Chapter 10
On Convergence and Efficiency in the Resolution of Systems of Nonlinear Equations from a Local Analysis

Miquel Grau-Sánchez and Miquel Noguera

Abstract The aim of this chapter is to provide an overview of theoretical results and numerical tools in some iterative schemes to approximate solutions of nonlinear equations. Namely, we examine the concept of iterative methods and their local order of convergence, numerical parameters that allow us to assess the order, and the development of inverse operators (derivative and divided differences). We also provide a detailed study of a new computational technique to analyze efficiency. Finally, we end the chapter with a consideration of adaptive arithmetic to accelerate computations.

10.1 Iteration Functions

Many problems in computational sciences and other disciplines can be formulated by means of an equation like the following

$$\mathscr{F}(x) = 0\,, \tag{10.1}$$

where $\mathscr{F} : D \subset \mathbb{X} \longrightarrow \mathbb{Y}$ is a continuous operator defined on a nonempty convex subset D of a Banach space $\mathbb{X}$ with values in a Banach space $\mathbb{Y}$. We face the problem of approximating a local unique solution $\alpha \in \mathbb{X}$ of Eq. (10.1). Since the exact solution of this equation can rarely be found, then we need to use iterative techniques

M. Grau-Sánchez (✉)
Department of Mathematics, Technical University of Catalonia · BarcelonaTech, Campus Nord, Edifici Omega, 08034 Barcelona, Spain
e-mail: miquel.grau@upc.edu

M. Noguera
Department of Mathematics, Technical University of Catalonia · BarcelonaTech, Edifici TR5 (ESEIAAT), 08222 Terrassa, Spain
e-mail: miquel.noguera@upc.edu

© Springer International Publishing Switzerland 2016
S. Amat, S. Busquier (eds.), *Advances in Iterative Methods for Nonlinear Equations*, SEMA SIMAI Springer Series 10,
DOI 10.1007/978-3-319-39228-8_10

to approximate it to the desired precision from one or several initial approximations. This procedure generates a sequence of approximations of the solution.

Traub [41] includes a classification of iteration functions, according to the information that is required to carry them out. We build up a sequence $\{x_n\}_{n\geq 1}$ in a Banach space $\mathbb{X}$ using the initial conditions $x_{-k},\dots,x_{-1},x_0$, $0\leq k\leq j-1$. Traub's classification of iteration functions is the following.

Type I. Term x_{n+1} is obtained using only the information at x_n and no other information. That is, given $x_0\in D$ we have

$$x_{n+1}=\Phi(x_n)\,,\quad n\geq 0\,. \tag{10.2}$$

The function Φ is called a *one-point iteration function* and Eq. (10.2) is called the one-point iterative method without memory.

Type II. Term x_{n+1} is obtained using the information at x_n and previous information at $x_{n-1}\,,\dots\,,x_{n-j}\in D$. Namely,

$$x_{n+1}=\Phi(x_n\,;x_{n-1}\,,\dots\,,x_{n-j})\,,\quad n\geq 0\,,\quad j\geq 1\,. \tag{10.3}$$

Function Φ is called a *one-point iteration function with memory* and Eq. (10.3) is called a one-point iterative method with memory (j points). The semicolon in (10.3) is written to distinguish the information provided by the new data from the information that was previously used.

Type III. Term x_{n+1} is determined using new information at x_n and previous information at $\varphi_1=\varphi_1(x_n)$, $\varphi_2=\varphi_2(\varphi_1,x_n),\dots,$ $\varphi_r=\varphi_r(\varphi_{r-1},\dots,\varphi_1,x_n)\in D$, $r\geq 1$. That is,

$$x_{n+1}=\Phi\left(x_n\,,\varphi_1\,,\dots\,,\varphi_r\right)\,,\quad n\geq 0\,,\quad r\geq 1\,. \tag{10.4}$$

Here, function Φ is called a *multipoint iteration function* without memory and Eq. (10.4) is called a multipoint iterative method without memory (r steps).

Type IV. Term x_{n+1} is obtained from new information at x_n and previous information at

$$\begin{aligned}\varphi_1&=\varphi_1(x_n\,;x_{n-1},\dots\,,x_{n-j}),\\ &\vdots\\ \varphi_r&=\varphi_r(x_n,\varphi_1,\dots,\varphi_{r-1}\,;x_{n-1},\dots\,,x_{n-j}).\end{aligned}$$

Namely,

$$x_{n+1}=\Phi\left(x_n\,,\varphi_1\,,\dots\,,\varphi_r;x_{n-1}\,,\dots\,,x_{n-j}\right)\,,\quad n\geq 0\,,\quad r\geq 1\,,\quad j\geq 1. \tag{10.5}$$

Function Φ is called a *multipoint iteration function with memory* and (10.5) is called a multipoint iteration method with memory (r steps and j points).

10.1.1 One-Dimensional Case

In particular, when the Banach spaces $\mathbb{X} = \mathbb{Y} = \mathbf{R}$, we have to solve the most simple, classical nonlinear problem. Namely, let $f : I \subseteq \mathbf{R} \to \mathbf{R}$ be a nonlinear function. We have to approximate a simple root α of the equation

$$f(x) = 0, \tag{10.6}$$

where I is a neighborhood of α. An approximation of α is usually obtained by means of an iterative function of type **I, II, III** or **IV**, defined in (10.2), (10.3), (10.4) or (10.5) whereby a sequence $\{x_n\}_{n\geq 1}$ is considered that converging converges to α.

Definition 1 The sequence $\{x_n\}$ is said to converge to α with order of convergence $\rho \in \mathbf{R}$, $\rho \geq 1$, if there exists a positive real constant $C \neq 0$ and $C \neq \infty$ such that

$$\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|^{\rho}} = C, \tag{10.7}$$

where $e_n = x_n - \alpha$ is the *error* in the nth iterate, and the constant C is called the *asymptotic error constant* (see [41]).

The local order of convergence of an iterative method in a neighborhood of a root is the order of its corresponding sequence generated by the iterative function and the corresponding initial approximations. For iterative methods without memory, the local order is a positive integer. The convergence is said to be linear if $\rho = 1$, quadratic if $\rho = 2$, cubic if $\rho = 3$, and, in general, superlinear if $\rho > 1$, superquadratic if $\rho > 2$, and so on.

The one-point iterative method without memory (10.7) can be written as

$$e_{n+1} = C\, e_n^{\rho} + O\big(e_n^{\rho+1}\big),\ n \geq n_0. \tag{10.8}$$

The expression (10.8) is called the *error difference equation* for the one-point iterative method. Note that the higher order terms in (10.8) are powers of $\rho + 1$.

For the one-point iterative method without memory, an approximation of the number of correct decimal places in the nth iterate, d_n, is given by

$$d_n = -\log_{10} |x_n - \alpha|. \tag{10.9}$$

From (10.8), for n large enough we have $e_{n+1} \approx C\, e_n^{\rho}$, which using logarithms yields

$$d_{n+1} \approx -\log_{10} C + \rho \cdot d_n, \tag{10.10}$$

from which follows

$$d_{n+1} \approx \rho \cdot d_n. \tag{10.11}$$

This means that, in each iteration, the number of correct decimal places is approximately the number of correct decimals in the previous iteration multiplied by the local error.

This is in agreement with Wall's definition [42]. That is, the local order of convergence of a one-point iteration function indicates the rate of convergence of the iteration method. Then, Wall defines the order ρ of the iteration formula by

$$\rho = \lim_{n\to\infty} \frac{\log|e_{n+1}|}{\log|e_n|} = \lim_{n\to\infty} \frac{d_{n+1}}{d_n}. \tag{10.12}$$

This expression will be used later on when we define some parameters employed in the computation of the local order of convergence of an iterative method.

For the one-point iterative method with memory (10.3) the error difference equation is

$$e_{n+1} = C e_n^{a_1} e_{n-1}^{a_2} \dots e_{n-j+1}^{a_j} + o(e_n^{a_1} e_{n-1}^{a_2} \dots e_{n-j+1}^{a_j}), \tag{10.13}$$

where a_k are nonnegative integers for $1 \le k \le j$ and $o(e_n^{a_1} e_{n-1}^{a_2} \dots e_{n-j+1}^{a_j})$ represents terms with high order than the term $e_n^{a_1} e_{n-1}^{a_2} \dots e_{n-j+1}^{a_j}$. In this case, the order of convergence ρ is the unique real positive root of the indicial polynomial (see [27, 28, 40, 41]) of the error difference equation (10.13) given by

$$p_j(t) = t^j - a_1 t^{j-1} - \dots - a_{j-1} t - a_j. \tag{10.14}$$

Notice that $p_j(t)$ in (10.14) has a unique real positive root ρ on account of Descartes's rule of signs. Moreover, we can write $e_{n+1} = C e_n^{\rho} + o(e_n^{\rho})$.

10.1.2 Multidimensional Case

When the Banach spaces $\mathbb{X} = \mathbb{Y} = \mathbf{R}^m$ we have to solve a system of nonlinear equations. Namely, let $F : D \subset \mathbf{R}^m \longrightarrow \mathbf{R}^m$ be a nonlinear function and $F \equiv (F_1, F_2, \dots, F_m)$ with $F_i : D \subseteq \mathbf{R}^m \to \mathbf{R}$, $i = 1, 2, \dots, m$, where D is an open convex domain in $\mathbf{R}^m$, so that we have to approximate a solution $\alpha \in D$ of the equation $F(x) = 0$.

Starting with a given set of initial approximations of the root α, the iteration function $\Phi : D \longrightarrow D$ of type **I**, **II**, **III** or **IV** is defined by (10.2), (10.3), (10.4) or (10.5), whereby a sequence $\{x_n\}_{n\le 1}$ XXX is considered to converge to α.

Definition 2 The sequence $\{x_n\}$ converges to α with *an order of convergence* of at least $\rho \in \mathbf{R}$, $\rho \ge 1$, if there is a positive real constant $0 < C < \infty$ such that

$$\|e_{n+1}\| \le C \|e_n\|^{\rho}, \tag{10.15}$$

where $e_n = x_n - \alpha$ is the error in the nth iterate, and the constant C is called the asymptotic error constant (see [41]). Here the norm used is the maximum norm.

The local order of convergence of an iterative method in a neighborhood of a root is the order of the corresponding sequence generated (in $\mathbf{R}^m$) by the iterative function Φ and the corresponding initial approximations.

Without using norms, a definition of the local order of convergence for the one-step iterative method without memory can be considered as follows. The local order of convergence is $\rho \in \mathbb{N}$ if there is an ρ–linear function $C \in \mathscr{L}\left(\mathbf{R}^m \times \overset{\rho}{\cdots} \times \mathbf{R}^m, \mathbf{R}^m\right) \equiv \mathscr{L}_\rho\left(\mathbf{R}^m, \mathbf{R}^m\right)$ such that

$$e_{n+1} = C\, e_n^{\rho} + O\big(e_n^{\rho+1}\big),\ n \geq n_0 \tag{10.16}$$

where e_n^ρ is $(e_n, \overset{\rho}{\cdots}, e_n) \in \mathbf{R}^m \times \overset{\rho}{\cdots} \times \mathbf{R}^m$. When $0 < C < \infty$ exists for some $\rho \in [1, \infty)$ from (10.15), then ρ is the R-order of convergence of the iterative method defined by Ortega and Rheinboldt [27]. Moreover, the local order ρ of (10.16) is also the R-order of convergence of the method.

For the one-point iterative method with memory, the error difference equation can be expressed by

$$e_{n+1} = C e_n^{a_1} e_{n-1}^{a_2} \dots e_{n-j+1}^{a_j} + o(e_n^{a_1} e_{n-1}^{a_2} \dots e_{n-j+1}^{a_j}), \tag{10.17}$$

where $C \in \mathscr{L}_{a_1+\cdots+a_j}\left(\mathbf{R}^m, \mathbf{R}^m\right)$ and a_k are nonnegative integers for $1 \leq k \leq j$.

As in the one-dimensional case, we can write the equation associated with (10.17), $p_j(t) = t^j - a_1 t^{j-1} - \cdots - a_{j-1} t - a_j = 0$. If we apply Descartes's rule to the previous polynomial, there is a unique real positive root ρ that coincides with the local order of convergence (see [27, 40]).

10.2 Computational Estimations of the Order

After testing the new iterative methods, we need to check the theoretical local order of convergence. The parameter Computational Order of Convergence (COC) is used in most studies published after Weerakoon and Fernando [43]. This parameter can only be used when the root α is known. To overcome this problem, the following parameters have been introduced:

Approximated Computational Order of Convergence (ACOC) by Hueso et al. [22],
Extrapolated Computational Order of Convergence (ECOC) by Grau et al. [12],
and Pétkovic Computational Order of Convergence (PCOC) by Petković [29].

The paper by Grau et al. [14] examines the relations between the parameters COC, ACOC and ECOC and the theoretical convergence order of iterative methods without memory.

Subsequently, using Wall's definition of the order (10.12), four new parameters (CLOC, ACLOC, ECLOC and PCLOC) were given in [19] to check this order. Note that the last three parameters do not require knowledge of the root.

Generalizations of COC, ACOC and ECOC from the one-dimensional case to the multi-dimensional one can be found in [15]. They will be presented in detail in the sequel.

