Choosing the Optimal Multi-Point Iterative Method for the Colebrook Flow Friction Equation

Authors:

Pavel Praks, Dejan Brki?

Date Submitted: 2018-08-28

Keywords: pipes, hydraulic resistances, turbulent flow, three-point methods, iterative methods, Colebrook–White, Colebrook equation, explicit approximations

Abstract:

The Colebrook equation is implicitly given in respect to the unknown flow friction factor ?; ? = ? (R e , ? * , ?) which cannot be expressed explicitly in exact way without simplifications and use of approximate calculus. A common approach to solve it is through the Newton?Raphson iterative procedure or through the fixed-point iterative procedure. Both require in some cases, up to seven iterations. On the other hand, numerous more powerful iterative methods such as three- or two-point methods, etc. are available. The purpose is to choose optimal iterative method in order to solve the implicit Colebrook equation for flow friction accurately using the least possible number of iterations. The methods are thoroughly tested and those which require the least possible number of iterations to reach the accurate solution are identified. The most powerful three-point methods require, in the worst case, only two iterations to reach the final solution. The recommended representatives are Sharma?Guha?Gupta, Sharma?Sharma, Sharma?Arora, Džuni??Petkovi??Petkovi?; Bi?Ren?Wu, Chun?Neta based on Kung?Traub, Neta, and the Jain method based on the Steffensen scheme. The recommended iterative methods can reach the final accurate solution with the least possible number of iterations. The approach is hybrid between the iterative procedure and one-step explicit approximations and can be used in engineering design for initial rough, but also for final fine calculations.

Record Type: Published Article

Submitted To: LAPSE (Living Archive for Process Systems Engineering)

Citation (overall record, always the latest version):	LAPSE:2018.0434
Citation (this specific file, latest version):	LAPSE:2018.0434-1
Citation (this specific file, this version):	LAPSE:2018.0434-1v1

DOI of Published Version: https://doi.org/10.3390/pr6080130

processes

MDPI

Concept Paper

Choosing the Optimal Multi-Point Iterative Method for the Colebrook Flow Friction Equation

Pavel Praks [1,2,*,†] and Dejan Brkić [1,*,†]

[1] European Commission, Joint Research Centre (JRC), Directorate C - Energy, Transport and Climate, Unit C3: Energy Security, Distribution and Markets, Via Enrico Fermi 2749, 21027 Ispra (VA), Italy

[2] IT4Innovations National Supercomputing Center, VŠB—Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava, Czech Republic

* Correspondence: Pavel.Praks@ec.europa.eu or Pavel.Praks@vsb.cz (P.P.); dejanbrkic0611@gmail.com (D.B.)

† Both authors contributed equally to this study.

Received: 24 July 2018; Accepted: 14 August 2018; Published: 16 August 2018

Abstract: The Colebrook equation is implicitly given in respect to the unknown flow friction factor λ; $\lambda = \zeta(Re, \varepsilon^*, \lambda)$ which cannot be expressed explicitly in exact way without simplifications and use of approximate calculus. A common approach to solve it is through the Newton–Raphson iterative procedure or through the fixed-point iterative procedure. Both require in some cases, up to seven iterations. On the other hand, numerous more powerful iterative methods such as three- or two-point methods, etc. are available. The purpose is to choose optimal iterative method in order to solve the implicit Colebrook equation for flow friction accurately using the least possible number of iterations. The methods are thoroughly tested and those which require the least possible number of iterations to reach the accurate solution are identified. The most powerful three-point methods require, in the worst case, only two iterations to reach the final solution. The recommended representatives are Sharma–Guha–Gupta, Sharma–Sharma, Sharma–Arora, Džunić–Petković–Petković; Bi–Ren–Wu, Chun–Neta based on Kung–Traub, Neta, and the Jain method based on the Steffensen scheme. The recommended iterative methods can reach the final accurate solution with the least possible number of iterations. The approach is hybrid between the iterative procedure and one-step explicit approximations and can be used in engineering design for initial rough, but also for final fine calculations.

