COMPUTATIONAL METHODS IN APPLIED MATHEMATICS, Vol. 10 (2010), No. 4, pp. 368–375

DERIVATIVE-FREE OPTIMAL ITERATIVE METHODS

S.K. KHATTRI[1] AND R.P. AGARWAL[2]

Abstract — In this study, we develop an optimal family of derivative-free iterative methods. Convergence analysis shows that the methods are fourth order convergent, which is also verified numerically. The methods require three functional evaluations during each iteration. Though the methods are independent of derivatives, computational results demonstrate that the family of methods are efficient and demonstrate equal or better performance as compared with many well-known methods and the classical Newton method. Through optimization we derive an optimal value for the free parameter and implement it adaptively, which enhances the convergence order without increasing functional evaluations.

2000 Mathematics Subject Classification: 65D05, 65H05.

Keywords: iterative methods, non-linear equations, derivative-free, convergence order, Newton, efficient, adaptivity, optimization.

1. Introduction

According to Kung's and Traub's conjecture, an optimal iterative method without memory based on $n+1$ evaluations can achieve an optimal convergence order of 2^n [13]. One of the best known optimal second order methods based on two evaluations for solving the equation $f(x)=0$ is the Newton method, which is given as follows (**NM**):

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \qquad n = 0, 1, 2, 3, \ldots, \quad \text{and} \quad |f'(x_n)| \neq 0. \tag{1.1}$$

Methods satisfying the Kung-Traub conjecture (still unproved) are called optimal methods. Therefore, 2^n is the optimal convergence order for methods involving $n+1$ evaluations. The aim of this paper is to develop a multiparameter derivative-free optimal family of fourth-order convergent methods. Additionally, we also optimize the parameters and implement them adaptively to enhance the convergence order.

[1] *Department of Engineering, Stord/Haugesund University College,* Haugesund Norway. E-mail: sanjay.khattri@hsh.no

[2] *Department of Mathematical Sciences, Florida Institute of Technology,* Melbourne, Florida 32901-6975, U.S.A. E-mail: agarwal@fit.edu

2. Methods and convergence analysis

We propose the following new fourth-order derivative-free family of methods:

$$\begin{cases} y_n &= x_n - \dfrac{f(x_n)}{\Phi(x_n)}, \\ x_{n+1} &= y_n - \dfrac{f(y_n)}{\Psi(x_n, y_n)}, \end{cases} \tag{2.1}$$

where

$$\Phi(x_n) = \frac{f(x_n) - f(x_n - \beta\, f(x_n))}{\beta\, f(x_n)},$$

$$\Psi(x_n, y_n) = \frac{\Phi(x_n)}{\left[1 + \dfrac{f(y_n)}{f(x_n)} + \alpha_1 \left(\dfrac{f(y_n)}{f(x_n)}\right)^2 + \dfrac{f(y_n)}{f(x_n - \beta\, f(x_n))} + \alpha_2 \left(\dfrac{f(y_n)}{f(x_n - \beta\, f(x_n))}\right)^2\right]}.$$

Here, α_1, α_2 and β are real parameters with $\beta \neq 0$. For $\beta = 1$, the first step of the preceding family is carried out by the Steffensen method [13]. It can be seen that during each iteration the family requires evaluation of the following three functions: $f(x_n)$, $f(x_n - \beta\, f(x_n))$, and $f(y_n)$. And it is totally free of derivatives. Therefore, according to the Kung-Traub conjecture, the maximum convergence order for the above family of methods is four. For this method, we give a precise analysis of convergence through the following theorem.

Theorem 2.1. *Let the function $f\colon \mathbf{D} \subset \mathbf{R} \mapsto \mathbf{R}$ have a root $\gamma \in \mathbf{D}$ in the open interval $\mathbf{D}$. Furthermore, the first, second, and third derivatives of the function $f(x)$ belong in the open interval $\mathbf{D}$. Then the family of methods* (2.1) *are at least fourth-order convergent only for the real values of $\beta(\neq 0)$, α_1, α_2 and the methods satisfy the error equation*