10.2.1 Computational Order of Convergence and Its Variants

Let $\{x_n\}_{n\geq 1}$ be a sequence of real numbers converging to α. It is obtained by carrying out an iteration function in $\mathbf{R}$, starting with an initial approximation x_0, or $x_{-j+1},\ldots x_0$, of the root α of (10.6). Let $\{e_n\}_{n\geq 1}$ be the sequence of errors given by $e_n = x_n - \alpha$. If functions (10.2)–(10.5) have local order of convergence ρ, then from (10.10) we have

$$\log|e_n| \approx \rho\cdot\log|e_{n-1}| + \log C,$$

$$\log|e_{n-1}| \approx \rho\cdot\log|e_{n-2}| + \log C.$$

By subtracting the second expression from the first one we get

$$\rho \approx \frac{\log|e_n\ /\ e_{n-1}|}{\log|e_{n-1}\ /\ e_{n-2}|}\,. \tag{10.18}$$

This expression is the same as that described in papers by Weerakoon and Fernando [43], and Jay [23].

Definition 3 The values $\overline{\rho}_n$ (COC), $\widehat{\rho}_n$ (ACOC), $\widetilde{\rho}_n$ (ECOC) and $\breve{\rho}_n$ (PCOC) are defined by

$$\overline{\rho}_n = \frac{\log|e_n\ /e_{n-1}|}{\log|e_{n-1}\ /e_{n-2}|},\quad e_n = x_n - \alpha\,,\quad n\geq 3, \tag{10.19}$$

$$\widehat{\rho}_n = \frac{\log\ |\hat{e}_n\ /\hat{e}_{n-1}|}{\log\ |\hat{e}_{n-1}\ /\hat{e}_{n-2}|},\quad \hat{e}_n = x_n - x_{n-1},\quad n\geq 4, \tag{10.20}$$

$$\widetilde{\rho}_n = \frac{\log|\tilde{e}_n\ /\ \tilde{e}_{n-1}|}{\log|\ \tilde{e}_{n-1}\ /\tilde{e}_{n-2}|},\quad \tilde{e}_n = x_n - \widetilde{\alpha}_n,\quad n\geq 5, \tag{10.21}$$

$$\widetilde{\alpha}_n = x_n - \frac{(\delta x_{n-1})^2}{\delta^2 x_{n-2}}, \quad \delta x_n = x_{n+1} - x_n,$$

$$\breve{\rho}_n = \frac{\log|\breve{e}_n|}{\log|\breve{e}_{n-1}|}, \quad \breve{e}_n = \frac{f(x_n)}{f(x_{n-1})}, \quad n \geq 2. \tag{10.22}$$

Note that the first variant of COC, ACOC, involves the parameter $\hat{e}_n = x_n - x_{n-1}$ and the second variant ECOC is obtained using Aitken's extrapolation procedure [1]. That is, from the iterates x_{n-2}, x_{n-1}, x_n we can obtain the approximation $\widetilde{\alpha}_n$ of the root α.

Sequences $\{\widetilde{\rho}_n\}_{n\geq 5}$ and $\{\widehat{\rho}_n\}_{n\geq 4}$ converge to ρ. The details of the preceding claim can be found in [14], where the relations between the error e_n and $\tilde{e}_n$ and $\hat{e}_n$ are also described.

From a computational viewpoint, ACOC has the least computational cost, followed by PCOC. Inspired by (10.12) given in [42], in our study [19] we present four new parameters that will be described in the following section.

10.2.2 New Parameters to Compute the Local Order of Convergence

A. Definitions Given the sequence $\{x_n\}_{n\geq 1}$ of iterates converging to α with order ρ, we consider the sequences of errors $e_n = x_n - \alpha$ and error parameters $\hat{e}_n = x_n - x_{n-1}$, $\tilde{e}_n = x_n - \widetilde{\alpha}_n$ and $\breve{e}_n = \frac{f(x_n)}{f(x_{n-1})}$ defined previously in (10.20), (10.21), (10.22). From the preceding, we define the following sequences $\{\overline{\lambda}_n\}_{n\geq 2}$ (CLOC), $\{\widehat{\lambda}_n\}_{n\geq 3}$ (ACLOC), $\{\widetilde{\lambda}_n\}_{n\geq 4}$ (ECLOC) and $\{\breve{\lambda}_n\}_{n\geq 2}$ (PCLOC):

$$\overline{\lambda}_n = \frac{\log|e_n|}{\log|e_{n-1}|}, \; \widehat{\lambda}_n = \frac{\log|\hat{e}_n|}{\log|\widehat{e}_{n-1}|}, \; \widetilde{\lambda}_n = \frac{\log|\tilde{e}_n|}{\log|\widetilde{e}_{n-1}|}, \; \breve{\lambda}_n = \frac{\log|f(x_n)|}{\log|f(x_{n-1})|}. \tag{10.23}$$

Note the analogy between $\overline{\lambda}_n$ and the definitions given by Wall in [42] and by Tornheim in [40]. To obtain $\overline{\lambda}_n$, we need knowledge of α; while to obtain $\widehat{\lambda}_n$, $\widetilde{\lambda}_n$ and $\breve{\lambda}_n$ we do not. The new parameters CLOC, ACLOC, ECLOC and PCLOC have a lower computational cost than their predecessors. A detailed description of their convergence can be found in our studies [19] and [20].

B. Relations Between Error and Error Parameters In the case of iterative methods to obtain approximates of the root α of $f(x) = 0$, where $f : I \subset \mathbf{R} \to \mathbf{R}$, the error difference equation is given by

$$e_{n+1} = C\, e_n^{\rho}\left(1 + O(e_n^{\sigma})\right), \quad 0 < \sigma \leq 1, \tag{10.24}$$

where C is the asymptotic error constant. With the additional hypothesis on the order, say $\rho \geq (1+\sqrt{5})/2$, in [19] the relations between ρ and the parameters $\overline{\lambda}_n$, $\widehat{\lambda}_n$, $\widetilde{\lambda}_n$ and $\breve{\lambda}_n$ are presented.

Using (10.24) and the definitions of $\hat{e}_n$, $\tilde{e}_n$ and $\breve{e}_n$, we obtain the following theoretical approximations of e_n. Namely,

$$e_n \approx C^{\frac{1}{1-\rho}} \left(\frac{\hat{e}_n}{\hat{e}_{n-1}} \right)^{\rho^2/(\rho-1)} \qquad n \geq 3\,, \tag{10.25a}$$

$$e_n \approx C^{\frac{\rho-1}{2\rho-1}} \left(\tilde{e}_n \right)^{\rho^2/\,(2\rho-1)} \qquad n \geq 3, \tag{10.25b}$$

$$e_n \approx C^{\frac{1}{1-\rho}} \left(\breve{e}_n \right)^{\rho/\,\rho-1} \qquad n \geq 2. \tag{10.25c}$$

From the preceding (10.25a), (10.25b) and (10.25c), we can obtain bounds of the error to predict the number of correct figures and establish a stopping criterion, all without knowledge of the root α.

C. Numerical Test The convergence of the new parameters has been tested in six iterative schemes with local convergence order equal to 2, 3, 4, $(1+\sqrt{5})/2$, $1+\sqrt{2}$ and $1+\sqrt{3}$ respectively, in a set of seven real functions shown in Table 10.1. The first three methods are one-point iterative methods without memory, known as the Newton method, the Chebyshev method [11] and the Schröder method [35]. The other three are iterative methods with memory, namely the Secant method and two of its variants (see [13]).

They are defined by

$$\phi_1(x_n) = x_n - u(x_n), \tag{10.26}$$

$$\phi_2(x_n) = \phi_1(x_n) - \frac{1}{2}\, L(x_n)\, u(x_n), \tag{10.27}$$

Table 10.1 Test functions, their roots and the initial points considered

$f(x)$	α	x_0	$\{x_{-1}\,,\, x_0\}$
$f_1(x) = x^3 - 3x^2 + x - 2$	2.893289196304497788906356	2.50	$\{2.25, 2.60\}$
$f_2(x) = x^3 + \cos x - 2$	1.172577964753970012673333	1.50	$\{1.50, 2.50\}$
$f_3(x) = 2\sin x + 1 - x$	2.380061273139339017212548	2.50	$\{1.00, 2.00\}$
$f_4(x) = (x+1)\,e^{x-1} - 1$	0.557145598997611416858672	1.00	$\{0.00, 0.75\}$
$f_5(x) = e^{x^2+7x-30} - 1$	3.0	2.94	$\{2.90, 3.10\}$
$f_6(x) = e^{-x} + \cos x.$	1.746139530408012417650703	1.50	$\{1.60, 1.90\}$
$f_7(x) = x - 3\ln x$	1.857183860207835336456981	2.00	$\{1.00, 2.00\}$

$$\phi_3(x_n) = \phi_2(x_n) - \left(\frac{1}{2}\,L(x_n)^2 \; - M(x_n)\right)\, u(x_n), \tag{10.28}$$

$$\phi_4(x_n) = x_n - \,[x_{n-1}, x_n]_f^{-1}\, f(x_n), \tag{10.29}$$

$$\phi_5(x_n) = \phi_4(x_n) - \,[x_n, \phi_4(x_n)]_f^{-1}\, f(\phi_4(x_n)), \tag{10.30}$$

$$\phi_6(x_n) = \phi_4(x_n) - \,[x_n, 2\phi_4(x_n) - x_n]_f^{-1}\, f(\phi_4(x_n)), \tag{10.31}$$

where

$$u(x) = \frac{f(x)}{f'(x)},\; L(x) = \frac{f''(x)}{f'(x)}\,u(x),\; M(x) = \frac{f'''(x)}{3!f'(x)}\,u(x)^2,\; [x,y]_f^{-1} = \frac{y-x}{f(y)-f(x)}.$$

The numerical results can be found in [20]. For each method from (10.26) to (10.31) and each function in Table 10.1, we have applied the four techniques with adaptive multi-precision arithmetic (see below) derived from relations (10.25a), (10.25b) and (10.25c) and the desired precision that for this study is 10^{-2200}. The number of necessary iterations to obtain the desired precision and the values of iterated points $x_1, \ldots, x_I$ are the same.

From the results of [20], we can conclude that CLOC gives the best approximation of the theoretical order of convergence of an iterative method. However, knowledge of the root is required. Conversely, as we can see in the definitions (10.23) of ACLOC, ECLOC and PCLOC, these parameters do not involve the expression of the root α. Actually, in real problems we want to approximate the root that is not known in advance. For practical purposes, we recommend ECLOC because it presents the best approximation of the local order (see [20]). Nevertheless, PCLOC is a good practical parameter in many cases because it requires fewer computations.

10.2.3 Multidimensional Case

A generalization to several variables of some parameters is carried out to approximate the local convergence order of an iterative method presented in the previous sections. In order to define the new parameters, we substitute the absolute value by the maximum norm, and all computations are done using the components of the vectors. Let $\{x_n\}_{n\in\mathbb{N}}$ be a convergence sequence of $\mathbf{R}^m$ towards $\alpha \in \mathbf{R}^m$, where $x_n = (x_n^{(1)}, x_n^{(2)}, \ldots, x_n^{(m)})^t$ and $\alpha = (\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(m)})^t$. We consider the vectorial sequence of the error $e_n = x_n - \alpha$ and the following vectorial sequences of parameters:

$$\hat{e}_n = x_n - x_{n-1}\,, \qquad \tilde{e}_n = \max_{1\le r\le m} \left| \frac{\left(\delta x_{n-1}^{(r)}\right)^2}{\delta^2 x_{n-2}^{(r)}} \right| \tag{10.32}$$

where $\delta x_n = x_{n+1} - x_n$. Notice that $\tilde{e}_n$ is the δ^2-Aitken procedure applied to the components of x_{n-1}, x_n and x_{n+1}, and all parameters are independent of knowledge of the root.

Definitions Let $\{\overline{\rho}_n\}_{n\geq 3}$, $\{\widehat{\rho}_n\}_{\geq 4}$, $\{\widetilde{\rho}_n\}_{\geq 5}$, $\{\breve{\rho}_n\}_{n\geq 3}$, $\{\overline{\lambda}_n\}_{n\geq 2}$, $\{\widehat{\lambda}_n\}_{\geq 3}$, $\{\widetilde{\lambda}_n\}_{\geq 4}$ y $\{\breve{\lambda}_n\}_{n\geq 2}$ be the following real sequences:

- Parameters COC, $\{\overline{\rho}_n\}_{n\geq 3}$ and CLOC, $\{\overline{\lambda}_n\}_{n\geq 2}$:

$$\overline{\rho}_n = \frac{\log\left(\|e_n\|/\|e_{n-1}\|\right)}{\log\left(\|e_{n-1}\|/\|e_{n-2}\|\right)}\,,\ n \geq 3\,,\quad \overline{\lambda}_n = \frac{\log\|e_n\|}{\log\|e_{n-1}\|}\,,\ n \geq 2\,. \tag{10.33a}$$

- Parameters ACOC, $\{\widehat{\rho}_n\}_{n\geq 4}$ and ACLOC $\{\widehat{\lambda}_n\}_{n\geq 3}$:

$$\widehat{\rho}_n = \frac{\log\left(\|\hat{e}_n\|/\|\hat{e}_{n-1}\|\right)}{\log\left(\|\hat{e}_{n-1}\|/\|\hat{e}_{n-2}\|\right)}\,,\ n \geq 4\,,\quad \widehat{\lambda}_n = \frac{\log\|\hat{e}_n\|}{\log\|\hat{e}_{n-1}\|}\,,\ n \geq 3\,. \tag{10.33b}$$

- Parameters ECOC $\{\widetilde{\rho}_n\}_{n\geq 5}$ and ECLOC $\{\widetilde{\lambda}_n\}_{n\geq 4}$:

$$\widetilde{\rho}_n = \frac{\log\left(\|\tilde{e}_n\|/\|\tilde{e}_{n-1}\|\right)}{\log\left(\|\tilde{e}_{n-1}\|/\|\tilde{e}_{n-2}\|\right)}\,,\ n \geq 5\,,\quad \widetilde{\lambda}_n = \frac{\log\|\tilde{e}_n\|}{\log\|\tilde{e}_{n-1}\|}\,,\ n \geq 4\,. \tag{10.33c}$$

- Parameters PCOC $\{\breve{\rho}_n\}_{n\geq 3}$ and PCLOC, $\{\breve{\lambda}_n\}_{n\geq 2}$:

$$\breve{\rho}_n = \frac{\|F(x_n)\|/\|F(x_{n-1})\|}{\|F(x_{n-1})\|/\|F(x_{n-2})\|},\ n \geq 3,\ \breve{\lambda}_n = \frac{\log\|F(x_n)\|}{\log\|F(x_{n-1})\|},\ n \geq 2. \tag{10.33d}$$

Approximations COC, ACOC and ECOC have been used in Grau et al. [15]. A complete study of these parameters has been carried out to compute the local convergence order for four iterative methods and seven systems of nonlinear equations.