Keywords: Colebrook equation; Colebrook–White; iterative methods; three-point methods; turbulent flow; hydraulic resistances; pipes; explicit approximations

1. Introduction

The Colebrook function is to date the most used relation in engineering practice for evaluation of flow friction in pipes [1]. It is given in implicit form $\lambda = \zeta(Re, \varepsilon^*, \lambda)$, Equation (1):

$$x = -2 \cdot log_{10}\left(\frac{2.51 \cdot x}{Re} + \frac{\varepsilon^*}{3.71}\right) \quad (1)$$

In Equation (1), $x = \frac{1}{\sqrt{\lambda}}$ is introduced because of linearization of the unknown flow friction factor λ, Re is the Reynolds number, and ε^* is the relative roughness of inner pipe surface (all quantities are dimensionless). Because the Colebrook function in the examined iterative methods sometimes needs to be evaluated in two or three points, y and z are used in the same meaning as x, where they are dimensionless parameters that depend on the friction factor. Practical domain of applicability is for the Reynolds number, $4000 < Re < 10^8$ and for the relative roughness of the inner pipe surface, $0 < \varepsilon^* < 0.05$.

The Colebrook equation is empirical [2] and hence its accuracy can be disputed (experiment of Nikuradse or recent experiments from Oregon or Princeton research groups [3]), but anyway, it is widely accepted in engineering practice. It is based on an experiment conducted by Colebrook and White with the flow of air through a set of pipes with different inner roughness [2]. The turbulent part of the Moody diagram [4,5] is based on the Colebrook equation. The exact solution to this equation does not exist with the exception of those through the Lambert *W*-function [6–8], but anyway, the Lambert *W*-function can be evaluated only approximately [9–11]. Many different explicit approximations of the Colebrook equation exist in the form $\lambda = \varsigma(Re, \varepsilon^*)$, and they have different degrees of accuracy and complexity [12–14]. Many of the approximations are based on internal iterative cycles [15–17], and therefore, it is better to use more accurate iterative procedures if they require only few iterative steps. Calculation of flow through complex networks of pipes, such as for water or gas distribution, requires multiple evaluations of flow friction factor [18–23]. In general, the less number of iterations is required, the solution is more efficient with a decreased burden for computers [24–28]. Moreover, the recent approach, based on Padé polynomials, shows how the computational burden can be minimized with the same number of used iterations [29].

The most used iterative methods for solving the Colebrook equation are the Newton–Raphson method or its simplified version, the fixed point methods [30,31]. On the other hand, various iterative procedures for solving a single nonlinear equation are available and here are tested, in total, 23 different methods: (i) Fixed-point [30,31]; (ii) Newton–Raphson [32,33]; Hansen–Patrick group of method described previously [34] such as: (iii) Halley [35]; (iv) Euler–Chebyshev [36,37]; and (v) Basto–Semiao–Calheiros method [38]; (vi) Super Halley [39]; (vii) Ostrowski method of King family [36,37,40]; (viii) Kung–Traub [41]; (ix) Maheshwari [42]; (x) Hermite interpolation [43] based on Jarratt method [44]; (xi) Khattri and Babajee [45]; (xii) Murakami [46]; (xiii) Neta [47]; (xiv) Chun-Neta [48] based on Kung–Traub method [41]; (xv) Wang–Liu [49] based on Hermit interpolation [43]; (xvi) Bi–Ren–Wu [50]; (xvii) Jain [51] based on Steffensen method [52–55]; (xviii) Sharma–Arora [56]; (xix) Džunić-Petković-Petković [57–59]; (xx) Neta–Johnson [60] based on Jarrat method [44]; (xxi) Cordero et al. [54,55]; (xxii) Sharma–Sharma [61]; and (xxiii) Sharma–Guha–Gupta method [62]. All methods [63] are thoroughly tested within the domain of applicability of the Colebrook equation.

Details about preparation of data for analysis of the iterative methods are given in Section 2, while the methods are shown in Section 3 along with numerical examples. Discussion and analysis are given in Section 4, while concluding remarks and recommendations are in Section 5.

