results in Table 18.

Results for the third category. For the algorithms of the third category and for $n = 100$, FJM11 is the best algorithm for this case, see Table 19, while for $n = 250$ as Table 20 shows, FJM6 is the best algorithm among the others.

6. The Classical Efficiency Indices and their Corrections

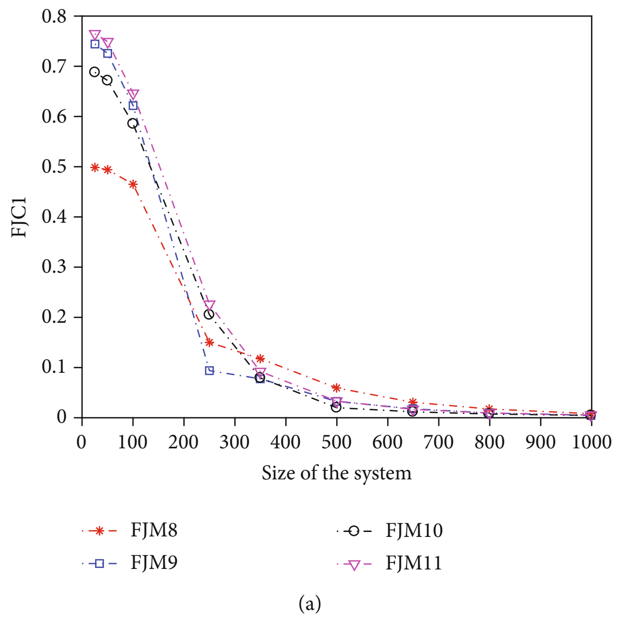
As we mentioned in the first section, there are some qualification criteria for nonlinear solvers which we call them the classical efficiency indices. To have a comparison between our qualification criteria and the classical ones, in this section we present numerical results for the classical efficiency indices for all mentioned methods in each category to solve Test Problem 1.

6.1. The First Category. The classical efficiency index (EI). Table 21 shows values of the classical efficiency index for the methods of the first category. As this table shows, methods M9 and M10 work better than the other methods.

The operational index (I_{op}). Table 22 shows the values of I_{op} for the methods in the first category. Therefore, by this criterion, M2 is the best method among the others which conflicts with EI.

The computational efficiency index (CEI). Table 23 shows the values of CEI for the methods in the first category. From the results of this table, we conclude that using this criterion, M2 is the best method among the others for $n \geq 10$.

*The flops-like efficiency index (*FLEI *).* Table 24 shows the values of FLEI for the methods in the first category. Here also, M2 is the distinguished method.



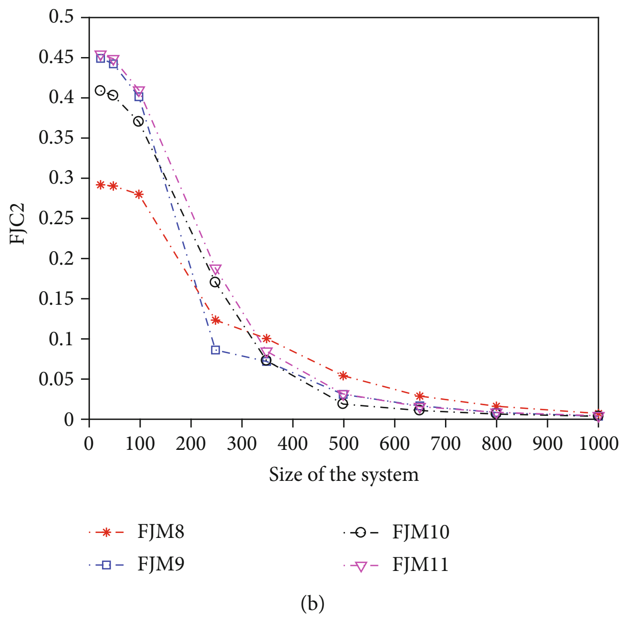


Figure 2: Values of FJC1 (a) and FJC2 (b) to solve Test Problem 1 for FJM8-FJM11 as a function of n.

Table 13: Parameters of methods 1-10 to solve Test Problem 5, by the algorithms of the first category for $n = 100$.

Method	OC	CPU (m)	IT	FE	JE
NM	2	0.4240	4	1	1
M2	3	1.4948	3	2	1
M3	4	0.4115	2	2	2
M4	4	1.5466	2	2	3
M5	4	1.6396	2	1	2
M6	3	0.5266	2	1	2
M7	5	0.8044	2	2	2
M8	5	1.2547	2	2	2
M9	6	1.2576	2	2	2
M10	6	1.6706	2	2	2

Table 14: Parameters of methods 1-10 to solve Test Problem 5, by the algorithms of the first category for $n = 250$.

Method	OC	CPU (m)	IT	FE	JE
NM	2	5.5477	4	1	1
M2	3	8.3057	3	2	1
M3	4	5.5638	2	2	2
M4	4	13.8172	2	2	3
M5	4	16.4128	2	1	2
M6	3	5.5128	2	1	2
M7	5	12.9727	2	2	2
M8	5	13.2445	2	2	2
M9	6	23.7789	2	2	2
M10	6	31.7586	2	2	2

The computational index. Lotfi et al. [13] defined the computational index as

$$CI = p^{1/(d+op)}, \tag{61}$$

where p is the order of convergence, d is the number of functional evaluations per step, and op shows the number of products/quotients per step. Table 25 shows the values of CI for the methods in the first category which shows M2 is the best method among the other methods.