10.3 The Vectorial Error Difference Equation

Here we present a generalization to several variables of a technique used to compute analytically the error equation of iterative methods without memory for one variable. We consider iterative methods to find a simple root of a system of non-linear equations

$$F(x) = 0\,,$$

where $F : D \subseteq \mathbf{R}^m \longrightarrow \mathbf{R}^m$ is sufficiently differentiable and D is an open convex domain in $\mathbf{R}^m$. Assume that the solution of $F(x) = 0$ is $\alpha \in D$, at which $F'(\alpha)$ is nonsingular.

The key idea is to use formal power series. The vectorial expression of the error equation obtained by carrying out this procedure, is

$$e_{n+1} = G\left(F'(\alpha),\, F''(\alpha), \ldots\right)\, e_n^{\rho} + \,O\left(e_n^{\rho+1}\right),$$

where ρ is a nonnegative integer. If the iterative scheme is with memory we obtain [see (10.13)]

$$e_{n+1} = H\left(F'(\alpha),\, F''(\alpha), \ldots\right)\, e_n^{a_1}\, e_{n-1}^{a_2} \cdots e_{n-j+1}^{a_j} + \,o\left(e_n^{a_1}\, e_{n-1}^{a_2} \cdots e_{n-j+1}^{a_j}\right),$$

where a_k are nonnegative integers for $1 \le k \le j$.

10.3.1 Notation

To obtain the vectorial equation of the error, we need some known results that, for ease of reference, are included in the following. Let $F : D \subseteq \mathbf{R}^m \longrightarrow \mathbf{R}^m$ be sufficiently differentiable (Fréchet-differentiable) in D, and therefore with continuous differentials. If we consider the kth derivative of F at $a \in \mathbf{R}^m$, we have the k-linear function

$$\begin{aligned} F^{(k)}(a) : \mathbf{R}^m \times \overset{k}{\cdots} \times \mathbf{R}^m &\longrightarrow \mathbf{R}^m \\ (h_1, \ldots, h_k) &\longmapsto F^{(k)}(a)\,(h_1, \ldots, h_k). \end{aligned}$$

That is, $F^{(k)}(a) \in \mathscr{L}\left(\mathbf{R}^m \times \overset{k}{\cdots} \times \mathbf{R}^m, \mathbf{R}^m\right) \equiv \mathscr{L}_k\,(\mathbf{R}^m, \mathbf{R}^m)$. It has the following properties:

P1. $F^{(k)}(a)\,(h_1, \ldots, h_{k-1}, \cdot) \in \mathscr{L}\,(\mathbf{R}^m,\, \mathbf{R}^m) \equiv \mathscr{L}\,(\mathbf{R}^m)$.

P2. $F^{(k)}(a)\,(h_{\sigma(1)}, \ldots, h_{\sigma(k)}) = F^{(k)}(a)\,(h_1, \ldots, h_k)$, where σ is any permutation of the set $\{1, 2, \ldots, k\}$.

Notice that from P1 and P2 we can use the following notation:

N1. $F^{(k)}(a)\,(h_1, \ldots, h_k) = F^{(k)}(a)\,h_1 \cdots h_k$. For $h_j = h$, $1 \le j \le k$, we write $F^{(k)}(a)\,h^k$.

N2. $F^{(k)}(a)\,h^{k-1}\,F^{(l)}(a)\,h^l = F^{(k)}(a)\,F^{(l)}(a)\,h^{k+l-1}$.

Hence, we can also express $F^{(k)}(a)\,(h_1,\ldots,h_k)$ as

$$\begin{aligned}F^{(k)}(a)\,(h_1,\ldots,h_{k-1})\,h_k &= F^{(k)}(a)\,(h_1,\ldots,h_{k-2})\,h_{k-1}\,h_k\\ &\;\;\vdots\\ &= F^{(k)}(a)\,h_1\cdots h_k\,.\end{aligned}$$

For any $q = a + h \in \mathbf{R}^m$ lying in a neighborhood of $a \in \mathbf{R}^m$, assuming that $[F'(a)]^{-1}$ exists, and taking into account this notation, we write Taylor's formulae in the following way:

$$\begin{aligned}F(a+h) &= F(a) + F'(a)\,h + \frac{1}{2!}F^{(2)}(a)\,h^2 + \cdots + \frac{1}{p!}F^{(p)}(a)\,h^p + O_{p+1}\,,\\ &= F(a) + F'(a)\left(h + \sum_{k=2}^{p} A_k(a)\,h^k + O_{p+1}\right),\end{aligned} \tag{10.34}$$

where $A_k(a) = \dfrac{1}{k!}\left[F'(a)\right]^{-1} F^{(k)}(a) \in \mathscr{L}_k\left(\mathbf{R}^m,\mathbf{R}^m\right),\;\; 2 \le k \le p,$

and $O_{p+1} = O(h^{p+1})\,.$

10.3.2 Symbolic Computation of the Inverse of a Function of Several Variables

We assume that $F : D \subseteq \mathbf{R}^m \longrightarrow \mathbf{R}^m$ has at least p-order derivatives with continuity on D for any $x \in \mathbf{R}^m$ lying in a neighborhood of a simple zero, $\alpha \in D$, of the system $F(x) = 0$. We can apply Taylor's formulae to $F(x)$. By setting $e = x - \alpha$, the local order, and assuming that $[F'(\alpha)]^{-1}$ exists, we have

$$F(x) = F(\alpha + e) = \Gamma\left(e + \sum_{k=2}^{p-1} A_k\,e^k\right) + O_p\,, \tag{10.35}$$

where

$$A_k = A_k(\alpha)\,,\; k \ge 2\,,\text{ with } \Gamma = F'(\alpha)\,,$$

$$e^k = (e,\overset{k}{\smile}\!\!\!\cdots,e) \in \mathbf{R}^m\times \overset{k}{\smile}\!\!\!\cdots \times\mathbf{R}^m \text{ and } O_p = O(e^p).$$

Moreover, from (10.35) noting the identity by I, the derivatives of $F(x)$ can be written as

$$F'(x) = \Gamma \left(I + \sum_{k=2}^{p-1} k\, A_k\, e^{k-1} \right) + \; O_p\,, \tag{10.36}$$

$$F''(x) = \Gamma \left(\sum_{k=2}^{p-2} k\,(k-1)\, A_k\, e^{k-2} \right) + \; O_{p-1}\,, \tag{10.37}$$

$$F'''(x) = \Gamma \left(\sum_{k=3}^{p-3} \frac{k!}{(k-3)!}\, A_k\, e^{k-3} \right) + \; O_{p-2}\,, \tag{10.38}$$

and so forth up to order p.

By developing a formal series expansion of e, the inverse of $F'(x)$ is

$$\left[F'(x)\right]^{-1} = \left(I + \sum_{j=1}^{4} K_j e^j + O_5 \right) \Gamma^{-1}, \tag{10.39}$$

where

$$\begin{aligned}
K_1 &= -2\, A_2,\\
K_2 &= 4\, A_2^2 - 3\, A_3,\\
K_3 &= -8\, A_2^3 + 6\, A_2\, A_3 + 6\, A_3\, A_2 - 4\, A_4,\\
K_4 &= 16\, A_2^4 - 12\, A_2^2 A_3 - 12\, A_2\, A_3\, A_2 - 12\, A_3\, A_2^2 + 8\, A_2\, A_4 + 8\, A_4\, A_2.
\end{aligned}$$

Example (Newton Method) We consider the Newton's method that we can write as

$$X = x - \; F'(x)^{-1}\, F(x). \tag{10.40}$$

The expression of the error $E = X - \alpha$ in terms of e is built up by subtracting α from both sides of (10.40) and taking into account (10.35) and (10.39). Namely,

$$\begin{aligned}
E = \; e - &\left(I + \sum_{j=1}^{3} K_j\, e^j + O_4 \right) \Gamma^{-1}\, \Gamma \left(e + \sum_{k=2}^{4} A_k\, e^k + \; O_5 \right)\\
&= A_2\, e^2 + 2(A_3 - A_2^2)\, e^3 + (3\, A_4 - 4\, A_2 A_3 - 3\, A_3 A_2 + 4\, A_2^3)\, e^4 + \; O_5.
\end{aligned} \tag{10.41}$$

The result (10.41) agrees with the classical asymptotic constant in the one-dimensional case and states that Newton's method has at least local order 2. Note that the terms A_2A_3 and A_3A_2 are noncommutative.

10.3.3 A Development of the Inverse of the First Order Divided Differences of a Function of Several Variables

We assume that $F : D \subseteq \mathbf{R}^m \longrightarrow \mathbf{R}^m$ has, at least, fifth-order derivatives with continuity on D. We consider the first divided difference operator of F in $\mathbf{R}^m$ as a mapping

$$\begin{aligned}[-,-;F]\, :\, D\times D &\longrightarrow \mathscr{L}(\mathbf{R}^m,\mathbf{R}^m)\\ (x+h,x) &\longrightarrow [x+h,x;\; F]\ ,\end{aligned}$$

which, for all $x, x+h \in D$, is defined by

$$[x+h\,,x;\; F]\,h = F(x+h) - F(x)\,, \tag{10.42}$$

where $\mathscr{L}(\mathbf{R}^m,\mathbf{R}^m)$ denotes the set of bounded linear functions (see [27, 32] and references therein). For F sufficiently differentiable in D, we can write:

$$F(x+h) - F(x) = \int_x^{x+h} F'(z)\,dz \ = \int_0^1 F'(x+th)\,dt. \tag{10.43}$$

By developing $F'(x+th)$ in Taylor's series at the point $x \in \mathbf{R}^m$ and integrating, we obtain

$$[x+h\,,x;\; F] \ = F'(x) + \frac{1}{2}F''(x)\,h + \cdots + \frac{1}{p!}F^{(p)}(x)\,h^{p-1} + \,O_p. \tag{10.44}$$

By developing $F(x)$ and its derivatives in Taylor's series at the point $x = \alpha + e$ lying in a neighborhood of a simple zero, $\alpha \in D$, of the system $F(x) = 0$, and assuming that $[F'(\alpha)]^{-1}$ exists, we obtain the expressions (10.35) and (10.38). Next, by replacing these expressions in (10.44), we obtain:

$$[x+h\,,x;\; F] = \Gamma\,\big(I + A_2(2e+h) + A_3(3\,e^2 + 3\,e\,h + h^2) + \ldots\big)\,, \tag{10.45}$$

or more precisely

$$[x+h\,,x;\;F]=\Gamma\left(I+\sum_{k=1}^{p-1}S_k(h,e)\;+O_p(\varepsilon,e)\right),\tag{10.46}$$

where $S_k(h,e)=A_{k+1}\sum_{j=1}^{k+1}\binom{k+1}{j}e^{k-j+1}\,h^{j-1}\,,\quad k\geq 1.$

We say that a function depending on ε and e is an $O_p(\varepsilon,e)$ if it is an $O(\varepsilon^{q_0}\,e^{q_1})$ with $q_0+q_1=p\,,q_i\geq 0\,,\;i=0,1.$

Setting $y=x+h$, $\varepsilon=y-\alpha$ and $h=\varepsilon-e$ in (10.45) and (10.46) we obtain

$$[\,y\,,x;\;F]=\Gamma\left(I+A_2(\varepsilon+e)+A_3(\varepsilon^2+\varepsilon\,e+e^2)+\ldots\right),\tag{10.47}$$

or more precisely

$$[\,y\,,x;\;F]=\Gamma\left(I+\sum_{k=1}^{p-1}T_k(\varepsilon,e)\;+O_p(\varepsilon,e)\right),\tag{10.48}$$

where $T_k(\varepsilon,e)=A_{k+1}\sum_{j=0}^{k}\varepsilon^{k-j}\,e^j.$

If we expand in formal power series of e and ϵ, the inverse of the divided difference given in (10.47) or in (10.48) is:

$$[\,y\,,x;\;F]^{-1}=\Big(I-A_2(\epsilon+e)-A_3(\varepsilon^2+\varepsilon\,e+e^2)+\big(A_2(\epsilon+e)\big)^2+O_3(\varepsilon,e)\Big)\,\Gamma^{-1}.\tag{10.49}$$

Notice that Eq. (10.49) is written explicitly until the 2nd-degree in ε and e, while in each specific circumstance it will be adapted and reduced to the necessary terms, with an effective contribution to the computation of the local order of convergence.

These developments of the divided difference operator (10.49) were first used in our study Grau et al. [18].