$$\begin{aligned} e_{n+1} = \frac{c_2}{12 c_1^3} \Big[&\left(-12\,\beta^2 {c_1}^3 + 24\,\beta\, {c_1}^2 - 12\, c_1\right) c_3 + \left(\left(12\,\alpha_1 \beta^3 - 12\,\beta^3\right) {c_1}^3 \right. \\ &\left. + \left(72\,\beta^2 - 36\,\alpha_1 \beta^2\right) {c_1}^2 + \left(12\,\alpha_2 \beta - 120\,\beta + 36\,\alpha_1 \beta\right) c_1 + 60 - 12\,\alpha_2 - 12\,\alpha_1\right) {c_2}^2 \Big] e_n^4 \\ &+ O(e_n^5). \end{aligned} \tag{2.2}$$

where the error after n iterations $e_n = x_n - \gamma$, the constants $c_k = f^{(k)}(\gamma)/k!$ with $k \geqslant 1$, and γ is a simple zero of $f(x)$.

Proof. Using the Taylor expansion of $f(x_n)$ around γ and taking into account $f(\gamma) = 0$, we have

$$f(x_n) = c_1 e_n + c_2 {e_n}^2 + c_3 {e_n}^3 + c_4 {e_n}^4 + O\left({e_n}^5\right). \tag{2.3}$$

Substituting (2.3) into the Taylor expansion of $f(x_n - \kappa\, f(x_n))$ around γ

$$f(x_n - \beta\, f(x_n)) = \sum_{i=1}^{\infty} c_i\, \left(x_n - \gamma - \beta\, f(x_n)\right)^i,$$

we obtain

$$f(x_n - \beta\, f(x_n)) = c_1\,(1 - \beta\, c_1)\, e_n + \left(c_2 - 3\, c_1 \beta\, c_2 + c_2 \beta^2 {c_1}^2\right) {e_n}^2 + O\left({e_n}^3\right). \tag{2.4}$$

From Eqs. (2.1), (2.3), and (2.4) we get

$$\Phi = c_1 + c_2\,(2 - \beta\, c_1)\, e_n + \left(3\, c_3 - 3\, c_1 \beta\, c_3 + c_3 \beta^2 {c_1}^2 - {c_2}^2 \beta\right) {e_n}^2 + O\left({e_n}^3\right). \tag{2.5}$$

Substituting Eqs. (2.3) and (2.5) into (2.1), we obtain

$$y_n - \gamma = \frac{c_2\,(1 - \beta\, c_1)}{c_1} {e_n}^2 + \frac{\left(2\, c_1 c_3 - 2\, {c_2}^2 - 3\, c_3 \beta\, {c_1}^2 + c_3 \beta^2 {c_1}^3 + 2\, {c_2}^2 \beta\, c_1 - {c_2}^2 \beta^2 {c_1}^2\right)}{{c_1}^2} {e_n}^3 + O\left({e_n}^4\right). \tag{2.6}$$

From the Taylor expansion of $f(y)$ around γ and using the preceding equation, we obtain

$$f(y_n) = c_2\,(1 - \beta\, c_1)\, {e_n}^2 + \frac{\left(2\, c_1 c_3 - 2\, {c_2}^2 - 3\, c_3 \beta\, {c_1}^2 + c_3 \beta^2 {c_1}^3 + 2\, {c_2}^2 \beta\, c_1 - {c_2}^2 \beta^2 {c_1}^2\right)}{c_1} {e_n}^3 + O\left({e_n}^4\right). \tag{2.7}$$

From (2.1), (2.3), (2.4), (2.5), and (2.7), we get

$$\Psi = c_1 - \frac{\left(\left(-\beta\, {c_1}^2 + c_1\right) c_3 + \left(\left(-\beta^2 + \alpha_1 \beta^2\right) {c_1}^2 + \left(-2\, \alpha_1 \beta + 6\, \beta\right) c_1 + \alpha_2 - 6 + \alpha_1\right) {c_2}^2\right) {e_n}^2}{c_1} + O\left({e_n}^3\right). \tag{2.8}$$