Therefore, by EI, CEI, FLEI, and CI, M2 is the best method among all methods of the first category.

6.2. The Second Category. ***The classical efficiency index.*** Table 26 shows the values of EI for the methods in the second category. This criterion selects JFM5 as the best method.

The operational index. Table 27 shows values of I_{op} for the methods in the second category. This criterion fails to introduce the best method. For small values of n, namely $n = 3$, JFNM shows a better result, for $5 \le n \le 80$, JFM5 is the better method while for $n = 80$, JFM6 is the best method.

Table 15: Values of CR1 and CR2 for methods 1-10 to solve Test Problem 5, by the algorithms of the first category for $n = 100$.

Method	CR1	CR2
NM	0.2374	0.3113
M2	0.2859	0.4003
M3	0.4755	0.6239
M4	0.3464	0.4680
M5	0.5236	0.6024
M6	0.4597	0.5428
M7	0.5679	0.7348
M8	0.5403	0.6892
M9	0.6481	0.8267
M10	0.6204	0.7822

Table 16: Values of CR1 and CR2 for methods 1-10 to solve Test Problem 5, by the algorithms of the first category for $n = 250$.

Method	CR1	CR2
NM	0.1476	0.1732
M2	0.1734	0.2097
M3	0.2949	0.3459
M4	0.1679	0.1921
M5	0.1785	0.1868
M6	0.2606	0.2854
M7	0.2384	0.2635
M8	0.2354	0.2598
M9	0.1888	0.2015
M10	0.1509	0.1589

Table 17: Values of JFC1 and JFC2 for methods 1-9 to solve Test Problem 5, by the algorithms of the second category for $n = 10$.

Method	JFC1	JFC2
JFNM	0.2401	0.1305
JFM2	0.3192	0.1630
JFM3	0.3144	0.1618
JFM4	0.4213	0.2163
JFM5	0.7026	0.4942
JFM6	0.6393	0.3263
JFM7	0.4661	0.2058
JFM8	0.4716	0.2068
JFM9	0.5945	0.2433

Table 18: Values of JFC1 and JFC2 for methods 1-9 to solve Test Problem 5, by the algorithms of the second category for $n = 25$.

Method	JFC1	JFC2
JFNM	0.2624	0.4328
JFM2	0.2080	0.1281
JFM3	0.2091	0.1285
JFM4	0.2411	0.1563
JFM5	0.5806	0.4306
JFM6	0.5212	0.3868
JFM7	0.3272	0.2303
JFM8	0.2670	0.1548
JFM9	0.2358	0.1721

The computational efficiency index. Table 28 shows the results of *CEI* for the methods in the second category. By this index, JFM6 is the best method among all methods.

The flops-like efficiency index. The values of FLEI are presented in Table 29 for the methods in the second category. By this criterion, JFM5 is the best method.

The computational index. Table 30 shows results of CI for the methods in the second category. From the reported results in this table, we conclude that by this criterion JFM5 is the best method.

Table 19: Values of FJC1 and FJC2 for methods 1-11 to solve Test Problem 5, by the algorithms of the third category for $n = 100$.

Method	FJC1	FJC2
NM	0.2374	0.1218
FJM2	0.3112	0.1609
FJM3	0.4098	0.2710
FJM4	0.3721	0.2388
FJM5	0.4930	0.3302
FJM6	0.4711	0.3009
FJM7	0.5210	0.3425
FJM8	0.4576	0.2767
FJM9	0.6198	0.3996
FJM10	0.5715	0.3637
FJM11	0.6436	0.4075

Table 20: Values of FJC1 and FJC2 for Methods 1-11 to solve Test Problem 5, by the algorithms of the third category for $n = 250$.

Method	FJC1	FJC2
NM	0.1476	0.0928
FJM2	0.1665	0.1110
FJM3	0.1870	0.1516
FJM4	0.1924	0.1493
FJM5	0.1683	0.1440
FJM6	0.2723	0.2052
FJM7	0.1322	0.1167
FJM8	0.2297	0.1730
FJM9	0.1890	0.1618
FJM10	0.1950	0.1632
FJM11	0.2195	0.1833

Therefore for the solvers in the second category, three criteria EI, FLEI, and CI introduce JFM5 as the best method while JFM6 is selected as the best method by I_{op}, and CEI.

6.3. The Third Category. The classical efficiency index. Table 31 shows the values of EI for the methods in the third category. For large values of n, FJM8 is better than the other methods by criterion EI.

The operational index. Table 32 shows the values of I_{op} for the methods in the third category. By this index, FJM11 works better than the other methods for large values of n while for small values of the problem size, FJM6 is the best method.

The computational efficiency index. The values of CEI for the methods in the third category are presented in Table 33. By CEI, there is a similar situation to I_{op}.

The flops-like efficiency index. Table 34 shows the values of FLEI for the methods in the third category. By this criterion, one cannot select a distinguished method which works better than the other methods for all values of n.