Example (Secant Method) The generic case, (10.47), (10.48) or (10.49) can be adapted to different cases. For example, the well-known iterative method called the Secant method [27, 32] is defined by the algorithm:

$$x_{n+1}=x_n-\;[x_{n-1}\,,x_n\,;\;F]^{-1}\;F(x_n)\,,\quad x_0\,,x_1\in D.\tag{10.50}$$

If $y=x_{n-1}$ and $x=x_n$ in (10.47) then we obtain an expression of the operator $[x_{n-1}\,,x_n\,;\;F]$ in terms of $e_{n-1}=x_{n-1}-\alpha$ and $e_n=x_n-\alpha$. If we expand in formal power series of e_{n-1} and e_n the inverse of the divided difference operator in the

Secant method we obtain

$$[x_{n-1}, x_n; F]^{-1} = \big(I - A_2\,(e_{n-1} + e_n) + (A_2^2 - A_3)\,e_{n-1}^2 + o(e_{n-1}^2)\big)\,\Gamma^{-1}, \tag{10.51}$$

where $A_2^2\,e_{n-1}^2 = (A_2\,e_{n-1})^2$. The expression of the error $e_{n+1} = x_{n+1} - \alpha$ in terms of e_n and e_{n-1} for the Secant method is built up by subtracting α from both sides of (10.50). Taking into account (10.35) and (10.51), we have

$$\begin{aligned} e_{n+1} &= e_n - \big(I - A_2(e_{n-1} + e_n) + (A_2^2 - A_3)e_{n-1}^2 + o(e_{n-1}^2)\big)\cdot \\ &\qquad \cdot \big(e_n + A_2 e_n^2 + O(e_n^3)\big) \\ &= A_2\,e_{n-1}\,e_n + (A_3 - A_2^2)\,e_{n-1}^2\,e_n + o(e_{n-1}^2\,e_n), \end{aligned} \tag{10.52}$$

where the indicial polynomial [see (10.17)] of the error difference equation (10.52) is $t^2 - t - 1 = 0$, with only one positive real root, which is the R-order of convergence of the Secant method, $\phi = (1 + \sqrt{5})/2$. The second term of the right side of (10.52) would give order 2, since its associated polynomial equation is $t^2 - t - 2 = 0$. This result agrees with the classical asymptotic constant in the one-dimensional case and states that the Secant method has at least local order ϕ.

A more complete expression of the error expression for the Secant method can be found in our studies Grau et al. [13] and Ezquerro et al. [7]. A generalization of Ostrowski method for several variables can be found in [17].

10.4 Efficiency Indices

We are interested in comparing iterative processes to approximate a solution α of a system of nonlinear equations. In the scalar case, the parameters of the efficiency index (*EI*) and computational efficiency (*CE*) are possible indicators of the efficiency of the scheme. Then, we consider the computational efficiency index (*CEI*) as a generalization to the multi-dimensional case. We show the power of this parameter by applying it to some numerical examples.

10.4.1 Efficiency Index and Computational Efficiency

To compare different iterative methods for solving scalar nonlinear equations, the efficiency index suggested by Ostrowski [28] is widely used,

$$EI = \rho^{1/a}, \tag{10.53}$$

where ρ is the local order of convergence of the method and a represents the number of evaluations of functions required to carry out the method per iteration.

Another classical measure of efficiency for iterative methods applied to scalar nonlinear equations is the computational efficiency proposed by Traub [41],

$$CE = \rho^{1/\omega}, \tag{10.54}$$

where ω is the number of operations, expressed in product units, that are needed to compute each iteration without considering the evaluations of the functions. In general, if we are interested in knowing the efficiency of a scalar scheme, the most frequently used parameter is *EI*, instead of any combination of this parameter with *CE*.

The efficiency index for Newton's method is $2^{1/2} \approx 1.414$ because an iterative step of Newton requires the computation of $f(x)$ and $f'(x)$ then $a = 2$, and the local order is $\rho = 2$. Note that the parameter *EI* is independent of the expression of f and its derivative, while the parameter *CE* does not consider the evaluation of the computational cost of the functions of the algorithm.

More precisely, note that an iteration step requires two actions: first the calculation of new functions values; and then the combination of data to calculate the next iterate. The evaluation of functions requires the invocation of subroutines, whereas the calculation of the next iterate requires only a few arithmetic operations. In general, these few arithmetic operations are not considered in the scalar case.

10.4.2 Computational Efficiency Index

The traditional way to present the computational efficiency index of iterative methods (see [28, 41]) is adapted for systems of nonlinear equations. When we deal with a system of nonlinear equations, the total operational cost is the sum of the evaluations of functions (the function and the derivatives involved) and the operational cost of doing a step of the iterative method.

In m-dimensional case, the choice of the most suitable iterative method, $x_{n+1} = \Phi(x_n)$, depends mainly on its efficiency which also depends on the convergence order and the computational cost. The number of operations per iteration increases the computational cost in such a way that some algorithms will not be used because they are not efficient. In general, we have a scheme such as the following

$$\Phi(x_n) = x_n - \Theta_n^{-1} F(x_n),$$

where instead of computing the inverse of the operator Θ_n, we solve the following linear system

$$\begin{aligned} \Theta_n y_n &= -F(x_n), \\ x_{n+1} &= x_n + y_n. \end{aligned}$$

Therefore, we choose the LU-decomposition plus the resolution of two linear triangular systems in the computation of the inverse operator that appears. In other words, in the multi-dimensional case we have to perform a great number of operations, while in the scalar case the number of operations is reduced to a very few products.

Let ℓ be the conversion factor of quotients into products (the time needed to perform a quotient, in time of product units). Recall that the number of products and quotients that we need to solve an m-dimensional linear system, using the LU-decomposition is

$$\omega_1 \;=\; \frac{m}{6}\,(2\,m^2 - 3\,m + 1) \;+\; \ell\,\frac{m}{2}\,(m-1),$$

and to solve the two triangular linear systems with ones in the main diagonal of matrix L we have $\omega_2 \;=\; m\,(m-1) \;+\; \ell\,m$ products. Finally, the total number of products is

$$\frac{m}{6}\,\left(2\,m^2 + 3\,(1+\ell)\,m + 3\,\ell - 5\right).$$

Definition 4 The computational efficiency index (CEI) and the computational cost per iteration ($\mathscr{C}$) are defined by (see [13, 16, 18, 32])

$$CEI(\mu_0,\mu_1,m,\ell) \;=\; \rho^{\frac{1}{\mathscr{C}(\mu_0,\mu_1,m,\ell)}}, \tag{10.55}$$

where $\mathscr{C}(\mu_0,\mu_1,m,\ell)$ is the computational cost given by

$$\mathscr{C}(\mu_0,\mu_1,m) = a_0(m)\,\mu_0 + a_1(m)\,\mu_1 + \omega(m,\ell), \tag{10.56}$$

$a_0(m)$ represents the number of evaluations of the scalar functions $(F_1,\ldots,F_m)$ used in one step of the iterative method.

$a_1(m)$ is the number of evaluations of scalar functions of F', say $\dfrac{\partial F_i}{\partial x_j}$, $1 \le i,j \le m$.

$\omega(m,\ell)$ represents the number of products needed per iteration.

The constants μ_0 and μ_1 are the ratios between products and evaluations required to express the value of $\mathscr{C}(\mu_0,\mu_1,m)$ in terms of products, and ℓ is the cost of one quotient in products.

Note that:

$$CEI(\mu_0,\mu_1,m,\ell) \;>\; 1, \qquad \lim_{m\to\infty} CEI(\mu_0,\mu_1,m,\ell) = 1.$$

Notice that for $\mu_0 = \mu_1 = 1$ and $\omega(m) = 0$, (10.55) is reduced to (10.53), that is the classic efficiency index of an iterative method, say $EI = \rho^{1/a}$ in the scalar case. Also observe that, if $a_0 = a_1 = 0$, (10.55) is written in the scalar case as (10.54); namely, $CE = \rho^{1/\omega}$.

According to (10.56), an estimation of the factors μ_0, μ_1 is required. To do this, we express the cost of the evaluation of the elementary functions in terms of products, which depend on the machine, the software and the arithmetic used. In [10, 38], we find comparison studies between a multi-precision library (MPFR) and other computing libraries. In Tables 10.2 and 10.3 our own estimation of the cost of the elementary functions is shown in product units, where the running time of one product is measured in milliseconds.

The values presented in Table 10.2 have been rounded to 5 unities because of the huge variability in the different repetitions that were carried out. In contrast, the averages are shown in Table 10.3, since the variability was very low. In addition, the compilator of C++ that was used ensures that the function `clock()` gives exactly the CPU time invested by the program. Table 10.3 shows that some relative values for the product are lower in multiple precision than in double precision, although the absolute time spent on a product is much higher in multiple precision.

This measure of computing efficiency is clearly more satisfactory than considering only the number of iterations or only the number of evaluated functions, which are used widely by others authors. Any change of software or hardware requires us to recompute the elapsed time of elemental functions, quotients and products.

In this section we compare some free derivative iterative methods that use the divided difference operator (see [13]). Firstly, we recall the classical Secant method (10.50) and we study a few two-step algorithms with memory.

Table 10.2 Computational cost of elementary functions computed with Matlab 2009b and Maple 13 on an Intel®Core(TM)2 Duo CPU P8800 (32-bit machine) processor with Microsoft Windows 7 Professional, where $x = \sqrt{3} - 1$ and $y = \sqrt{5}$

Software	$x * y$	x/y	$\sqrt{x}$	$\exp(x)$	$\ln(x)$	$\sin(x)$	$\cos(x)$
Matlab 2009b	4.5E−7 ms	10	55	80	145	35	50
Maple13 16 digits	1.2E−3 ms	1	10	25	45	25	20
Maple13 1024 digits	4.0E−2 ms	1	5	45	10	90	90
Maple13 4096 digits	3.5E−1 ms	1	5	50	10	105	105

Table 10.3 Computational cost of elementary functions computed with a program written in C++, compiled by `gcc(4.3.3)` for i486-linux-gnu with `libgmp (v.4.2.4)` and `libmpfr (v.2.4.0)` libraries on an Intel®Xeon E5420, 2.5 GHz, 6 MB cache processor where $x = \sqrt{3} - 1$ and $y = \sqrt{5}$

Arithmetics	$x * y$	x/y	$\sqrt{x}$	$\exp(x)$	$\ln(x)$	$\sin(x)$	$\cos(x)$
C++ `double`	2.3E−7 ms	29	29	299	180	181	192
C++ MPFR 1024 digits	1.16E−2 ms	2.4	1.7	62	57	69	65
C++ MPFR 4096 digits	1.04E−1 ms	2.5	1.7	88	66	116	113

10.4.3 Examples of Iterative Methods

Secant Method We call Φ_0 to the well-known iterative secant method (10.50). That is, by setting $x_{-1}, x_0 \in D$

$$x_{n+1} = \Phi_0(x_{n-1}, x_n) = x_n - [x_{n-1}, x_n; F]^{-1} \, F(x_n), \tag{10.57}$$

with the local order of convergence $\phi = (1 + \sqrt{5})/2 = 1.618\ldots$

Frozen Divided Difference Method We consider two steps of the Secant method and by setting x_{-1}, x_0 given in D,

$$\begin{cases} y_n = \Phi_0(x_{n-1}, x_n), \\ x_{n+1} = \Phi_1(x_{n-1}, x_n) = y_n - [x_{n-1}, x_n; F]^{-1} \, F(y_n), \quad n \geq 0. \end{cases} \tag{10.58}$$

In this case, the local order is at least 2. Therefore, this method is a quadratic method where the divided difference operator is only computed once, which is why the reason why it is called the frozen divided difference method (for more details see [18]).

The next two iterative methods are pseudo-compositions of two known schemes, which are two-step algorithms.

First Superquadratic Method We take the Secant method twice. That is, by putting x_{-1}, x_0 given in D,

$$\begin{cases} y_n = \Phi_0(x_{n-1}, x_n), \\ x_{n+1} = \Phi_2(x_{n-1}, x_n) = y_n - [x_n, y_n; F]^{-1} \, F(y_n), \quad n \geq 0. \end{cases} \tag{10.59}$$

The order of the two-step iterative method with memory defined in (10.59) is $1 + \sqrt{2} = 2.414\ldots$

Second Superquadratic Method We define the pseudo-composition of the Secant method with the Kurchatov method [9, 26]. The result is the following scheme: Given x_{-1}, x_0 in D,

$$\begin{cases} y_n = \Phi_0(x_{n-1}, x_n), \\ x_{n+1} = \Phi_3(x_{n-1}, x_n) = y_n - [x_n, 2y_n - x_n; F]^{-1} \, F(y_n), \quad n \geq 0. \end{cases} \tag{10.60}$$

This two-step scheme with memory has a local order of convergence equal to $1 + \sqrt{3} = 1.732\ldots$

Finally, we observe that we have moved from a superlinear method such as the Secant method with local order equal to $(1 + \sqrt{5})/2$ to a superquadratic method with local order equal to $1 + \sqrt{3}$.