2. Preparation of Data for Analysis

Calculation of complex networks of pipes for distribution of water or natural gas is a computationally demanding task, where flow friction needs to be evaluated a few million times, which can cause a significant burden for computers. Moreover, a fast but reliable approximation of pipeline hydraulics is needed also for probabilistic risk assessment of pipeline networks. A large number of network simulations of random component failures and their combinations must be automatically evaluated and statistically analyzed in this case [20,22]. These tasks can cause significant a burden for computing, which even powerful computer resources cannot easily deal with. In general logarithmic functions noninteger power terms are extremely demanding, and therefore their use should be discouraged [24–27]. Both are used frequently in explicit approximations, but noninteger power terms occur very rarely in iterative procedures. So, the main goal of this paper is to find an optimal iterative solution for the Colebrook equation, which leads to the accurate solution within the domain of applicability of the Colebrook equation using the least possible number of iterations. The Colebrook equation is based on logarithmic law and therefore it is hardly possible that the logarithmic function can be eliminated entirely from the iterative procedure [29].

To maintain compatibility in this analysis with the study of Brkić [13], which evaluates accuracy of explicit approximations of the Colebrook equation, all iterative methods are tested in 740 uniformly

distributed points over the domain of applicability of the Colebrook equation. Examples from five points of 740 in total are shown (five triplets $\{Re, \varepsilon^*\} \rightarrow \{\lambda\}$), of which three are randomly selected, and two are from the most problematic zones which need a slightly higher number of iterations to reach the demanded level of accuracy: (1) $Re = 3.78 \times 10^6$, $\varepsilon^* = 0.00854$; (2) $Re = 6.23 \times 10^4$, $\varepsilon^* = 0.012$; (3) $Re = 1.18 \times 10^7$, $\varepsilon^* = 0.032$; (4) $Re = 5.74 \times 10^7$, $\varepsilon^* = 0.0008$; and (5) $Re = 8.31 \times 10^3$, $\varepsilon^* = 0.024$. The corresponding final solutions are: (1) $x = 5.274511499$; (2) $x = 4.928634498$; (3) $x = 4.128359435$; (4) $x = 7.331277467$; and (5) $x = 4.22204103$. Examples 2 and 5 are those from the more problematic zones of the domain, which in general require more iterative steps to reach the desired accuracy.

The initial starting point for the examined iterative procedures is set to; $x_0 = 7.273626085$. This value is chosen after numerous tests over the domain and it is from the problematic zone, which needs, in general, more iterative cycles to reach the final accurate solution. The fixed initial starting point for the testing is a more suitable option, because the iterative procedures in that way can be compared in a better way. Although the starting point x_0 for the iterative procedure can be chosen using different formulas [29], numerous tests showed that any fixed value within the domain of applicability of the Colebrook equation leads to the final accurate solution without significant variation of the required number of iterations. From the examined literature, cases when the iterative procedure diverges, oscillates, or converges outside the domain of applicability of the Colebrook equation are not reported.

For the purpose of all examined iterative methods, the Colebrook equation should be in the appropriate form as in Equation (2); for point x; $x \rightarrow F(x) = x - \zeta(x)$, where ζ is the functional symbol for the Colebrook equation. Sometimes it need to be evaluated in additional points $y \rightarrow F(y) = y - \zeta(y)$ and $z \rightarrow F(z) = z - \zeta(z)$. The first F' and the second derivatives F'' are also needed for some methods, but in most cases only in point x. Additional symbols specific for the certain method are explained in Section 3.

$$\left.\begin{array}{l} x \rightarrow F(x) = x + 2 \cdot log_{10}\left(\frac{2.51 \cdot x}{Re} + \frac{\varepsilon^*}{3.71}\right) \\ y \rightarrow F(y) = y + 2 \cdot log_{10}\left(\frac{2.51 \cdot y}{Re} + \frac{\varepsilon^*}{3.71}\right) \\ z \rightarrow F(z) = z + 2 \cdot log_{10}\left(\frac{2.51 \cdot z}{Re} + \frac{\varepsilon^*}{3.71}\right) \\ x \rightarrow F'(x) = \frac{5.02}{2.3026 \cdot Re \cdot \left(\frac{10 \cdot \varepsilon^*}{37.1} + \frac{2.51 \cdot x}{Re}\right)} + 1 \\ x \rightarrow F''(x) = -\frac{12.6}{2.3026 \cdot Re^2 \cdot \left(\frac{10 \cdot \varepsilon^*}{37.1} + \frac{2.51 \cdot x}{Re}\right)^2} \end{array}\right\} \quad (2)$$

In Equation (2), $\ln(10) \approx 2.3026$.