The computational index. Table 35 shows the values of CI for the methods in the third category. As this table shows, for $n \leq 10$, the value of CI for FJM6 is better with respect to

Table 21: Values of EI for methods in the first category to solve Test Problem 1, and $n = 3, 5, 10, 100, 250, 500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.059463094	1.023373892	1.006321233	1.000068631	1.000011046	1.000002767
M2	1.075989625	1.031886750	1.009197139	1.000107713	1.000017438	1.000004377
M3	1.059463094	1.023373892	1.006321233	1.000068631	1.000011046	1.000002767
M4	1.042903781	1.016443069	1.004341567	1.000045905	1.000007374	1.000001846
M5	1.068241691	1.025525693	1.006623239	1.000068972	1.000011068	1.000002770
M6	1.053707472	1.020175600	1.005245195	1.000054659	1.000008771	1.000002195
M7	1.069359545	1.027186966	1.007342451	1.000079678	1.000012824	1.000003212
M8	1.069359545	1.027186966	1.007342451	1.000079678	1.000012824	1.000003212
M9	1.077514117	1.030313019	1.008177617	1.000088705	1.000014277	1.000003576
M10	1.077514117	1.030313019	1.008177617	1.000088705	1.000014277	1.000003576

Table 22: The values of I_{op} for the methods in the first category to solve Test Problem 1, and $n = 3, 5, 10, 100, 250, 500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.041616011	1.010720864	1.001613270	1.000002019	1.000000132	1.000000017
M2	1.043159740	1.012281610	1.002075003	1.000003110	1.000000206	1.000000026
M3	1.032764723	1.008983950	1.001445100	1.000001990	1.000000131	1.000000016
M4	1.032764723	1.008983950	1.001445100	1.000001990	1.000000131	1.000000016
M5	1.027018051	1.007731369	1.001308680	1.000001962	1.000000130	1.000000016
M6	1.032839831	1.008486673	1.001278272	1.000001600	1.000000104	1.000000013
M7	1.032060827	1.008287682	1.001248405	1.000001563	1.000000102	1.000000013
M8	1.026735373	1.007881816	1.001388409	1.000002246	1.000000150	1.000000019
M9	1.020569600	1.006419659	1.001227986	1.000002400	1.000000164	1.000000021
M10	1.022939658	1.007051252	1.001318338	1.000002432	1.000000165	1.000000021

Table 23: The values of CEI for the methods in the first category to solve Test Problem 1, with $n = 5, 10, 100, 250, 500$.

Method	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.005792941	1.001035083	1.000001906	1.000000128	1.000000016
M2	1.007605412	1.001593460	1.000003019	1.000000204	1.000000026
M3	1.007731369	1.001238529	1.000001960	1.000000130	1.000000016
M4	1.005245007	1.001083629	1.000001931	1.000000129	1.000000016
M5	1.007322969	1.001346823	1.000001988	1.000000130	1.000000016
M6	1.005245195	1.000981385	1.000001553	1.000000103	1.000000013
M7	1.006091831	1.001069966	1.000001539	1.000000101	1.000000013
M8	1.006590758	1.001298776	1.000002212	1.000000149	1.000000019
M9	1.007995167	1.001559266	1.000002533	1.000000168	1.000000021
M10	1.006915183	1.001411830	1.000002497	1.000000167	1.000000021

the other methods while for $n \geq 100$, FJM11 is the best method.

Hence for this category, one cannot select the best method by all indices.

6.4. Corrections on the Classical Efficiency Indices. As we know, CPU time is a very important parameter to show quality of an algorithm, but this has not considered in the classical efficiency indices. In this part, we enter this parameter to the classical efficiency indices and compute the values of corrected indices to solve Test Problem 1. Since for all nonlinear solvers CPU time has a negative role, hence we introduce the corrected efficiency indices as follows:

$$\text{COREI} = \frac{\text{EI}}{\text{CPUtime}}, \text{CORIOP} = \frac{I_{op}}{\text{CPUtime}}, \text{CORCEI} = \frac{\text{CEI}}{\text{CPUtime}},$$
$$\text{CORFLEI} = \frac{\text{FLEI}}{\text{CPUtime}}, \text{CORCI} = \frac{\text{CI}}{\text{CPUtime}}. \tag{62}$$

Table 24: The values of FLEI for the methods in the first category to solve Test Problem 1, and $n = 3, 5, 10, 100, 250, 500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.014545335	1.004252776	1.000709959	1.000000995	1.000000065	1.000000008
M2	1.016049346	1.005044493	1.000926226	1.000001533	1.000000102	1.000000013
M3	1.014545335	1.004252776	1.000709959	1.000000995	1.000000065	1.000000008
M4	1.011334439	1.003457318	1.000615409	1.000000974	1.000000065	1.000000008
M5	1.013683900	1.003500974	1.000505460	1.000000584	1.000000038	1.000000005
M6	1.011883085	1.003421214	1.000565483	1.000000788	1.000000052	1.000000007
M7	1.012267346	1.003504906	1.000570886	1.000000774	1.000000051	1.000000006
M8	1.014218004	1.004281986	1.000747696	1.000001139	1.000000075	1.000000010
M9	1.014673768	1.003974874	1.000606875	1.000000749	1.000000049	1.000000006
M10	1.012788598	1.003577999	1.000568373	1.000000742	1.000000048	1.000000006