10.4.4 Comparisons Between These Methods

We study the efficiency index of the four iterative methods Φ_j, $0 \le j \le 3$, given by (10.57), (10.58), (10.59) and (10.60) respectively. The computational efficiency index (CEI_j) of each iterative method and the computational cost per iteration ($\mathscr{C}_j$) are defined in (10.55) as

$$CEI_j(\mu, m, \ell) = \rho^{\dfrac{1}{\mathscr{C}_j(\mu, m, \ell)}},$$

where

$$\mathscr{C}_j(\mu, m, \ell) = a_j(m)\mu + \omega_j(m, \ell).$$

Note that we denote μ_0 by μ in these examples. For each method Φ_j for $j = 0, 1, 2, 3$, Table 10.4 shows: the local order of convergence ρ_j; the number of evaluations of F (NEF); the number of computations of the divided differences (DD); the value of $a_j(m)$; and $\omega_j(m, \ell)$. In order to obtain these results, we consider the following computational divided difference operator (10.61). To compute the $[x_{n-1}, x_n; F]$ operator, we need m^2 divisions and $m(m-1)$ scalar evaluations. Note that, for the $[x_n, y_n; F]$ or $[x_n, 2y_n - x_n; F]$ operators we need m^2 scalar evaluations.

$$\begin{aligned}[y, x; F]^{(1)}_{ij} = \big(&F_i(y_1, \ldots, y_{j-1}, y_j, x_{j+1}, \ldots, x_m) - \\ &F_i(y_1, \ldots, y_{j-1}, x_j, x_{j+1}, \ldots, x_m)\big) / (y_j - x_j), \quad 1 \le i, j \le m.\end{aligned} \tag{10.61}$$

Summarizing the results of Table 10.4 we have

$$\begin{aligned}
\mathscr{C}_0(\mu, m, \ell) &= \frac{m}{6}(2m^2 + 6m\mu + 3m + 9\ell m + 3\ell - 5), & \rho_0 &= \tfrac{1+\sqrt{5}}{2}\,;\\
\mathscr{C}_1(\mu, m, \ell) &= \frac{m}{6}(2m^2 + 6m\mu + 9m + 9\ell m + 6\mu + 9\ell - 11), & \rho_1 &= 2;\\
\mathscr{C}_2(\mu, m, \ell) &= \frac{m}{3}(2m^2 + 6m\mu + 3m + 9\ell m + 3\ell - 5), & \rho_2 &= 1 + \sqrt{2}\,;\\
\mathscr{C}_3(\mu, m, \ell) &= \frac{m}{3}(2m^2 + 6m\mu + 3m + 9\ell m + 3\mu + 3\ell - 2), & \rho_3 &= 1 + \sqrt{3}\,.
\end{aligned}$$

In order to compare the corresponding CEIs we use the following quotient

$$R_{ij} = \frac{\log CEI_i}{\log CEI_j} = \frac{\log \rho_i}{\log \rho_j}\, \frac{\mathscr{C}_j}{\mathscr{C}_i},$$

and we have the following theorem [13]. In Table 10.5, we present the different situations of this theorem.

Table 10.4 Local convergence order and computational cost of methods Φ_j for $0 \leq j \leq 3$

	ρ_j	NEF	DD	$a_j(m)$	$\omega_j(m,\ell)$
Φ_0	$(1+\sqrt{5})/2$	1	1	$m(m-1)+m=m^2$	$m(2m^2+3m+9\ell m+3\ell-5)/6$
Φ_1	2	2	1	$a_0(m)+m=m^2+m$	$p_0(m)+m(m-1)+m\ell=$ $m(2m^2+9m+9\ell m+9\ell-11)/6$
Φ_2	$1+\sqrt{2}$	1	2	$a_0(m)+m^2=2m^2$	$2p_0(m)=$ $m(2m^2+3m+9\ell m+3\ell-5)/3$
Φ_3	$1+\sqrt{3}$	2	2	$a_1(m)+m=2m^2+m$	$2p_0(m)+1=$ $m(2m^2+3m+9\ell m+3\ell-2)/3$

Table 10.5 The four situations of Theorem 1

$m=2$	$m=3$	$4\leq m\leq 11$	$m\geq 12$
$CEI_0>CEI_2$			
$CEI_1>CEI_2$	$CEI_1>CEI_0>CEI_2$	$CEI_1>CEI_0>CEI_2$	
$CEI_1>CEI_3$	$CEI_1>CEI_3$	$CEI_1>CEI_3>CEI_2$	$CEI_1>CEI_0>CEI_3>CEI_2$

Theorem 1 *For all $\ell \geq 1$ we have:*

1. *$CEI_1 > CEI_2$ and CEI_3, for all $m \geq 2$.*
2. *$CEI_0 > CEI_2$ for all $m \geq 2$.*
3. *$CEI_1 > CEI_0$, for all $m \geq 3$.*
4. *$CEI_3 > CEI_2$ for all $m \geq 4$.*
5. *$CEI_3 > CEI_0$, for all $m \geq 12$,*

10.4.5 Numerical Results

The numerical computations listed in Tables 10.7, 10.8, 10.9, 10.10 10.11, 10.12 and 10.13 were performed on an MPFR library of C++ multiprecision arithmetics [39] with 4096 digits of mantissa. All programs were compiled by `gcc(4.3.3)` for i486-linux-gnu with `libgmp (v.4. 2.4)` and `libmpfr (v.2.4.0)` libraries on an Intel®Xeon E5420, 2.5 GHz and 6 MB cache processor. For this hardware and software, the computational cost of the quotient with respect to the product is $\ell = 2.5$ (see Table 10.3). Within each example the starting point is the same for all methods tested. The classical stopping criterion $||e_I|| = ||x_I - \alpha|| > 0.5 \cdot 10^{-\varepsilon}$ and $||e_{I+1}|| \leq 0.5 \cdot 10^{-\varepsilon}$, with $\varepsilon = 4096$, is replaced by

$$E_I = \frac{||\hat{e}_I||}{||\hat{e}_{I-1}||} < 0.5 \cdot 10^{-\eta},$$

where $\hat{e}_I = x_I - x_{I-1}$ and $\eta = \frac{\rho-1}{\rho^2}\,\varepsilon$ [see (10.25a)]. Notice that this criterium is independent of the knowledge of the root. Furthermore, in all computations we have substituted the computational order of convergence (COC) [43] by the approximation ACOC, $\hat{\rho}_I$ (10.33b).

Examples We present the system defined by

$$F_i(x_1,\dots,x_m) = \sum_{\substack{j=1\\ j\neq i}}^{m} x_j - \exp(-x_i) = 0, \quad 1 \le i \le m, \tag{10.62}$$

where $m = 3, 5, 13$ and $\mu = 87.8$ for arithmetics of 4096 digits, since in (10.62) μ is independent from m. The three values of m correspond to three situations of the Theorem 1 (see Table 10.6). Tables 10.7, 10.8 and 10.9 show the results obtained for the iterative methods Φ_0, Φ_1, Φ_2 and Φ_3 respectively.

Table 10.6 The three cases of Theorem 1 for $\mu = 87.8$ and $\ell = 2.5$

Case 1. $m = 3$	Case 2. $m = 5$	Case 3. $m = 13$
$CEI_1 > CEI_0 > CEI_2 > CEI_4$	$CEI_1 > CEI_0 > CEI_4 > CEI_2$	$CEI_1 > CEI_4 > CEI_0 > CEI_2$

Table 10.7 Numerical results for case 1, where $m = 3$ and $t_p = 0.1039$

Φ_i	I	T	D_I	CEI	TF	$\lvert\Delta\hat{\rho}_I\rvert$
Φ_0	17	1540	3130	1.000573924	4013.15	5.18E−6
Φ_1	10	1340	3310	1.000621515	3705.95	7.79E−3
Φ_2	9	1620	3960	1.000525578	4382.20	1.76E−5
Φ_3	7	1480	2140	1.000517189	4453.27	2.05E−3

Table 10.8 Numerical results for case 2, where $m = 5$ and $t_p = 0.1039$

Φ_i	I	T	D_I	CEI	TF	$\lvert\Delta\hat{\rho}_I\rvert$
Φ_0	17	4290	3470	1.000205229	11220.74	2.65E−6
Φ_1	10	3040	2180	1.000246133	9356.20	8.25E−3
Φ_2	8	4030	1880	1.000187944	12252.59	1.12E−5
Φ_3	7	3850	2140	1.000195783	11762.06	1.28E−3

Table 10.9 Numerical results for case 3, where $m = 13$ and $t_p = 0.1039$

Φ_i	I	T	D_I	CEI	TF	$\lvert\Delta\hat{\rho}_I\rvert$
Φ_0	17	29070	2630	1.000029533	77967.67	3.99E−6
Φ_1	11	20460	2950	1.000039330	58546.41	7.58E−3
Φ_2	9	30950	3380	1.000027046	85137.03	1.45E−5
Φ_3	7	24810	1650	1.000029786	77305.42	1.58E−3

For each case, we present one table where we read the method Φ_i, the number of iterations needed I to reach the maximum precision requested, the computational elapsed time T in milliseconds of the C++ execution for these iterations, the correct decimals reached in D_I approximately, the computational efficiency index CEI, the time factor TF $= 1/\log CEI$, an error's higher bound of the ACOC computation $\Delta\hat{\rho}_I$ where $\rho = \hat{\rho}_I \pm \Delta\hat{\rho}_I$.

Case 1 We begin with the system (10.62) for $m = 3$ where $CEI_1 > CEI_0 > CEI_2 = CEI_3 > CEI_4$. The root $\alpha = (\alpha_i),\ 1 \leq i \leq m$, and the two initial points $x_{-1},\ x_0$ are

$$\alpha_1 = -0.8320250398, \quad \alpha_{2,3} = 1.148983754,$$
$$x_{-1} = (-0.8, 1.1, 1.1)^t \quad x_0 = (-0.9, 1.2, 1.2)^t.$$

The numerical results of this case are shown in Table 10.7.

Case 2 The second case is the system (10.62) for $m = 5$ where $CEI_1 > CEI_0 > CEI_4 > CEI_2 = CEI_3$. The numerical results of this case are shown in Table 10.8. The root α and the two initial points $x_{-1},\ x_0$ are

$$\alpha_{1,2,5} = -2.153967996, \quad \alpha_{3,4} = 6.463463374,$$
$$x_{-1} = (-2.1, -2.1, 6.4, 6.4, -2.1)^t \quad x_0 = (-2.2, -2.2, 6.5, 6.5, -2.2)^t.$$

Case 3 Finally, the third case is the system (10.62) for $m = 13$ where $CEI_1 > CEI_4 > CEI_0 > CEI_2 = CEI_3$. The numerical results of this case are in Table 10.9. The root α and the two initial points $x_{-1},\ x_0$ are

$$\alpha_{1,2,3,5,7,10} = 1.371341671, \quad \alpha_{4,6,8,9,11,12,13} = -.9432774419,$$
$$x_{-1} = (1.3, 1.3, 1.3, -0.9, 1.3, -0.9, 1.3, -0.9, -0.9, 1.3, -0.9, -0.9, -0.9)^t,$$
$$x_0 = (1.4, 1.4, 1.4, -1.0, 1.4, -1.0, 1.4, -1.0, -1.0, 1.4, -1.0, -1.0, -1.0)^t.$$

Remark 1 In case 1, we can arrange methods Φ_2 and Φ_3 according to the elapsed time T or the time factor TF. The results are different because the final precisions D_I obtained in each method are not comparable. In Sect. 10.5 we explain a better way to compare the elapsed time that is more consistent with the theoretical results of the computational efficiency index CEI.

Remark 2 The first numerical definition of divided difference (10.61) has a counterexample in the following 2×2 system of nonlinear equations

$$F(x_1, x_2) = \begin{cases} x_1^2 + x_2^2 - 9 = 0, \\ x_1 x_2 - 1 = 0. \end{cases} \tag{10.63}$$

Scheme Φ_3 gives a PCLOC $\breve{\rho} = 1 + \sqrt{2}$, instead of the theoretical value $\rho = 1 + \sqrt{3}$. Furthermore, a comparison of the expression (10.43), taking into account the definition of the divided differences operator (10.61), gives the following result

$$\int_0^1 F'(x+th)\,dt = \begin{pmatrix} 2x_1+h_1 & 2x_2+h_2 \\ x_2+h_2/2 & x_1+h_1/2 \end{pmatrix} \neq [x+h,x;F]^{(1)} = \begin{pmatrix} 2x_1+h_1 & 2x_2+h_2 \\ x_2 & x_1+h_1 \end{pmatrix},$$

where $x=(x_1,x_2)^t$, $h=(h_1,h_2)^t$ and $t\in\mathbf{R}$. Due to Potra [30, 32] we have the following necessary and sufficient condition to characterize the divided difference operator by means of a Riemann integral.

Theorem 2 *If F satisfies the following Lipschitz condition $\|[x,y;F]-[u,v;F]\|\leq H\,(\|x-u\|+\|y-v\|)$, then equality (10.61) holds for every pair of distinct points $(x+h,x)\in D\times D$ if, and only if, for all $(u,v)\in D\times D$ with $u\neq v$ and $2v-u\in D$ the following relation is satisfied:*

$$[u,v;F] \;=\; 2\,[u,2v-u;F]-[v,2v-u;F]. \tag{10.64}$$

We can check that the function considered in (10.75) does not hold (10.64). We need a new definition of divided differences instead of the definition given in (10.61) to obtain the local order required when we apply algorithms Φ_k, $k=0,1,2,3$, in this case. We use the following method to compute the divided difference operator

$$[y,x;F]^{(2)}_{ij} = \frac{1}{2}\left([y,x;F]^{(1)}_{ij}+[x,y;F]^{(1)}_{ij}\right),\quad 1\leq i,j\leq m. \tag{10.65}$$

Notice that the operator defined in (10.65) is symmetric: $[y,x;F]=[x,y;F]$. If we use definition (10.65), we have to double the number of evaluations of the scalar functions in the computation of $[y,x;F]$, and by rewriting (10.65) we have m^2 quotients.