In Section 3, the presented iterative methods are listed in general from the simplest to the most complex [63]; (Section 3.1) One log-call per iteration: (Section 3.1.1); Fixed-point [30,31]; (Section 3.1.2) Newton–Raphson [32,33]; (Section 3.1.3) Hansen–Patrick [34]: (Section 3.1.3a) Halley [35], (Section 3.1.3b) Euler–Chebyshev [36,37], (Section 3.1.3c) Basto–Semiao–Calheiros method [38]; (Section 3.1.4) Super Halley [39]; (Section 3.1.5) Murakami [46]; (Section 3.2) Two log-calls per iteration (two-point methods): (Section 3.2.1) Ostrowski method of the King family [36,37,40]; (Section 3.2.2) Kung–Traub [41]; (Section 3.2.3) Maheshwari [42]; (Section 3.2.4) Khattri and Babajee [45]; (Section 3.2.5) Hermite interpolation [43] based on Jarratt method [44]; (Section 3.2.6) Wang–Liu method [49] based on Hermit interpolation [43]; and finally (Section 3.3) Three-log-calls per iteration (three-point methods): (Section 3.3.1) Neta [47]; (Section 3.3.2) Chun–Neta [48] based on Kung–Traub [41]; (Section 3.3.3) Džunić–Petković–Petković method [57–59]; (Section 3.3.4) Neta–Johnson [60] based on Jarrat method [44]; (Section 3.3.5) Jain [51] based on Steffensen method [52–55]; (Section 3.3.6) Bi–Ren–Wu [50]; (Section 3.3.7) Cordero et al. [54,55]; (Section 3.3.8) Sharma–Arora [56]; (Section 3.3.9) Sharma–Sharma [61]; and the (Section 3.3.10) Sharma–Guha–Gupta method [62].

The complexity of the methods cannot be evaluated only based on the number of log-calls per iteration, because many of them require evaluation of derivatives in one or more points. For example, the first derivative in point x is required for (2) Newton–Raphson, but not for (1) fixed-point method.

On the other hand, both methods require the same number of iterations to reach the same level of accuracy. Also, in certain cases, computationally costly log-calls in all subsequent iterations can be substituted with inexpensive Padé polynomials [64], as described by Praks and Brkić [29], but also in that way, the number of required iterations to reach the final desired accuracy remained unchanged. Therefore, for evaluation of the efficiency of each of the presented iterative methods, number of iterations are counted in each of the tested 740 points within the domain of applicability. Then, the highest value is chosen as the worst possible representative. We have presented five carefully selected numerical examples that were chosen from these 740, in order to illustrate the calculations.

3. Iterative Methods and Numerical Examples

Iterative methods with one log-call per iteration are presented in Section 3.1, with two log-calls in Section 3.2, and three-log calls in Section 3.3. With possible exceptions, they are sorted with increased complexity. As already explained, numerical examples are given for all presented methods.

In the following formulas indexes i and $i + 1$ refer to the two subsequent iterations.

In the following tests, the iterative procedures stop when the accuracy in respect to eight decimal places is reached or when sign #div0! appears in the meaning division with zero (desired accuracy reached). Final accurate solutions are marked with shading pattern. Moreover, the final accurate solutions that require the highest number of iterations for the observed methods (the worst cases), are highlighted in the framed cells.

3.1. Iterative Methods with One Log-Call Per Iteration

3.1.1. Fixed-Point Method

Fixed-Point method [30,31]; Equation (3):

$$x_{i+1} = x_i - F(x_i) \tag{3}$$

Example 1:	$x_1 = 5.274011505,$	$x_2 = 5.274511624,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.905054156,$	$x_2 = 4.928874894,$	$x_3 = 4.928632047,$	$x_4 = 4.928634523,$
	$x_5 = 4.928634490,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128292072,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331287607,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.124365599,$	$x_2 = 4.225356319,$	$x_3 = 4.221928724,$	$x_4 = 4.222044834,$
	$x_5 = 4.222040901,$	$x_6 = 4.222041034,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

3.1.2. Newton–Raphson Method

Newton–Raphson method [32,33]; Equation (4):

$$x_{i+1} = x_i - \frac{F(x_i)}{F'(x_i)} \tag{4}$$

$F'(x_i) = 1$ gives the fixed-point method [30,31].