Table 25: The values of CI for the methods in the first category to solve Test Problem 1, and $n = 3, 5, 10, 100, 250, 500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.024189560	1.007322969	1.001284430	1.000001961	1.000000130	1.000000016
M2	1.027157647	1.008827634	1.001691602	1.000003022	1.000000204	1.000000026
M3	1.020906503	1.006468713	1.001175516	1.000001934	1.000000129	1.000000016
M4	1.018408093	1.005792941	1.001083629	1.000001907	1.000000128	1.000000016
M5	1.019171797	1.005916559	1.001092166	1.000001908	1.000000128	1.000000016
M6	1.020175600	1.005956112	1.001027268	1.000001555	1.000000103	1.000000013
M7	1.021691076	1.006331481	1.001066421	1.000001533	1.000000101	1.000000013
M8	1.019114959	1.006091831	1.001166940	1.000002184	1.000000148	1.000000019
M9	1.016126503	1.005283791	1.001067092	1.000002337	1.000000162	1.000000021
M10	1.017547910	1.005704333	1.001134668	1.000002368	1.000000163	1.000000021

Table 26: Values of EI for the methods in the second category to solve Test Problem 1, and $n = 3, 5, 10, 25, 50, 80$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 25$	$n = 50$	$n = 80$
JFNM	1.059463094	1.023373892	1.006321233	1.001066949	1.000271859	1.000106973
JFM2	1.053707472	1.020175600	1.005245195	1.000862028	1.000217571	1.000085300
JFM3	1.053707472	1.020175600	1.005245195	1.000862028	1.000217571	1.000085300
JFM4	1.052685203	1.018655810	1.004631674	1.000739630	1.000184856	1.000072205
JFM5	1.079653224	1.036412558	1.011562318	1.002222380	1.000596266	1.000239528
JFM6	1.068612910	1.027948975	1.007820682	1.001353186	1.000347975	1.000137414
JFM7	1.047421253	1.016348109	1.003979141	1.000627910	1.000156310	1.000060964
JFM8	1.047421253	1.016348109	1.003979141	1.000627910	1.000156310	1.000060964
JFM9	1.060740208	1.023157118	1.006099496	1.001011374	1.000256074	1.000100517

The results of these corrected indices for solving Test Problem 1 are reported in Tables 36–38 for our three categories and some selected values of n. As we can see from Tables 26–28 and 31–35, by using the classical efficiency indices, we cannot crucially talk about the best method in some categories. But as Tables 36–38 show, for the corrected efficiency indices and for any case, there is a unique method which acts better than the other methods. For the first category, M2 is the best method (Cf. Table 36). For the second category, JFM5 is the best method (the results may be found in Table 37). Finally, for the third category, FJM2 is the best method (Cf. Table 38).

7. Concluding Remarks

Solving systems of nonlinear equations is a very applicable field in different areas of mathematics, physics, etc. Each year many different algorithms are presented to solve nonlinear systems. As there are different types for these solvers, hence comparing them must be done in a right category. In this paper, some new qualification criteria for nonlinear solvers were introduced. They use all parameters in each relevant algorithm, such as order of convergence, CPU time, number of function, and Jacobian evaluations. Hence, they are suitable to compare different algorithms which solve

Table 27: Values of I_{op} for the methods in the second category to solve Test Problem 1, and $n = 3, 5, 10,25,50,80$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 25$	$n = 50$	$n = 80$
JFNM	1.041616011	1.010720864	1.001613270	1.000119002	1.000015700	1.000003915
JFM2	1.032839831	1.008486673	1.001278272	1.000094306	1.000012442	1.000003103
JFM3	1.032839831	1.008486673	1.001278272	1.000094306	1.000012442	1.000003103
JFM4	1.032008280	1.009951291	1.001900838	1.000180054	1.000026841	1.000007064
JFM5	1.037255346	1.011562318	1.002207142	1.000209040	1.000031161	1.000008201
JFM6	1.025557174	1.008368587	1.001741086	1.000187146	1.000030292	1.000008317
JFM7	1.028603026	1.007974116	1.001306833	1.000103926	1.000014157	1.000003578
JFM8	1.038892354	1.010028983	1.001509596	1.000111360	1.000014692	1.000003664
JFM9	1.022104931	1.007370084	1.001583293	1.000179777	1.000030334	1.000008526

Table 28: Results of CEI for the methods in the second category to solve Test Problem 1, and $n = 5,10,25,50,80$.

Method	$n = 5$	$n = 10$	$n = 25$	$n = 50$	$n = 80$
JFNM	1.004109885	1.000878900	1.000088314	1.000013343	1.000003524
JFM2	1.004439715	1.000880686	1.000079914	1.000011414	1.000002937
JFM3	1.004531274	1.000887803	1.000080059	1.000011420	1.000002938
JFM4	1.006264107	1.001504704	1.000165601	1.000025853	1.000006911
JFM5	1.007730386	1.001853774	1.000199975	1.000030745	1.000008155
JFM6	1.008447497	1.001986200	1.000215301	1.000032609	1.000008787
JFM7	1.005419941	1.001066821	1.000095768	1.000013594	1.000003488
JFM8	1.006005376	1.001141948	1.000099059	1.000013846	1.000003530
JFM9	1.006573397	1.001697961	1.000204918	1.000033698	1.000009231