If we use (10.65) then method Φ_3 applied to system (10.75), the computational order of convergence is equal to $1+\sqrt{3}$. Another example with the same behavior as (10.75) is

$$F_i(x_1,\ldots,x_3) = x_i-\cos\left(\sum_{\substack{j=1\\ j\neq i}}^{3} x_j\; - x_i\right),\quad 1\leq i\leq 3. \tag{10.66}$$

In Table 10.10, we show the numerical results of method Φ_3 applied to the systems of nonlinear equations (10.75) and (10.66). We denote by $\Phi_3^{(j)}$, $j=1,2$, method Φ_3 using the numerical definition of the divided difference operator $[x+h,x;F]^{(j)}, j=1,2$ respectively. By setting $\mathrm{TF}_3^{(j)}=1/\log CEI_3^{(j)}$ as the time factors of methods $\Phi_3^{(j)}$ and by comparing the two time factors of system (10.75), we can conclude (see Table 10.10) that method $\Phi_3^{(2)}$ is more efficient than $\Phi_3^{(1)}$. This behavior is reversed in example (10.66).

Table 10.10 Numerical results for the scheme Φ_3 applied to systems (10.75) and (10.66)

		System (10.75)				System (10.66)			
	$\rho_3^{(j)}$	$\mathrm{TF}_3^{(j)}$	I	D_I	$\lvert\Delta\hat{\rho}_I\rvert$	$\mathrm{TF}_3^{(j)}$	I	D_I	$\lvert\Delta\hat{\rho}_I\rvert$
$\Phi_3^{(1)}$	$1+\sqrt{2}$	65.8	7	2918	1.2E−4	2205.4	9	3386	2.3E−4
$\Phi_3^{(2)}$	$1+\sqrt{3}$	63.7	6	2853	4.3E−3	2982.3	8	4199	1.7E−4

10.5 Theoretical Numerical Considerations

Theoretical and experimental studies of numerical applications often move away from one another. From studies by Ostrowski [28], Traub [41] and Ralston [33], we have introduced new concepts that allow us to estimate the execution time. We revisit the time factor [18] and we present a relation between these measures and the number of iterations, to achieve a given precision in the root computation. In other words, in the classical comparison between two iterative methods, the following ratio of efficiency logarithms was introduced

$$\frac{\Theta_1}{\Theta_2}=\frac{\log CEI_2}{\log CEI_1}=\frac{\mathscr{C}_1/\log\rho_1}{\mathscr{C}_2/\log\rho_2}, \tag{10.67}$$

where Θ is the total cost of products to obtain the required precision when we apply a method. That is, if I is the total number of iterations, then

$$\Theta = I\cdot\mathscr{C}. \tag{10.68}$$

In this section, we also introduce a new factor that provides us with an explicit expression of the total time $\tilde{\Theta}$.

10.5.1 Theoretical Estimations

As an iterative method has local order of convergence ρ and local error $e_n = x_n-\alpha$, we define $D_n = -\log_{10}||e_n||$. That is, D_n is approximately the number of correct decimal places in the nth iteration. From the definition of local order, we have $||e_{I+1}|| \approx C||e_I||^{\rho}$ and $D_{I+1} \approx -\log_{10}C + \rho D_I$. The solution of this difference equation is

$$D_I \approx D_0\rho^I + \log_{10}M, \quad \text{where} \quad M = C^{1/(\rho-1)},$$

and we obtain $D_I \approx D_0\rho^I$. If we apply logarithms to both sides of the preceding equation and take into account (10.68) we get

$$I=(\log q)/(\log\rho) \quad \text{and} \quad \Theta=\log q\frac{\mathscr{C}}{\log\rho}=\frac{\log q}{\log CEI}, \tag{10.69}$$

where $q = D_I/D_0$. From (10.68), if we take $\tilde{\Theta}(q) = \Theta\, t_p$, where t_p is the time required to do one product, then $\tilde{\Theta}(q)$ is the total time. Taking into account (10.55) and (10.69), the total time is $\tilde{\Theta}(q) \approx \log q \dfrac{t_p}{\log CEI}$. If we consider the time in product units, then $1/\log CEI$ will be called the time factor (TF). Notice that the term $\log q$ introduced in (10.69) is simplified in the quotient of Eq. (10.67). In [33], (10.67) is obtained from

$$\frac{\Theta_1}{\Theta_2} = \frac{I_1}{I_2}\frac{\mathscr{C}_1}{\mathscr{C}_2}. \tag{10.70}$$

Then, considering $\Theta = \mathscr{C}/\ln\rho$, the efficiency index is

$$EI = \frac{1}{\Theta} = \frac{\ln\rho}{\mathscr{C}} = \ln\rho^{1/\mathscr{C}}.$$

Not only have we taken CEI as defined in (10.55), but we have also expressed the factor that is simplified in (10.70) as

$$I_k = \frac{\log q}{\log\rho_k}\quad k = 1, 2,$$

as we inferred and deduced in (10.69). We are now in a position to state the following theorem.

Theorem 3 *In order to obtain D_I correct decimals from D_0 correct decimals using an iterative method, we can estimate*

- *the number of iterations* $I \approx \dfrac{\log q}{\log\rho}$,
- *the necessary time* $\tilde{\Theta}(q) \approx \log q \dfrac{t_p}{\log CEI}$,

where $q = D_I/D_0$ and t_p is the time needed for one product and $\mathscr{C}$ is the computational cost per iteration.

When we are faced with a numerical problem we rewrite the estimation of time as in the following equation of the straight line in variables $(\log D_I, \tilde{\Theta})$:

$$\tilde{\Theta}(D_I) = \frac{t_p}{\log CEI}(\log D_I - \log D_0) = \kappa \log\frac{D_I}{D_0} = \kappa\log q,$$

where $\kappa = t_p/\log(CEI)$. That is, κ is a coefficient that measures the time of execution in function of the approximate number of correct decimals. In order to study and analyze an iterative method, we can consider the straight line $\tilde{\Theta}(D_I)$ with slope κ in a semi-logarithmic graph. If we approximate the $(\log D_j, \Theta(D_j))$ pairs in a least-squares sense by a polynomial of degree one, we can compute an experimental slope $\tilde{\kappa}$ that is used in Figs. 10.1, 10.2 and 10.3, and Tables 10.11, 10.12 and 10.13.

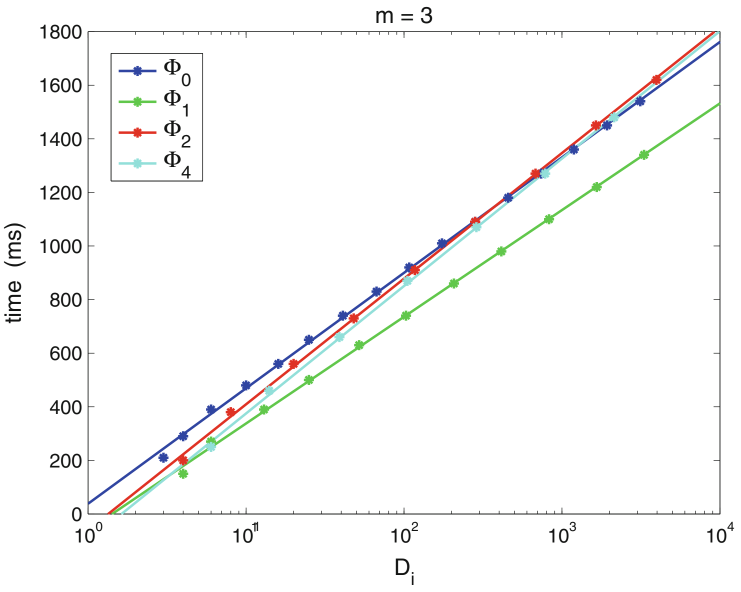

Fig. 10.1 Time t versus number of correct decimals D_i for $m = 3$

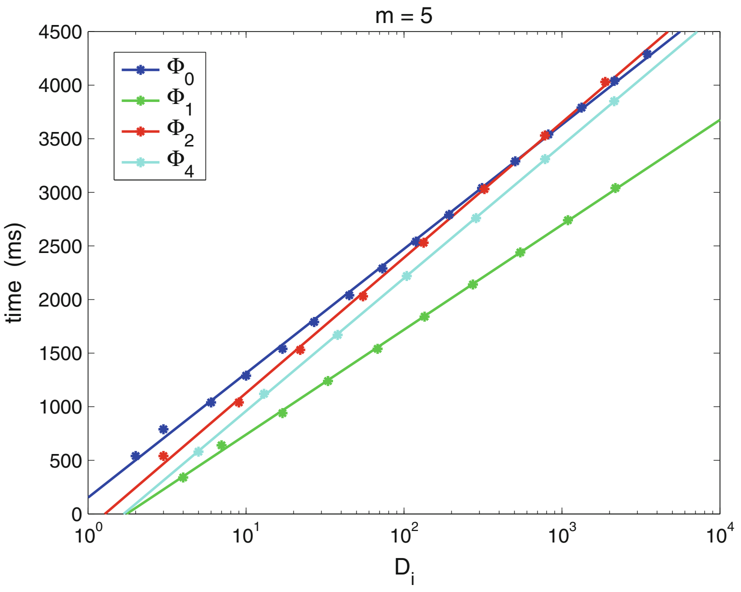

Fig. 10.2 Time t versus number of correct decimals D_i for $m = 5$

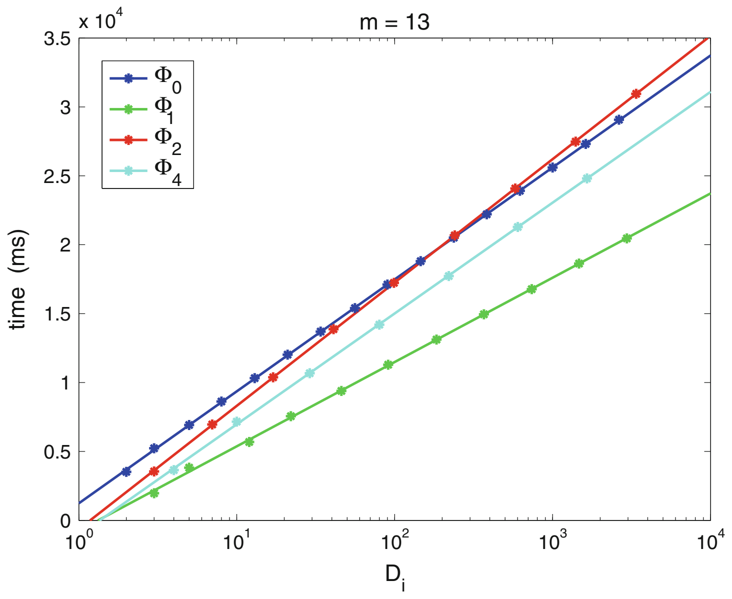

Fig. 10.3 Time t versus number of correct decimals D_i for $m = 13$

Table 10.11 Numerical results for case 1, where $m = 3$ and $t_p = 0.1039$

Φ_i	TF	I	D_I	$\tilde{\Theta}$	$\tilde{\kappa}$	r_{TF}
Φ_0	4013.15	17	3130	1292	430.7	0.997
Φ_1	3705.95	11	3310	1123	398.1	1.024
Φ_2	4382.20	9	3960	1388	468.8	0.994
Φ_3	4453.27	7	2140	1187	475.7	1.008

Table 10.12 Numerical results for case 2, where $m = 5$ and $t_p = 0.1039$

Φ_i	TF	I	D_I	$\tilde{\Theta}$	$\tilde{\kappa}$	r_{TF}
Φ_0	11220.74	17	3470	3776	1159.8	0.985
Φ_1	9356.20	10	2180	2660	978.6	1.008
Φ_2	12252.59	8	1880	3561	1262.2	0.989
Φ_3	11762.06	7	2140	3216	1239.2	1.003

Table 10.13 Numerical results for case 3, where $m = 13$ and $t_p = 0.1039$

Φ_i	TF	I	D_I	$\tilde{\Theta}$	$\tilde{\kappa}$	r_{TF}
Φ_0	77967.67	17	2630	25266	8125.6	1.008
Φ_1	58546.41	11	2950	18205	6117.1	1.010
Φ_2	85137.03	9	3380	26995	8950.9	1.003
Φ_3	77305.42	7	1650	21007	8040.0	1.006

Tables 10.11, 10.12 and 10.13 show the time factor (*TF*); the last iteration reached (I); the approximated number of correct decimal places in the Ith iteration (D_I); the elapsed time $\tilde{\Theta}$; the slope $\tilde{\kappa}$; and the computed time factor $\widetilde{TF}$ defined by

$$\widetilde{TF} = \frac{\widetilde{\Theta}(D_I)}{t_p \, \log q} = \frac{\tilde{\kappa}}{t_p} \approx TF = \frac{1}{\log CEI}.$$

Furthermore, the last column shows the percentage of relative error r_{TF} between *TF* and $\widetilde{TF}$. Note that the ordering of the methods according to time factor (*TF*) (or *CEI*) matches the ordering according to $\tilde{\kappa}$.

Figures 10.1, 10.2 and 10.3 show in a semi-logarithmic graph the $(\log D_j, \Theta(D_j))$ pairs and the straight line of each method. Note that the smaller the slope of the line, the more efficient the method.

10.6 Ball of Local Convergence

In this section, we use the convergence ball to study an aspect of local convergence theory. For this purpose, we present an algorithm devised by Schmidt and Schwetlick [34] and subsequently studied by Potra and Pták [31]. The procedure consists of fixing a natural number k, and keeping the same linear operator of the Secant method for sections of the process consisting of k steps each. It may be described as follows: starting with $x_n, z_n \in D$, for $0 \le j \le k-1$, $n \ge 0$ and $x_n^{(0)} = z_n$,

$$x_n^{(j+1)} = \Phi_{4,j+1}(x_n;\, x_{n-1}) = x_n^{(j)} - [x_n, z_n; F]^{-1}\, F(x_n^{(j)}). \qquad (10.71)$$

In the two last steps we take $x_{n+1} = \Phi_{4,\,k-1}(x_n; x_{n-1}) = x_n^{(k-1)}$ and finally $z_{n+1} = \Phi_{4,\,k}(x_n; x_{n-1}) = x_n^{(k)}$.