Example 1:	$x_1 = 5.274061596,$	$x_2 = 5.274511600,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.907591018,$	$x_2 = 4.928826193,$	$x_3 = 4.928632752,$	$x_4 = 4.928634513,$
	$x_5 = 4.928634497,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128298809,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331286591,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.136669811,$	$x_2 = 4.224588192,$	$x_3 = 4.221965175,$	$x_4 = 4.222043289,$
	$x_5 = 4.222040962,$	$x_6 = 4.222041032,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

3.1.3. Hansen–Patrick Method; (a) Halley; (b) Euler–Chebyshev and (c) Basto, Semiao, and Calheiros Methods

Hansen–Patrick Method [34] is represented here through (a) Halley [35]; (b) Euler–Chebyshev [36,37] and (c) Basto, Semiao, and Calheiros methods [38]

(a) Halley method [35]; Equation (5):

$$x_{i+1} = x_i - \frac{\frac{F(x_i)}{F'(x_i)}}{1 - \frac{F''(x_i)}{2 \cdot F'(x_i)} \cdot \frac{F(x_i)}{F'(x_i)}} \tag{5}$$

Example 1:	$x_1 = 5.274061858,$	$x_2 = 5.274511600,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.908003000,$	$x_2 = 4.928822465,$	$x_3 = 4.928632785,$	$x_4 = 4.928634513,$
	$x_5 = 4.928634497,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128298811,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331286721,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.140502543,$	$x_2 = 4.224478344,$	$x_3 = 4.221968451,$	$x_4 = 4.222043191,$
	$x_5 = 4.222040965,$	$x_6 = 4.222041032,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

(b) Euler–Chebyshev method [36,37]; Equation (6):

$$x_{i+1} = x_i - \frac{F(x_i)}{F'(x_i)} - \frac{f^2(x_i) \cdot F''(x_i)}{2 \cdot \left(F'(x_i)\right)^3} \tag{6}$$

Example 1:	$x_1 = 5.274061740,$	$x_2 = 5.274511600,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.907907814,$	$x_2 = 4.928823333,$	$x_3 = 4.928632778,$	$x_4 = 4.928634513,$
	$x_5 = 4.928634497,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128298812,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331286591,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.141841176,$	$x_2 = 4.224438148,$	$x_3 = 4.221969647,$	$x_4 = 4.222043156,$
	$x_5 = 4.222040966,$	$x_6 = 4.222041032,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

(c) Basto-Semiao-Calheiros method [38]; Equation (7):

$$x_{i+1} = x_i - \frac{F(x_i)}{F'(x_i)} - \frac{f^2(x_i) \cdot F''(x_i)}{2 \cdot F'(x_i) \cdot \left(\left(F'(x_i)\right)^2 - F(x_i) \cdot F''(x_i)\right)} \tag{7}$$

Example 1:	$x_1 = 5.274061452,$	$x_2 = 5.274511600,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.907358658,$	$x_2 = 4.928828337,$	$x_3 = 4.928632732,$	$x_4 = 4.928634514,$
	$x_5 = 4.928634497,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128298808,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331286591,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.134234684,$	$x_2 = 4.224665949,$	$x_3 = 4.221962865,$	$x_4 = 4.222043358,$
	$x_5 = 4.222040960,$	$x_6 = 4.222041032,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

3.1.4. Super Halley Method

Super Halley method [39]; Equation (8):

$$\left.\begin{array}{c} x_{i+1} = x_i - \left(1 + \frac{1}{2} \cdot \frac{L}{1-L}\right) \cdot \frac{F(x_i)}{F'(x_i)} \\ L = \frac{F(x_i) \cdot F''(x_i)}{\left(F'(x_i)\right)^2} \end{array}\right\} \tag{8}$$

Auxiliary parameter L is introduced as described.