Table 29: Values of FLEI for the methods in the second category to solve Test Problem 1, and $n = 3, 5, 10,25,50,80$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 25$	$n = 50$	$n = 80$
JFNM	1.014545335	1.004252776	1.000709959	1.000056279	1.000007627	1.000001922
JFM2	1.011883085	1.003421214	1.000565483	1.000044645	1.000006046	1.000001523
JFM3	1.011883085	1.003421214	1.000565483	1.000044645	1.000006046	1.000001523
JFM4	1.012918795	1.003622967	1.000585930	1.000045578	1.000006139	1.000001544
JFM5	1.017456397	1.005799162	1.001144805	1.000108082	1.000015930	1.000004165
JFM6	1.014321903	1.004241472	1.000717918	1.000057584	1.000007842	1.000001980
JFM7	1.011054497	1.003021470	1.000475886	1.000036220	1.000004835	1.000001211
JFM8	1.013057245	1.003749145	1.000629943	1.000051075	1.000007014	1.000001778
JFM9	1.012313625	1.003429757	1.000542687	1.000041104	1.000005467	1.000001367

Table 30: Values of CI for the methods in the second category to solve Test Problem 1, and $n = 3, 5, 10,25,50,80$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 25$	$n = 50$	$n = 80$
JFNM	1.024189560	1.007322969	1.001284430	1.000107055	1.000014843	1.000003777
JFM2	1.020175600	1.005956112	1.001027268	1.000085003	1.000011769	1.000002994
JFM3	1.020175600	1.005956112	1.001027268	1.000085003	1.000011769	1.000002994
JFM4	1.019717138	1.006468713	1.001346823	1.000144793	1.000023437	1.000006435
JFM5	1.025069672	1.008737616	1.001851641	1.000191049	1.000029613	1.000007930
JFM6	1.018451422	1.006419659	1.001423043	1.000164395	1.000027866	1.000007842
JFM7	1.017685287	1.005345497	1.000983266	1.000089164	1.000012981	1.000003379
JFM8	1.021144202	1.006196613	1.001093806	1.000094581	1.000013429	1.000003456
JFM9	1.016077965	1.005575227	1.001256214	1.000152632	1.000027121	1.000007859

Table 31: Values of EI for the methods in the third category to solve Test Problem 1, and $n = 3, 5, 10,100,250,500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.059463094	1.023373892	1.006321233	1.000068631	1.000011046	1.000002767
FJM2	1.075989625	1.031886750	1.009197139	1.000107713	1.000017438	1.000004377
FJM3	1.059463094	1.023373892	1.006321233	1.000068631	1.000011046	1.000002767
FJM4	1.080059739	1.035264924	1.010720864	1.000134601	1.000021918	1.000005512
FJM5	1.069359545	1.027186966	1.007342451	1.000079678	1.000012824	1.000003212
FJM6	1.093532430	1.041056380	1.012457245	1.000156268	1.000025446	1.000006399
FJM7	1.077514117	1.030313019	1.008177617	1.000088705	1.000014277	1.000003576
FJM8	1.084457205	1.039685437	1.013057245	1.000185342	1.000030525	1.000007707
FJM9	1.084781613	1.034381284	1.009598927	1.000108244	1.000017473	1.000004381
FJM10	1.075989625	1.031886750	1.009197139	1.000107713	1.000017438	1.000004377
FJM11	1.079775162	1.033441064	1.009640276	1.000112878	1.000018275	1.000004587

Table 32: Values of I_{op} for the methods in the third category to solve Test Problem 1, and $n = 3, 5, 10,100,250,500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.041616011	1.010720864	1.001613270	1.000002019	1.000000132	1.000000017
FJM2	1.043159740	1.012281610	1.002075003	1.000003110	1.000000206	1.000000026
FJM3	1.026501581	1.008437178	1.001671630	1.000003617	1.000000251	1.000000032
FJM4	1.040403283	1.012127685	1.002202890	1.000003816	1.000000257	1.000000033
FJM5	1.020321704	1.006728527	1.001425296	1.000003894	1.000000282	1.000000037
FJM6	1.047057595	1.014093502	1.002557929	1.000004430	1.000000298	1.000000038
FJM7	1.016886406	1.005704333	1.001253764	1.000004042	1.000000304	1.000000040
FJM8	1.037397617	1.011863211	1.002347220	1.000005077	1.000000352	1.000000045
FJM9	1.013008737	1.004494201	1.001032093	1.000004281	1.000000347	1.000000048
FJM10	1.014560410	1.005006181	1.001139107	1.000004454	1.000000354	1.000000048
FJM11	1.015263908	1.005246865	1.001193761	1.000004668	1.000000371	1.000000050

Table 33: Values of CEI for the methods in the third category to solve Test Problem 1, and $n = 5,10,100,250,500$.