The iterative method $\Phi_{4,\,k}$ defined in (10.71) has a local convergence order equal to at least $\rho_{4,\,k} = \frac{1}{2}\left(k + \sqrt{k^2+4}\right)$ [21].

We introduce a theorem (see [21]) on the local convergence of sequences defined in (10.71), following the ideas given in [8, 36]. We denote $B(\alpha, r)$ the open ball $\{x \in \mathbf{R^m};\ \|x-\alpha\| < r\}$.

Theorem 4 *Let α be a solution of $F(x) = 0$ such that $[F'(\alpha)]^{-1}$ exists. We suppose that there is a first order divided difference $[x, y; F] \in \mathscr{L}(D, \mathbf{R^m})$, for all $x, y \in D$, that satisfies*

$$\left\| [F'(\alpha)]^{-1} \left([x,y;F] - [u,v;F]\right) \right\| \le K\left(\|x-u\| + \|y-v\|\right), \ x, y, u, v \in D, \qquad (10.72)$$

and $B(\alpha, r) \subset D$, where $r = \dfrac{1}{5K}$. Then, for $x_0, z_0 \in B(\alpha, r)$, the sequence $\{x_n^{(j+1)}\}$, $0 \le j \le k-1$, $n \ge 0$, given in (10.71) is well-defined, belongs to $B(\alpha, r)$

and converges to α. *Moreover,*

$$\|x_n^{(j+1)} - \alpha\| = \frac{K\left(\|x_n - \alpha\| + \|z_n - \alpha\| + \|x_n^{(j)} - \alpha\|\right)}{1 - K\left(\|x_n - \alpha\| + \|z_n - \alpha\|\right)} \|x_n^{(j)} - \alpha\|. \tag{10.73}$$

Theorem 4 can be seen as a result of the accessibility of the solution in the following way, if $x_0, z_0 \in B(\alpha, r)$, where $r = \dfrac{1}{5K}$, the sequence $\{x_n^{(j+1)}\}$, given in (10.71), converges to α. The radius r could be slightly increased if we consider center-Lipschitz conditions, as in [3] or [4], for instance. In fact, let us assume that, together with (10.72), the following condition holds:

$$\left\| [F'(\alpha)]^{-1} \left([\alpha, \alpha; F] - [u, v; F]\right) \right\| \le K_1 \left(\|\alpha - u\| + \|\alpha - v\|\right), \quad u, v \in D. \tag{10.74}$$

Obviously K_1 is always less than or equal to K. Then, we can mimic the proof of Theorem 4 (see [21]) to obtain the radius $r_1 = 1/(2K_1 + 3K) \ge r$.

Application Now, we consider an application of the previous analysis to the nonlinear integral equation of mixed Hammerstein type of the form:

$$x(s) = 1 + \frac{1}{2}\int_0^1 G(s, t)\, x(t)^3\, dt, \quad s \in [0, 1],$$

where $x \in C[0, 1]$, $t \in [0, 1]$, and the kernel G is the Green function

$$G(s, t) = \begin{cases} (1 - s)\, t, \; t \le s, \\ s\,(1 - t), \; s < t. \end{cases}$$

Using the following Gauss-Legendre formula to approximate an integral, we obtain the following nonlinear system of equations

$$x_i = 1 + \frac{1}{2}\sum_{j=1}^{8} b_{ij}\, x_j^3, \quad b_{ij} = \begin{cases} \varpi_j t_j (1 - t_i), \text{ if } j \le i, \\ \varpi_j t_i (1 - t_j), \text{ if } j > i, \end{cases} \quad i = 1, \ldots, 8, \tag{10.75}$$

where the abscissas t_j and the weights ϖ_j are determined for $m = 8$ (see [6]). We have denoted the approximation of $x(t_i)$ by x_i $(i = 1, 2, \ldots, 8)$. The solution of this system is $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_4, \alpha_3, \alpha_2, \alpha_1)^t$, where

$$\alpha_1 \approx 1.00577, \; \alpha_2 \approx 1.02744, \alpha_3 \approx 1.05518 \text{ and } \alpha_4 \approx 1.07441.$$

The nonlinear system (10.75) can be written in the form

$$F(x) = x - \overline{1} - \frac{1}{2}\, B\,\hat{x} \;=\; 0,$$

for $x = (x_1, \ldots, x_m)^t$, $\overline{1} = (1, 1, \ldots, 1)^t$, $\hat{x} = (x_1^3, \ldots, x_m^3)^t$ and $B = (b_{ij})$.

Taking the definition of the divided difference operator (10.61) we have

$$[x, y; F] = I - \frac{1}{2}\, B\, \mathrm{diag}(\tilde{p}),$$

where $\tilde{p} \in \mathbf{R^m}$ and $\tilde{p}_i = x_i^2 + x_i y_i + y_i^2$, $1 \le i \le m$. The Fréchet-derivative of operator F is given by

$$F'(X) = I - \frac{3}{2}\, B\, \mathrm{diag}(q),$$

where $q \in \mathbf{R^m}$ and $q_i = X_i^2$, $1 \le i \le m$, and, in addition, we have $F'(X) - F'(Y) = -\frac{3}{2}\, B\, \mathrm{diag}(r)$, where $r \in \mathbf{R^m}$ and $r_i = X_i^2 - Y_i^2$, $1 \le i \le m$. Setting

$$\Omega = \{X \in \mathbf{R^m} \,|\, \|X\|_\infty \le \delta\} \qquad (10.76)$$

and taking norms we obtain

$$\|F'(X) - F'(Y)\| \;\le\; \frac{3}{2}\, \|B\| \max_{1 \le i \le m} |2c_i|\; \|X - Y\|,$$

where $c \in \Omega$, and we get

$$\|F'(X) - F'(Y)\| \;\le\; 3\,\delta\, \|B\|\, \|X - Y\|. \qquad (10.77)$$

The divided difference operator can also be written as follows (see [32]):

$$[x, y; F] = \int_0^1 F'\left(\tau x + (1-\tau) y\right)\, d\tau.$$

Then we have

$$\begin{aligned}
\|[x, y; F] - [u, v; F]\| &\le \int_0^1 \|F'\left(\tau x + (1-\tau)y\right) - F'\left(\tau u + (1-\tau)v\right)\|\, d\tau \\
&\le 3\,\delta\,\|B\| \int_0^1 \left(\tau\|x - u\| + (1-\tau)\|y - v\|\right)\, d\tau \\
&= \frac{3}{2}\,\delta\,\|B\|\,\left(\|x - u\| + \|y - v\|\right).
\end{aligned}$$

Table 10.14 Numerical results for the radius of the existence ball in the nonlinear system (10.75) for different values of δ defined in (10.76) and the corresponding K introduced in (10.78)

δ	K	r
1.3	0.29708	0.6732
1.2	0.27423	0.7293
1.1	0.25137	0.7956

Here r is the radius for the frozen Schmidt-Schwetlick method (Theorem 4)

Next, we compute an upper bound for the inverse of $F'(\alpha)$, $\|F'(\alpha)^{-1}\| \leq 1.233$, and finally, taking into account

$$\|F'(\alpha)^{-1}\left([x,y;F]-[u,v;F]\right)\| \leq \|F'(\alpha)^{-1}\|\,\frac{3}{2}\,\delta\,\|B\|\left(\|x-u\|+\|y-v\|\right),$$

we deduce the following value for the parameter K introduced in (10.72):

$$K = \frac{3}{2}\,\delta\,\|F'(\alpha)^{-1}\|\,\|B\|. \tag{10.78}$$

Table 10.14 shows the size of the balls centered in α, $r = 1/(5K)$, for the frozen Schmidt-Schwetlick method and for $x_{-1} = \overline{\delta}$ and $x_0 = \overline{1}$ (see Theorem 4).

The numerical computations were performed on an MPFR library of C++ multi-precision arithmetics with 4096 digits of mantissa. All programs were compiled by `g++(4.2.1)` for i686-apple-darwin1 with `libgmp (v.4.2.4)` and `libmpfr (v.2.4.0)` libraries on an Intel® Core i7, 2.8 GHz (64-bit machine) processor. For this hardware and software, the computational cost of the quotient respect to the product is $\ell = 1.7$. Within each example, the starting point is the same for all the methods tested. The classical stopping criteria $||e_{I+1}|| = ||x_{I+1}-\alpha|| < 10^{-\varepsilon}$ and $||e_I|| > 10^{-\varepsilon}$, where $\varepsilon = 4096$, is replaced by $||\tilde{e}_{I+1}|| = ||x_{I+1}-\tilde{\alpha}_{I+1}|| < \varepsilon$ and $||\tilde{e}_I|| > \varepsilon$, where $\tilde{\alpha}_n$ is obtained by the δ^2-Aitken procedure, that is

$$\tilde{e}_n = \left(\frac{(\delta x_{n-1}^{(r)})^2}{\delta^2 x_{n-2}^{(r)}}\right)_{r=1\div m} \tag{10.79}$$

where $\delta x_{n-1} = x_n - x_{n-1}$ and the stopping criterion is now $||\tilde{e}_{I+1}|| < 10^{-\eta}$, where $\eta = \left[\varepsilon\,(2\rho-1)/\rho^2\right]$. Note that this criterion is independent of the knowledge of the root (see [14]).

In this case the concrete values of the parameters for the method $\Phi_{4,k}$ are $(m,\mu) = (8,11)$. Taking as initial approximations $x_{-1} = \overline{1}$ and $x_0 = \overline{1.1}$, which satisfy the conditions of Theorem 4 (see Table 10.14), we compare the convergence of the methods $\Phi_{4,k}$, towards the root α. We get the results shown in Table 10.15.

Table 10.15 Numerical results for the nonlinear system (10.75), where we show I, the number of iterations, $\rho_{4,k}$, the local order of convergence, *CEI*, the computational efficiency index, *TF*, the time factor, D_I and the estimation of the corrected decimal number in the last iteration x_I

	I	$\rho_{4,k}$	*CEI*	*TF*	D_I
$\Phi_{4,5}$	4	5.193	1.000969201	1032.3	1477
$\Phi_{4,6}$	4	6.162	1.000979191	1021.8	2973
$\Phi_{4,7}$	3	7.140	1.000975729	1025.4	757

Finally, in the computations we substitute the computational order of convergence (COC) [43] by an extrapolation (ECLOC) denoted by $\tilde{\rho}$ and defined as follows

$$\tilde{\rho} = \frac{\ln ||\tilde{e}_I||}{\ln ||\tilde{e}_{I-1}||},$$

where $\tilde{e}_I$ is given in (10.79). If $\rho = \tilde{\rho} \pm \Delta\tilde{\rho}$, where ρ is the local order of convergence and $\Delta\tilde{\rho}$ is the error of ECLOC, then we get $\Delta\tilde{\rho} < 10^{-3}$. This means that in all computations of ECLOC we obtain at least three significant digits. Therefore, it is a good check of the local convergence orders of the family of iterative methods presented in this section.

10.7 Adaptive Arithmetic

In this section, we present a new way to compute the iterates. It consists of adapting the length of the mantissa to the number of significant figures that should be computed in the next iteration. The role of the local convergence order ρ is the key concept to forecast the precision of the next iterate. Furthermore, if the root is known, then an expression of the forecast of digits in terms of x_n and e_n is

$$\Delta_{e_n} = \lceil \rho\,(-\log_{10} ||e_n|| + 4) + \log_{10} ||x_n|| \rceil,$$

where 4 is a security term that is empirically obtained. When the root is unknown according to

$$D_k \approx -\log_{10} \|e_k\| \approx -\frac{\rho}{\rho - 1} \log_{10} \|\check{e}_k\|. \tag{10.80}$$

(see Sect. 10.2 for more details) we may use the following forecast formulae

$$\Delta_{\check{e}_n} = \left\lceil \frac{\rho^2}{\rho - 1} (-\log_{10} ||\check{e}_n|| + 2) + \log_{10} ||x_n|| \right\rceil,$$

where Δ_{e_n} and $\Delta_{\check{e}_n}$ are the lengths of the mantissa for the next iteration.

10.7.1 Iterative Method

The composition of two Newton's iteration functions is a well-known technique that allows us to improve the efficiency of iterative methods in the scalar case. The two-step iteration function obtained in this way is

$$\begin{cases} y = \mathscr{N}(x) = x - \dfrac{f(x)}{f'(x)}, \\ X = \mathscr{N}(y). \end{cases} \tag{10.81}$$

In order to avoid the computation and the evaluation of a new derivative function $f'(y)$ in the second step, some authors have proposed the following variant:

$$X = y - \frac{f(y)}{f'(x)}. \tag{10.82}$$

In this case, the derivative is "frozen" (we only need to calculate $f'(x)$ at each step). Note that the local order of convergence decreases from $\rho = 4$ to $\rho = 3$, but the efficiency index in (10.81) is $EI = 4^{1/4} = 1.414$, while in (10.82) we have an improvement: $EI = 3^{1/3} = 1.441$.