Finally, substituting Eqs. (2.6), (2.7), and (2.8) into the second step of the proposed method (2.1), we obtain the error equation

$$\begin{aligned} e_{n+1} = \frac{c_2}{12 c_1^3} \Big[& \left(-12\, \beta^2 {c_1}^3 + 24\, \beta\, {c_1}^2 - 12\, c_1\right) c_3 + \left(\left(12\, \alpha_1 \beta^3 - 12\, \beta^3\right) {c_1}^3 \right. \\ & \left. + \left(72\, \beta^2 - 36\, \alpha_1 \beta^2\right) {c_1}^2 + \left(12\, \alpha_2 \beta - 120\, \beta + 36\, \alpha_1 \beta\right) c_1 + 60 - 12\, \alpha_2 - 12\, \alpha_1\right) {c_2}^2 \Big] e_n^4 \\ & + O(e_n^5). \end{aligned} \tag{2.9}$$

Therefore, the proposed method is at least forth order for any real choice of the parameters $\beta(\neq 0)$, α_1 and α_2. This completes our proof. □

In the next section, the proposed method is compared with many methods from the published literature ascertaining the efficient disposition of the contribution. Furthermore, we improve the convergence order of the method, through optimization and adaptivity, without increasing the functional estimates.

3. Numerical examples

If the convergence order ξ of the iterative method is defined as

$$\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|^{\xi}} = c \neq 0,$$

then the approximate computational order of convergence (COC) is

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

Computations are done in the programming language C^{++}. Scientific computations in many areas of science and engineering demand a very high degree of numerical precision [2,8]. For applications of high-precision computations in experimental mathematics and physics, we refer to [1] and references therein. For performing high-precision computation, we use the high-precision C^{++} library ARPREC [2]. The ARPREC library supports an arbitrarily high level of numeric precision [2]. In the program, the precision in decimal digits is set by the command "mp::mp init(2005)" [2]. For convergence, it is required that $|x_{n+1}-x_n| < \epsilon$ and $|f(x_n)| < \epsilon$. Here, $\epsilon = 10^{-310}$. We have tested many iterative methods for the following functions:

$f_0(x) = \sin(x) - x/100,$	γ=0.0.	$f_1(x) = x^3 + 4\,x^2 - 10,$	$\gamma\approx$1.365.
$f_2(x) = \tan^{-1}(x)\,,$	γ=0.0.	$f_3(x) = x^4 + \sin\big(\pi/x^2\big) - 5,$	$\gamma=\sqrt{2}$.
$f_4(x) = \exp\big(-x^2 + x + 2\big) - 1,$	$\gamma = -1.0$.	$f_5(x) = \cos(x)^2 - x/5$	$\gamma\approx$2.3.
$f_6(x) = 4/5x - 1/6x^3,$	γ=0.0.	$f_7(x) = 1/3x^4 - x^2 - 1/3x + 1,$	γ=1.0.

In the proposed method (2.1) (**PM**), we select: $\beta = 10^{-20}$, $\alpha_1 = 1$ and $\alpha_2 = 1$. The computational results are reported in Table 3.1. Table 3.1 presents the number of functional estimates of the COC during the second iterative step for various methods. In Table 3.1, the various methods are abbreviated as follows: **CM** Chebyshev method [10, 13]; **HM**, Halley method [10,13]; **EM**, Euler method also referred to as the Cauchy method [6,10,13]; **NM**, Newton iterative method [10,13]; **RWB**, method proposed by Ren et al. [11]; **NETA**, method proposed by Neta et al. [9]; **CH**, method developed by Chun et al. [3]; **WKL**, method developed by Wang et. al. [14], **PM**, proposed method (2.1); and **SG**, method constructed by Sharma et al. [12]. Free parameters are randomly selected as follows: for **RWB** $a = b = c = 1$, (**CH**) $\beta = 1$, in **WKL** $\alpha = \beta = 1$, in **NETA** $a = 10$, in **SG** $p = m = -5$.

An optimal iterative method must require the least number of functional evaluations for convergence. In Table 3.1, the methods that require the least number of functional evaluations are in bold face. We see that the methods used in this paper work better than those presented in the literature. And our methods also offer the advantage of being completely free of derivatives.