Method	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.005792941	1.001035083	1.000001906	1.000000128	1.000000016
FJM2	1.007605412	1.001593460	1.000003019	1.000000204	1.000000026
FJM3	1.007322969	1.001671630	1.000003704	1.000000254	1.000000032
FJM4	1.007521629	1.001692033	1.000003705	1.000000254	1.000000032
FJM5	1.007881816	1.001820229	1.000004184	1.000000291	1.000000037
FJM6	1.008079655	1.001895250	1.000004298	1.000000295	1.000000038
FJM7	1.007192783	1.001749587	1.000004597	1.000000323	1.000000042
FJM8	1.007974116	1.001918992	1.000004994	1.000000350	1.000000045
FJM9	1.006208559	1.000000050	1.000005227	1.000000382	1.000000050
FJM10	1.006389091	1.001972548	1.000005230	1.000000383	1.000000050
FJM11	1.006507236	1.001663899	1.000005478	1.000000401	1.000000053

systems of nonlinear equations. In this paper, just to show usefulness of these criteria, we only compared some nonlinear solvers from different categories. As a matter of fact, we must categorise different algorithms, e.g., for their order of convergence, after this categorization, we can compare any algorithm with same convergence order. In fact, we cannot compare a fourth order method with a second or a third order one. Therefore, we can use some of the presented criteria in this paper to nonlinear solvers from a similar category. Thus, to compare different algorithms, first we categorised nearly all solvers to three different categories, namely, solvers which use Jacobian matrix, the Jacobian free methods, and the frozen Jacobian ones. After that, for any category two qualification criteria were presented which use all parameters of the algorithm. Next, in each category we presented numerical results of our criteria and between

Table 34: Values of FLEI for the methods in the third category to solve Test Problem 1, and $n = 3, 5, 10, 100, 250, 500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.014545335	1.004252776	1.000709959	1.000000995	1.000000065	1.000000008
FJM2	1.016049346	1.005044493	1.000926226	1.000001533	1.000000102	1.000000013
FJM3	1.012234716	1.004045913	1.000822252	1.000001808	1.000000126	1.000000016
FJM4	1.015522513	1.005084692	1.000993066	1.000001881	1.000000128	1.000000016
FJM5	1.010787355	1.003636909	1.000771594	1.000001995	1.000000143	1.000000019
FJM6	1.018043495	1.005905557	1.001153006	1.000002184	1.000000148	1.000000019
FJM7	1.009679663	1.003303160	1.000720806	1.000002116	1.000000155	1.000000020
FJM8	1.014850939	1.005089194	1.001071717	1.000002504	1.000000175	1.000000023
FJM9	1.007906449	1.002756056	1.000628574	1.000002320	1.000000181	1.000000024
FJM10	1.008357551	1.002920927	1.000664704	1.000002370	1.000000182	1.000000024
FJM11	1.008760060	1.003061205	1.000696589	1.000002484	1.000000191	1.000000026

Table 35: Values of CI for the methods in the third category to solve Test Problem 1, and $n = 3, 5, 10, 100, 250, 500$.

Method	$n = 3$	$n = 5$	$n = 10$	$n = 100$	$n = 250$	$n = 500$
NM	1.024189560	1.007322969	1.001284430	1.000001961	1.000000130	1.000000016
FJM2	1.027157647	1.008827634	1.001691602	1.000003022	1.000000204	1.000000026
FJM3	1.018166869	1.006180328	1.001321152	1.000003436	1.000000246	1.000000032
FJM4	1.026501581	1.008983950	1.001825736	1.000003711	1.000000254	1.000000032
FJM5	1.015595728	1.005379209	1.001192887	1.000003713	1.000000276	1.000000036
FJM6	1.030832526	1.010437566	1.002119925	1.000004308	1.000000295	1.000000038
FJM7	1.013771519	1.004789458	1.001086505	1.000003866	1.000000298	1.000000040
FJM8	1.025593593	1.009091827	1.001987595	1.000004941	1.000000348	1.000000045
FJM9	1.011215856	1.003966810	1.000931461	1.000004118	1.000000340	1.000000047
FJM10	1.012145831	1.004317577	1.001013059	1.000004277	1.000000347	1.000000048
FJM11	1.012731936	1.004525080	1.001061662	1.000004482	1.000000364	1.000000050

Table 36: Values of corrected efficiency indices for the methods in the first category to solve Test Problem 1, and $n = 500$.

Method	COREI	CORIOP	CORCEI	CORFLEI	CORCI
M2	0.01717822115	0.01717814641	0.01717814641	0.01717814619	0.01717814641
M3	0.01702987487	0.01702982802	0.01702982802	0.01702982788	0.01702982802
M4	0.00693383982	0.00693382714	0.00693382714	0.00693382708	0.00693382714
M5	0.00570318522	0.00570316952	0.00570316952	0.00570316945	0.00570316952
M6	0.01716096772	0.01716093028	0.01716093028	0.01716093017	0.01716093028
M7	0.00975377848	0.00975374727	0.00975374727	0.00975374721	0.00975374727
M8	0.00859218537	0.00859215793	0.00859215793	0.00859215785	0.00859215793
M9	0.00862912645	0.00862909577	0.00862909577	0.00862909564	0.00862909577
M10	0.00639700555	0.00639698281	0.00639698281	0.00639698271	0.00639698281

our selected algorithms we distinguished the better algorithm. About the numerical results, it must point that CPU time can be computed in seconds, minutes, or hours. Each of these units conduces to different values for the criterion. As a suggestion, we can compute the CPU time in seconds or minutes for small sizes of the problems. But for large values of n, it is better that computes in hours.

The presented criteria will be useful for researchers which work on nonlinear systems. They can use a relevant criterion from the category of their method and present its value to show qualification of their algorithm. A new solver must be able to solve any problem with any size as well. Finally from our presented results, we suggest the followings to solve nonlinear systems and/or presenting any qualification criterion:

(i) To investigate validity of an algorithm, using one criterion is not enough. Hence, applying two or more criteria can show a better validity

Table 37: Values of corrected efficiency indices for the methods the second category to solve Test Problem 1, for $n = 80$.