Chung in [5] considers a different denominator in the second step of (10.81); namely,

$$X = y - \frac{f(y)}{f'(x)\, h(t)}, \tag{10.83}$$

where $t = f(y)/f(x)$ and $h(t)$ is a real valued function. The conditions established by Chung [5] in order to obtain a fourth-order method with only three evaluations of functions, $f(x), f(y), f'(x)$, are the following

$$h(0) = 1, \ h'(0) = -2 \ \text{ and } \ |h''(0)| < \infty.$$

In particular, we consider a specific member of the King's family [25] defined by $h(t) = \dfrac{1}{1+2t}$ (see also [2]) and the second step of (10.81) is

$$X = y - (1+2t)\frac{f(y)}{f'(x)}.$$

Taking into account that

$$t = \frac{f(y)}{f(x)} = \frac{f(x) + f(y) - f(x)}{f(x)} = 1 + \frac{f(y) - f(x)}{f(x)},$$

Table 10.16 Computational cost of iterative methods Φ_5

	$a_0(m)$	$a_1(m)$	$\omega(m,\ell)$
Φ_5	m^2+m	m^2	$\frac{m}{6}(2m^2+3(3\ell+7)m+(15\ell-17))$

and $f(x) = -f'(x)\,(y-x)$, we have

$$t = 1 - \frac{f(y)-f(x)}{f'(x)\,(y-x)} = 1 - \frac{[x,y]_f}{f'(x)}, \tag{10.84}$$

we generalize to the operator T defined by

$$T = I - F'(x)^{-1}\,[x,y;F].$$

So we consider the iterative method

$$\begin{cases} y_n = x_n - F'(x_n)^{-1}\,F(x_n), \\ x_{n+1} = \Phi_5(x_n) = y_n - (I+2\,T)\;F'(x)^{-1}\,F(y) \\ \qquad\;\; = y_n - \big(3\,I - 2\,F'(x_n)^{-1}\,[x_n,y_n;F]\big)\;F'(x)^{-1}\,F(y). \end{cases} \tag{10.85}$$

This algorithm has been deduced independently by Sharma and Arora [37], who prove that the R-order is at least four. In Table 10.16 we present the parameters of the computational cost of this scheme.

10.7.2 Numerical Example

We have applied the method (10.85) to the system (10.62) for $m = 11$ and the initial vector x_0 with all its components equal to 0.1. The solution reached with this initial point is the vector α with their eleven components approximately equal to $0.09127652716\ldots$ In this case, taking into account $\mu_1 = \mu_0/m$, we have $(m,\mu_0,\mu_1) = (11, 76.4, 6.95)$.

Table 10.17 shows the correct decimals and forecast for this function using the Φ_5 method. In the first two rows, we consider the correct decimals number D_n using fixed arithmetic (FA) and adaptive arithmetic (AA). In the third and fourth rows, the forecasts of lengths of the mantissa are obtained in adaptive arithmetic when we know the root Δ_{e_n}, or not $\Delta_{\check{e}_n}$, respectively.

As can be seen, all the forecasts overestimate the real values. Note that the values of Δ_{e_n} and α are very similar.

The two first rows in Table 10.18 show the partial and total elapsed time (t_e and T_e) when the root is known. The third and fourth rows show these times ($t_{\check{e}}$ and $T_{\check{e}}$) when the root is unknown. Moreover, using fixed arithmetic, the total elapsed time

Table 10.17 Number of correct decimals and forecasts for each iteration $1 \le n \le 5$

n	1	2	3	4	5
D_n FA	12	51	209	839	3358
D_n AA	12	51	209	837	3346
Δ_{e_n}	32	63	221	850	3363
$\Delta_{\breve{e}_n}$	32	63	220	850	3369

Table 10.18 Partial and total elapsed time in ms for each iteration $1 \le n \le 5$

n	1	2	3	4	5
t_e	1.112	1.591	3.639	23.598	209.953
T_e	1.112	2.703	6.342	29.940	239.893
$t_{\breve{e}}$	1.215	1.722	3.902	25.352	226.234
$T_{\breve{e}}$	1.215	2.937	6.839	32.191	258.425

Table 10.19 Elapsed time in ms for 100,000 products

Digits	32	64	128	256	512	1024	2048	4096	8192	16,384	32,768
Time (ms)	5.0	9.1	16.0	33.1	89.1	274.6	820.1	2419.0	7190.9	19,019.7	53,083.7

to obtain the same solution is 1364.151 ms. Henceforth, the time in the forecast of the mantissa length is sufficiently short that we can state that the use of adaptive arithmetic is a technique five times faster than the fixed arithmetic technique (in fact, 5.7 using e and 5.3 using $\breve{e}$).

10.7.3 Practical Result

Note that, if the number of digits is increased, then the time needed to compute a product is also increased. In particular, when the number of digits is doubled from 256 digits the computational time is tripled (see Table 10.19). Following Karatsuba's algorithm [24], we have

$$t_n = a\,\Delta_n^{\lambda},$$

where $\lambda = \log_2 3 = 1.58496\ldots$ Note that $\Delta_n = D_n + \Lambda$ where D_n is the number of correct decimals and Λ is the number of integer digits of x_n. For the range $256 \le \Delta_n \le 4096$ we have $\Delta_n^{\lambda} \approx D_n^{\lambda}$. Therefore, from $D_{n+1} \approx \rho\, D_n$ we obtain

$$\mathscr{C}_{n+1} = a\,\Delta_{n+1}^{\lambda} \approx a\,D_{n+1}^{\lambda} \approx a\,\rho^{\lambda}\,D_n^{\lambda} \approx \rho^{\lambda}\,\mathscr{C}_n.$$

Denoting by $\mathscr{C}_I$ the computational cost of the last iteration, if we consider an infinite number of iterations, the total cost is

$$\widetilde{\mathscr{C}} = \mathscr{C}_I \left(1 + \frac{1}{\rho^\lambda} + \frac{1}{\rho^{2\lambda}} + \cdots\right) = \mathscr{C}_I \, \frac{\rho^\lambda}{\rho^\lambda - 1}.$$

If we only consider the last iterate then we have

$$r = \frac{\mathscr{C}_I}{\widetilde{\mathscr{C}}} = \frac{\rho^\lambda - 1}{\rho^\lambda}.$$

Notice that for $\rho = 4$ we have $r = 0.889$. From Table 10.18 we can deduce that in the two cases (knowledge of the root and no knowledge of the root) in adaptive arithmetic, the computational cost of the last iteration is 87.5 % of the total elapsed time. Actually, we can assert that for iterative methods of an order equal to or greater than the fourth order, we only need to consider the cost of the last iteration to obtain a first approximation.

References

1. Aitken, A.: On Bernoulli's numerical solution of algebraic equations. Proc. R. Soc. Edinb. **46**, 289–305 (1926)
2. Amat, S., Busquier, S., Plaza, S.: Dynamics of the King and Jarrat iterations. Aequationes Math. **69**, 212–223 (2005)
3. Argyros, I.K., Gutiérrez, J.M.: A unified approach for enlarging the radius of convergence for Newton's method and applications. Nonlinear Funct. Anal. Appl. **10**, 555–563 (2005)
4. Argyros, I.K., Gutiérrez, J.M.: A unifying local and semilocal convergence analysis of Newton-like methods. Adv. Nonlinear Var. Inequal. **10**, 1–11 (2007)
5. Chung, C.: Some fourth-order iterative methods for solving nonlinear equations. Appl. Math. Comput. **195**, 454–459 (2008)
6. Ezquerro, J.A., Grau-Sánchez, M., Grau, A., Hernández, M.A., Noguera, M., Romero, N.: On iterative methods with accelerated convergence for solving systems of nonlinear equations. J. Optim. Theory Appl. **151**, 163–174 (2011)
7. Ezquerro, J.A., Grau-Sánchez, M., Grau, A., Hernández, M.A., Noguera, M.: Analysing the efficiency of some modifications of the secant method. Comput. Math. Appl. **64**, 2066–2073 (2012)
8. Ezquerro, J.A., Grau-Sánchez, M., Grau, A., Hernández, M.A.: Construction of derivative-free iterative methods from Chebyshev's method. Anal. Appl. **11**(3), 1350009 (16 pp.) (2013)
9. Ezquerro, J.A., Grau-Sánchez, M., Hernández, M.A., Noguera, M.: Semilocal convergence of secant-like methods for differentiable and nondifferentiable operator equations. J. Math. Anal. Appl. **398**, 100–112 (2013)
10. Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P.: MPFR: a multiple-precision binary floating-point library with correct rounding. ACM Trans. Math. Softw. **33** (2007). doi:10.1145/1236463.1236468
11. Grau, M., Díaz-Barrero, J.L.: A weighted variant family of Newton's method with accelerated third-order convergence. Appl. Math. Comput. **186**, 1005–1009 (2007)

12. Grau-Sánchez, M., Gutiérrez, J.M.: Zero-finder methods derived from Obreshkov's techniques. Appl. Math. Comput. **215**, 2992–3001 (2009)
13. Grau-Sánchez, M., Noguera, M.: A technique to choose the most efficient method between secant method and some variants. Appl. Math. Comput. **218**, 6415–6426 (2012)
14. Grau-Sánchez, M., Noguera, M., Gutiérrez, J.M.: On some computational orders of convergence. Appl. Math. Lett. **23**, 472–478 (2010)
15. Grau-Sánchez, M., Grau, A., Díaz-Barrero, J.L.: On computational order of convergence of some multi-precision solvers of nonlinear systems of equations. ArXiv e-prints (2011). Available at http://arxiv.org/pdf/1106.0994.pdf
16. Grau-Sánchez, M., Grau, A., Noguera, M.: On the computational efficiency index and some iterative methods for solving systems of nonlinear equations. J. Comput. Appl. Math. **236**, 1259–1266 (2011)
17. Grau-Sánchez, M., Grau, A., Noguera, M.: Ostrowski type methods for solving systems of nonlinear equations. Appl. Math. Comput. **218**, 2377–2385 (2011)
18. Grau-Sánchez, M., Grau, A., Noguera, M.: Frozen divided differences scheme for solving systems of nonlinear equations. J. Comput. Appl. Math. **235**, 1739–1743 (2011)
19. Grau-Sánchez, M., Grau, A., Noguera, M., Herrero, J.R.: On new computational local orders of convergence. Appl. Math. Lett. **25**, 2023–2030 (2012)
20. Grau-Sánchez, M., Grau, A., Noguera, M., Herrero, J.R.: A study on new computational local orders of convergence. ArXiv e-prints (2012). Available at http://arxiv.org/pdf/1202.4236.pdf
21. Grau–Sánchez, M., Noguera, M., Gutiérrez, J.M.: Frozen iterative methods using divided differences "à la Schmidt-Schwetlick". J. Optim. Theory Appl. **160**, 93–948 (2014)
22. Hueso, J.L., Martínez, E., Torregrosa, J.R.: Third and fourth order iterative methods free from second derivative for nonlinear systems. Appl. Math. Comput. **211**, 190–197 (2009)
23. Jay, L.O.: A note on Q-order of convergence. BIT **41**, 422–429 (2001)
24. Karatsuba A., Ofman, Y.: Multiplication of many-digital numbers by automatic computers. Proc. USSR Acad. Sci. **145**, 293–294 (1962). Transl. Acad. J. Phys. Dokl. **7**, 595–596 (1963)
25. King, R.F.: A family of fourth-order methods for nonlinear equations. SIAM J. Numer. Anal. **10**, 876–879 (1973)
26. Kurchatov, V.A.: On a method of linear interpolation for the solution of functional equations. Dokl. Akad. Nauk SSSR **198**, 524–526 (1971). Transl. Sov. Math. Dokl. **12**, 835–838 (1971)
27. Ortega, J.M., Rheinboldt, W.C.: Iterative Solution of Nonlinear Equations in Several Variables. Academic, New York (1970)
28. Ostrowski, A.M.: Solutions of Equations and System of Equations. Academic, New York (1960)
29. Petković, M.S.: Remarks on "On a general class of multipoint root-finding methods of high computational efficiency". SIAM J. Numer. Anal. **49**, 1317–1319 (2011)
30. Potra, F.A.: A characterisation of the divided differences of an operator which can be represented by Riemann integrals. Revue d'analyse numérique et de la théorie de l'approximation **2**, 251–253 (1980)
31. Potra, F.A., Pták, V.: A generalization of Regula Falsi. Numer. Math. **36**, 333–346 (1981)
32. Potra, F.A., Pták, V.: Nondiscrete Induction and Iterative Processes. Research Notes in Mathematics, vol. 103. Pitman Advanced Publishing Program, Boston (1984)
33. Ralston, A.: A First Course in Numerical Analysis. McGraw-Hill, New York (1965)
34. Schmidt, J.W., Schwetlick, H.: Ableitungsfreie Verfahren mit höherer Konvergenzgeschwindigkeit. Computing **3**, 215–226 (1968)
35. Schröder, E.: Über unendlich viele Algorithmen zur Auflösung der Gleichungen. Math. Ann. **2**, 317–365 (1870). Translated by G.W. Stewart, On Infinitely Many Algorithms for Solving Equations (1998). Available at http://drum.lib.umd.edu/handle/1903/577
36. Shakno, S.M.: On an iterative algorithm with superquadratic convergence for solving nonlinear operator equations. J. Comput. Appl. Math. **231**, 222–335 (2009)
37. Sharma, J.R., Arora, H.: On efficient weighted-Newton methods for solving systems of nonlinear equations. Appl. Math. Comput. **222**, 497–506 (2013)
38. The MPFR library 3.0.0. (2010). Timings in http://www.mpfr.org/mpfr-3.0.0/timings.html

39. The MPFR library 3.1.0. (2011). Available in http://www.mpfr.org
40. Tornheim, L.: Convergence of multipoint iterative methods. J. ACM **11**, 210–220 (1964)
41. Traub, J.F.: Iterative Methods for the Solution of Equations. Prentice-Hall, Englewood Cliffs (1964)
42. Wall, D.D.: The order of an iteration formula. Math. Tables Aids Comput. **10**, 167–168 (1956)
43. Weerakoon, S., Fernando, T.G.I.: A variant of Newton's method with accelerated third-order convergence. Appl. Math. Lett. **13**, 87–93 (2000)