3.1. Convergence enhancement through optimization and adaptivity

To find the optimal value of the free parameters, we optimize the absolute value of the asymptotic constant in the error equation (2.2). We have found that for $\beta = c_1^{-1}$ the

Table 3.1. **(Number of functional estimates of the COC) for various iterative methods**

$f(x)$	x_0	**HM**	**CM**	**EM**	**NM**	**RWB**	**NETA**	**CH**	**WKL**	**PM**	**SG**
$f_0(x)$	0.9	$(27,3)$	$(30,3)$	$(27,3)$	$(20,2)$	$(20,6)$	$(20,6)$	$(20,6)$	$(20,6)$	$(\mathbf{18},\mathbf{4})$	$(21,4)$
$f_1(x)$	1.0	$(36,3)$	$(39,3)$	$(39,3)$	$(20,2)$	$(20,6)$	$(20,6)$	$(20,6)$	$(20,6)$	$(\mathbf{18},\mathbf{4})$	$(21,4)$
$f_2(x)$	0.5	$(21,3)$	$(21,3)$	$(21,3)$	$(18,2)$	$(20,6)$	$(\mathbf{16},\mathbf{7})$	$(\mathbf{16},\mathbf{7})$	$(20,6)$	$(18,5)$	$(18,5)$
$f_3(x)$	0.85	div	$(54,3)$	$(24,3)$	$(20,2)$	$(20,6)$	$(20,6)$	$(20,6)$	$(20,6)$	$(\mathbf{18},\mathbf{4})$	$(\mathbf{18},\mathbf{4})$
$f_4(x)$	−0.45	$(24,3)$	$(24,3)$	$(21,3)$	$(\mathbf{20},\mathbf{2})$	$(\mathbf{20},\mathbf{6})$	$(\mathbf{20},\mathbf{6})$	$(\mathbf{20},\mathbf{6})$	$(\mathbf{20},\mathbf{6})$	$(21,4)$	$(21,4)$
$f_5(x)$	2.5	$(30,3)$	$(27,3)$	$(30,3)$	$(24,2)$	$(24,6)$	$(24,6)$	$(24,6)$	$(24,6)$	$(\mathbf{18},\mathbf{4})$	$(24,4)$
$f_6(x)$	0.5	$(21,3)$	$(21,3)$	$(21,3)$	$(20,2)$	$(20,6)$	$(20,6)$	$(20,6)$	$(20,6)$	$(\mathbf{18},\mathbf{4})$	$(\mathbf{18},\mathbf{4})$
$f_7(x)$	0.5	$(24,3)$	$(27,3)$	$(24,3)$	$(26,2)$	$(20,6)$	$(20,6)$	$(20,6)$	$(20,6)$	$(\mathbf{18},\mathbf{4})$	$(\mathbf{18},\mathbf{4})$

asymptotic error constant vanishes yielding thereby a fifth-order iterative method, since we cannot evaluate c_1 a priori. Consequently, we define the parameter β adaptively as follows:

$$\beta_{n+1} = \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}, \quad \text{with} \quad n \geqslant 1. \tag{3.1}$$

For the first iterate, we choose $\beta_1 = 10^{-20}$. And $\alpha_1 = \alpha_2 = 1$. For evaluating β_m with $m \geqslant 2$ by the preceding equation, we use the previous two iterates, and it does not increase functional evaluations. We expect that the computational order of convergence to be 5. The computational results are reported in Table 3.2.

Table 3.2. $|x_{n+1} - x_n|$ **with** $n \geqslant 1$ **and COC produced, together with the number of functional evaluations (NFE) required, by the contributed method with optimal parameters**

$f_0(x)$		$f_2(x)$		$f_4(x)$		$f_6(x)$	
$\lvert x_{n+1} - x_n\rvert$	**COC**	$\lvert x_{n+1} - x_n\rvert$	**COC**	$\lvert x_{n+1} - x_n\rvert$	**COC**	$\lvert x_{n+1} - x_n\rvert$	**COC**
7×10^{-1}	$***$	5×10^{-1}	$***$	4×10^{-1}	$***$	5×10^{-1}	$***$
1×10^{-1}	8.4	5×10^{-3}	6.3	3×10^{-2}	4.0	1×10^{-3}	6.3
1×10^{-7}	5.5	3×10^{-15}	5.6	5×10^{-7}	4.4	5×10^{-20}	5.6
9×10^{-42}	5.7	4×10^{-84}	5.7	6×10^{-28}	4.5	2×10^{-112}	5.7
$1 \times 10^{-1,347}$	$***$	3×10^{-477}	$***$	2×10^{-121}	4.5	2×10^{-639}	$***$
$***$	$***$	$***$	$***$	9×10^{-537}	$***$	$***$	$***$
NFE = 15		**NFE = 15**		**NFE = 18**		**NFE = 15**	