Method	COREI	CORIOP	CORCEI	CORFLEI	CORCI
JFM2	0.00480303382	0.00480263906	0.00480263826	0.00480263147	0.00480263853
JFM3	0.00473460226	0.00473421312	0.00473421234	0.00473420564	0.00473421260
JFM4	0.00343224182	0.00343201825	0.00343201773	0.00343199931	0.00343201609
JFM5	0.01299267170	0.01298966682	0.01298966622	0.01298961439	0.01298966330
JFM6	0.00640615197	0.00640532507	0.00640532808	0.00640528448	0.00640532203
JFM7	0.00260296971	0.00260282035	0.00260282012	0.00260281419	0.00260281983
JFM8	0.00273099823	0.00273084175	0.00273084139	0.00273083660	0.00273084118
JFM9	0.00456214022	0.00456172058	0.00456172380	0.00456168793	0.00456171754

Table 38: Values of corrected efficiency indices for the methods in the third category to solve Test Problem 1, and $n = 500$.

Method	COREI	CORIOP	CORCEI	CORFLEI	CORCI
FJM2	0.01375748923	0.01375742937	0.013757429370	0.013757429190	0.013757429370
FJM3	0.00777013794	0.00777011669	0.007770116695	0.007770116568	0.007770116691
FJM4	0.01043997478	0.01043991757	0.010439917570	0.010439917400	0.010439917570
FJM5	0.00380709226	0.00380708017	0.003807080169	0.003807080097	0.003807080165
FJM6	0.00631023570	0.00631019558	0.006310195557	0.006310195439	0.006310195557
FJM7	0.00588831078	0.00588828996	0.005888289968	0.005888289843	0.005888289958
FJM8	0.00952827500	0.00952820200	0.009528202002	0.009528201786	0.009528202001
FJM9	0.00370674161	0.00370672555	0.003706725557	0.003706725461	0.003706725546
FJM10	0.00222114704	0.00222113743	0.002221137431	0.002221137374	0.002221137426
FJM11	0.00340749109	0.00340747563	0.003407475635	0.003407475543	0.003407475626

(ii) A qualification criterion depends on the algorithm and type of the nonlinear system. Hence, a suitable criterion must be selected for the problem. This will need more works in the future

(iii) The suitable criterion directly depends on the category of the used algorithm

(iv) The numerical results of this paper have obtained by four, ten, or eleven digits depending on how the results are near together. But it must be noted that, the choice of data type influences the performance of the solvers. For example, using single, double, and quad precision floating-point data. In fact, the performance usually suggests precision evaluation

(v) The practical implementations of methods heavily depend on computer type. For example, specialized computational platforms such as FPGA (Field Programmable Gate Array) or vector computers can speed up the calculations for specific algorithms

(vi) Here, we only considered explicit algorithms. Introducing qualification criteria for using implicit-explicit solvers will be a new research and an attractive work for the future

(vii) The CPU time is a very important index to be in any criterion. Therefore, the corrected classical efficiency indices will be very useful to show the quality of any new nonlinear solver

In summary, for the first time we have presented some new robust qualification criteria which remove the gaps of the previous ones. These criteria consider all parameters of the nonlinear solvers. Further, we have proposed some suggestions for each old criterion by entering the CPU time in its formula. Also, some outlines for the continuation of this work are presented for the future researches.

Data Availability

The data that support the findings of the study are available upon reasonable request from the authors.

Conflicts of Interest

The authors declare that they have no conflict of interest.

References

[1] A. M. Ostrowski, *Solution of Equations and Systems of Equations, Acad*, Press, New York, 1966.

[2] J. F. Traub, *Iterative Methods for the Solution of Equations*, Chelsea Publishing Company, New York, 1982.

[3] M. Grau-Sánchez, A. Grau, and M. Noguera, "On the computational efficiency index and some iterative methods for solving systems of nonlinear equations," *Journal of Computational and Applied Mathematics*, vol. 236, no. 6, pp. 1259–1266, 2011.

[4] H. Montazeri, F. Soleymani, S. Shateyi, and S. S. Motsa, "On a new method for computing the numerical solution of systems of nonlinear equations," *Journal of Applied Mathematics*, vol. 2012, Article ID 751975, 15 pages, 2012.

[5] M. T. Darvishi and A. Barati, "A third-order Newton-type method to solve systems of nonlinear equations," *Applied Mathematics and Computation*, vol. 187, no. 2, pp. 630–635, 2007.

[6] M. T. Darvishi and A. Barati, "Super cubic iterative methods to solve systems of nonlinear equations," *Applied Mathematics and Computation*, vol. 188, no. 2, pp. 1678–1685, 2007.

[7] D. K. R. Babajee, M. Z. Dauhoo, M. T. Darvishi, and A. Barati, "A note on the local convergence of iterative methods based on Adomian decomposition method and 3-node quadrature rule," *Applied Mathematics and Computation*, vol. 200, no. 1, pp. 452–458, 2008.

[8] M. T. Darvishi and A. Barati, "A fourth-order method from quadrature formulae to solve systems of nonlinear equations," *Applied Mathematics and Computation*, vol. 188, no. 1, pp. 257–261, 2007.

[9] F. Soleymani, T. Lotfi, and P. Bakhtiari, "A multi-step class of iterative methods for nonlinear systems," *Optimization Letters*, vol. 8, no. 3, pp. 1001–1015, 2014.