To our surprise, we notice in Table 3.2 that for the functions $f_0(x)$, $f_2(x)$ and $f_6(x)$; the method is displaying a higher order of convergence than expected. Therefore, we derived the error equation for $\beta = c_1^{-1}$ and obtain

$$e_{n+1} = \left(\frac{c_2}{c_1}\right)^4 (1 - \alpha_2)\, {e_n}^5 + O\left({e_n}^6\right). \tag{3.2}$$

In the above error equation, we notice that for $\alpha_2 = 1$ the theoretical order of convergence for the method is six.

From Tables 3.1 and 3.2, we see that the proposed family of methods work better than the literature methods for all eight functions. Consequently, the proposed methods are more efficient than those given in the literature.

3.2. Robustness of numerical methods with respect to the initialization

Let us find the zeros of the function

$$f(x) = x^3 + 3\,x^2 - 10, \tag{3.3}$$

for various initializations. Computational results are reported in Table 3.3. In the proposed method (2.1), for the first iterate $\beta = 1.0$, while for the successive iterates β is computed using Eq. (3.1).

Table 3.3. **(number of functional evaluations, COC) for various iterative methods for the function** (3.3)

x_0	**HM**	**CM**	**EM**	**NM**	**RWB**	**NETA**	**CH**	**WKL**	**PM**	**SG**
0.0	div	div	div	div	div	div	div	div	$(\mathbf{24}, \mathbf{4.5})$	div
-2.0	div	div	div	div	div	div	div	div	$(\mathbf{24}, \mathbf{4.5})$	div
10000	$(63, 3)$	$(66, 2)$	$(63, 3)$	$(60, 2)$	$(40, 6)$	$(40, 6)$	$(40, 6)$	$(40, 6)$	$(51, 4.5)$	$(93, 4)$

Now we find the zeros of the function

$$f(x) = \cos(x)^2 - x/5, \tag{3.4}$$

for various initializations. Computational results are reported in Table 3.4. In the proposed method (2.1), for the first iterate $\beta = 1.0$, while for the successive iterates β is computed using Eq. (3.1).

Table 3.4. **(number of functional evaluations, COC) for various iterative methods for the function** (3.4)

x_0	**HM**	**CM**	**EM**	**NM**	**RWB**	**NETA**	**CH**	**WKL**	**PM**	**SG**
-0.1	div	div	div	div	div	div	div	div	$(\mathbf{33}, \mathbf{4.5})$	div
0.0	$(42, 3)$	$(42, 3)$	$(42, 3)$	$(40, 2)$	$(40, 6)$	$(44, 6)$	$(40, 6)$	$(40, 6)$	$(\mathbf{30}, \mathbf{4.5})$	$(42, 4)$
-10000	div	div	div	div	div	div	div	div	$(\mathbf{24}, \mathbf{4.5})$	div
10000	div	div	div	div	div	div	div	div	$(\mathbf{21}, \mathbf{4.5})$	div

From the computational results reported in Tables 3.3 and 3.4, we see that the developed methods display robustness with respect to the initialization.

3.3. Matlab and the developed method

Here we compare Matlab's "**fzero()**" and the developed method (2.1) for finding the zeros of the following functions:

$$f(x) = 1/(1 + x^2) - 1, \qquad \gamma = 0. \tag{3.5}$$

$$f(x) = \exp(x^4 + x^2 + 1) - \exp(1), \qquad \gamma = 0. \tag{3.6}$$

Method (2.1) is implemented (without any stopping condition) in Matlab as follows:

Listing 3.1. **Matlab implemention of the developed method** (2.1)

```
clear all; format short g; x = 0.05; beta =
1.0; alpha1 = 1.0; alpha2 = 1.0; for n = 1:5
    fo = f(x);
    phi = (f(x) - f(x-beta*f(x)))/ ( beta*f(x));
    y = x-f(x)/phi;
    t1 = ( f(y)/f(x));
    t2 = f(y)/f(x-beta*f(x));
    psi = phi/( 1.0 + t1 + alpha1 * t1 * t1 + t2 + alpha2 *t2*t2);
    x = y-(f(y)/(psi)),
end;
```

where the function subprogram is

Listing 3.2. **Matlab subprogram for the function** (3.5)

```
function [f] = f(x)
f = 1.0/(1.0+x*x) - 1.0;
```

on execution of these commands, the successive iterates x_i for the function (3.5) are

$$0.015162, \quad 0.0045339, \quad 0.001349, \quad 0.00040075, \quad 0.000119.$$

We observe that these iterates converge, though very slowly, towards zero. Zeros of function (3.5) are computed by Matlab's "**fzero()**" as

Listing 3.3. **Matlab commands for finding zero of a function**

```
>> f= @(x) 1.0/(1.0+x*x) - 1.0;
>> options = optimset('display', 'iter');
>> fzero(f,0.05,options)
```

Upon executing the preceding commands the Matlab iterates indefinitely. Now we apply Listing 1 to find the zeros of function (3.6). The successive iterates x_i are

$$0.013819, \quad 0.0040255, \quad 0.0011885, \quad 0.00035227, \quad 0.00010453.$$

We observe that these iterates converge, though very slowly, towards zero of function (3.6). While upon executing the Matlab commands given in Listing 3 for finding the zeros of function (3.6), we obtain

```
ans =

   NaN
```

We observe that, for the preceding two examples, the Matlab "**fzero()**" command is divergent, while our method is convergent. However, in a general context, the Matlab command can be excellent.

Acknowledgments. We are grateful to the reviewers for constructive remarks and hints that have improved our work.

References

1. D. H. Bailey and J. M. Borwein, *High-Precision Computation and Mathematical Physics*, XII Advanced Computing and Analysis Techniques in Physics Research, LBNL-2160E, http://crd.lbl.gov/~dhbailey/dhbpapers/.

2. D. H. Bailey, Y. Hida, X. S. Li, and B. Thompson, *ARPREC(C++/Fortran-90 arbitrary precision package)*, Available at: `http://crd.lbl.gov/~dhbailey/mpdist/`.

3. C. Chun and Y. Ham, *Some sixth-order variants of Ostrowski root-finding methods*, Appl. Math. Comput., **193** (2003), pp. 389–394.

4. P. Jarratt, *Some fourth order multipoint methods for solving equations*, Math. Comp., **20** (1966), pp. 434–437.

5. R. F. King, *A family of forth-order methods for nonlinear equations*, SIAM J. Numer. Anal., **10** (1976), pp. 876–879.

6. J. Kou, *Some variants of Cauchy's method with accelerated fourth-order convergence*, J. Comput. Appl. Math., **213** (2008), pp. 71–78.

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

8. X. Li, C. Mu, J. Ma, and C. Wang, *Sixteenth order method for nonlinear equations*, Appl. Math. Comput., **215** (2009), no. 10, pp. 3769–4054 .

9. B. Neta, *A Sixth-Order Family of Methods for Nonlinear Equations*, Int. J. Comput. Math., **7** (1979), pp. 157–161.

10. A. M. Ostrowski, *Solutions of Equations and System Equations*, Academic Press, New Youk, 1960.

11. H. Ren, Q. Wu, and W. Bi, *New variants of Jarratt's method with sixth-order convergence*, Numer. Algorithms, **52** (2009), pp. 585-603.

12. J. R. Sharma and R. K. Goyal, *Fourth-order derivative-free methods for solving non-linear equations*, Intern. J. Computer Math., **83** (2006), pp. 101–106.

13. J. F. Traub, *Iterative methods for the solution of equations*, Chelsea Publishing Company, New Youk, 1977.

14. X. Wang, J. Kou, and Y. Li, *A variant of Jarratt method with sixth-order convergence*, Appl. Math. Comput., **204** (2008), pp. 14–19.