[10] M. Frontini and E. Sormani, "Third-order methods from quadrature formulae for solving systems of nonlinear equations," *Applied Mathematics and Computation*, vol. 149, no. 3, pp. 771–782, 2004.

[11] A. Cordero, J. L. Hueso, E. Martínez, and J. R. Torregrosa, "Increasing the convergence order of an iterative method for nonlinear systems," *Applied Mathematics Letters*, vol. 25, pp. 2369–2374, 2012.

[12] X. Y. Xiao and H. W. Yin, "Increasing the order of convergence for iterative methods to solve nonlinear systems," *Calcolo*, vol. 53, no. 3, pp. 285–300, 2016.

[13] T. Lotfi, P. Bakhtiari, A. Cordero, K. Mahdiani, and J. R. Torregrosa, "Some new efficient multi-point iterative methods for solving nonlinear systems of equations," *International Journal of Computer Mathematics*, vol. 92, no. 9, pp. 1921–1934, 2015.

[14] M. Grau-Sánchez, M. Noguera, and J. M. Gutiérrez, "On some computational orders of convergence," *Applied Mathematics Letters*, vol. 23, no. 4, pp. 472–478, 2010.

[15] M. Grau-Sánchez, A. Grau, and M. Noguera, "Frozen divided difference scheme for solving systems of nonlinear equations," *Journal of Computational and Applied Mathematics*, vol. 235, no. 6, pp. 1739–1743, 2011.

[16] A. R. Amiri, A. Cordero, M. T. Darvishi, and J. R. Torregrosa, "Preserving the order of convergence: low-complexity Jacobian-free iterative schemes for solving nonlinear systems," *Journal of Computational and Applied Mathematics*, vol. 337, pp. 87–97, 2018.

[17] F. I. Chicharro, A. Cordero, N. Garrido, and J. R. Torregrosa, "On the improvement of the order of convergence of iterative methods for solving nonlinear systems by means of memory," *Applied Mathematics Letters*, vol. 104, article 106277, 2020.

[18] J. R. Sharma and H. Arora, "An efficient derivative free iterative method for solving systems of nonlinear equations," *Applicable Analysis and Discrete Mathematics*, vol. 7, no. 2, pp. 390–403, 2013.

[19] A. Cordero, C. Jordán, E. Sanabria, and J. R. Torregrosa, "A new class of iterative processes for solving nonlinear systems by using one divided differences operator," *Mathematics*, vol. 7, no. 9, p. 776, 2019.

[20] X. Wang and X. Fan, "Two efficient derivative-free iterative methods for solving nonlinear systems," *Algorithms*, vol. 9, no. 1, p. 14, 2016.

[21] X. Wang and T. Zhang, "A family of Steffensen type methods with seventh-order convergence," *Numerical Algorithms*, vol. 62, no. 3, pp. 429–444, 2013.

[22] X. Wang, T. Zhang, W. Qian, and M. Teng, "Seventh-order derivative-free iterative method for solving nonlinear systems," *Numerical Algorithms*, vol. 70, no. 3, pp. 545–558, 2015.

[23] U. Qasim, Z. Ali, F. Ahmad, S. Serra-Capizzano, M. Zaka Ullah, and M. Asma, "Constructing frozen Jacobian iterative methods for solving systems of nonlinear equations, associated with ODEs and PDEs using the homotopy method," *Algorithms*, vol. 9, no. 1, p. 18, 2016.

[24] F. Ahmad, E. Tohidi, and J. A. Carrasco, "A parameterized multi-step Newton method for solving systems of nonlinear equations," *Numerical Algorithms*, vol. 71, no. 3, pp. 631–653, 2016.

[25] A. Cordero, J. M. Gutiérrez, A. Magreñán, and J. R. Torregrosa, "Stability analysis of a parametric family of iterative methods for solving nonlinear models," *Applied Mathematics and Computation*, vol. 285, pp. 26–40, 2016.

[26] S. Kouser, S. U. Rehman, F. Ahmad et al., "Generalized Newton multi-step iterative methods GMNp,m for solving system of nonlinear equations," *International Journal of Computer Mathematics*, vol. 95, no. 5, pp. 881–897, 2018.

[27] E. O. Alzahrani, E. S. Al-Aidarous, A. M. M. Younas, and F. Ahmad, "A higher order frozen Jacobian iterative method for solving Hamilton-Jacobi equations," *Journal of Nonlinear Sciences and Applications*, vol. 9, no. 12, pp. 6210–6227, 2016.

[28] F. Ahmad, S. U. Rehman, M. Zaka Ullah et al., "Frozen Jacobian multistep iterative method for solving nonlinear IVPs and BVPs," *Complexity*, vol. 2017, Article ID 9407656, 30 pages, 2017.

[29] S. Qasim, Z. Ali, F. Ahmad, S. Serra-Capizzano, Z. U. Malik, and A. Mahmood, "Solving systems of nonlinear equations when the nonlinearity is expensive," *Computers & Mathematcs with Applications*, vol. 71, no. 7, pp. 1464–1478, 2016.

[30] H. F. Alqahtani, R. Behl, and M. Kansal, "Higher-order iteration schemes for solving nonlinear systems of equations," *Mathematics*, vol. 7, no. 10, p. 937, 2019.