Example 1:	$x_1 = 5.274061740,$	$x_2 = 5.274511600,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.907907729,$	$x_2 = 4.928823333,$	$x_3 = 4.928632778,$	$x_4 = 4.928634513,$
	$x_5 = 4.928634497,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128298812,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435,$
	$x_5 = 4.128359435;$			
Example 4:	$x_1 = 7.331286591,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.141824182,$	$x_2 = 4.224438659,$	$x_3 = 4.221969632,$	$x_4 = 4.222043156,$
	$x_5 = 4.222040966,$	$x_6 = 4.222041032,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

3.1.5. Murakami Method

Murakami [46] method; Equation (9):

$$\left.\begin{aligned} x_{i+1} = x_i - 0.3 \cdot \frac{F(x_i)}{F'(x_i)} + \frac{1}{2} \cdot \frac{F(x_i)}{F'(\omega_i)} - \frac{2}{3} \cdot \frac{F(x_i)}{F'(\eta_i)} - \frac{32 \cdot F(x_i)}{75 \cdot F'(\omega_i) - 15 \cdot F(x_i)} \\ \omega_i = x_i - \frac{F(x_i)}{F'(x_i)} \\ \eta_i = x_i - \frac{1}{2} \cdot \frac{F(x_i)}{F'(x_i)} \end{aligned}\right\} \tag{9}$$

In addition to the point x, the Murakami method requires an additional evaluation of the function F in points ω and η, but only for the first derivative, which does not contain the logarithmic function.

Example 1:	$x_1 = 4.918789857,$	$x_2 = 5.226554297,$	$x_3 = 5.269211229,$	$x_4 = 5.273944807,$
	$x_5 = 5.274451138,$	$x_6 = 5.274505072,$	$x_7 = 5.274510815,$	$x_8 = 5.274511426,$
	$x_9 = 5.274511491,$	$x_{10} = 5.274511498,$	$x_{11} = 5.274511499,$	$x_{12} = 5.274511499;$
Example 2:	$x_1 = 4.253158283,$	$x_2 = 4.827210463,$	$x_3 = 4.917765778,$	$x_4 = 4.927553367,$
	$x_5 = 4.928527872,$	$x_6 = 4.928623991,$	$x_7 = 4.928633462,$	$x_8 = 4.928634396,$
	$x_9 = 4.928634487,$	$x_{10} = 4.928634497,$	$x_{11} = 4.928634497;$	
Example 3:	$x_1 = 2.187900719,$	$x_2 = 3.689925107,$	$x_3 = 4.066519315,$	$x_4 = 4.121441863,$
	$x_5 = 4.127617602,$	$x_6 = 4.128280272,$	$x_7 = 4.128350992,$	$x_8 = 4.128358535,$
	$x_9 = 4.128359339,$	$x_{10} = 4.128359425,$	$x_{11} = 4.128359434,$	$x_{12} = 4.128359435,$
	$x_{13} = 4.128359435;$			
Example 4:	$x_1 = 7.324855661,$	$x_2 = 7.330589866,$	$x_3 = 7.33120418,$	$x_4 = 7.331269659,$
	$x_5 = 7.331276635,$	$x_6 = 7.331277378,$	$x_7 = 7.331277457,$	$x_8 = 7.331277466,$
	$x_9 = 7.331277467,$	$x_{10} = 7.331277467;$		
Example 5:	$x_1 = 2.218159942,$	$x_2 = 3.806239564,$	$x_3 = 4.174433434,$	$x_4 = 4.21802767,$
	$x_5 = 4.221718266,$	$x_6 = 4.222015179,$	$x_7 = 4.22203896,$	$x_8 = 4.222040864,$
	$x_9 = 4.222041017,$	$x_{10} = 4.222041029,$	$x_{11} = 4.222041030,$	$x_{12} = 4.222041030.$

3.2. Iterative Methods with Two Log-Calls Per Iteration

3.2.1. Ostrowski Method of the King Family

Ostrowski method of the King Family [36,37,40]; Equation (10):

$$\left.\begin{aligned} x_{i+1} = y_i - \frac{F(y_i)}{F'(x_i)} \cdot \frac{F(x_i)}{F(x_i) - 2 \cdot F(y_i)} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \end{aligned}\right\} \tag{10}$$

Example 1:	$x_1 = 5.274511398,$	$x_2 = 5.274511499,$	$x_3 =$ #div0;	
Example 2:	$x_1 = 4.928451807,$	$x_2 = 4.928634512,$	$x_3 = 4.928634498,$	$x_4 = 4.928634498;$
Example 3:	$x_1 = 4.128359434,$	$x_2 = 4.128359435,$	$x_3 =$ #div0;	
Example 4:	$x_1 = 7.331277468,$	$x_2 = 7.331277467,$	$x_3 =$ #div0;	
Example 5:	$x_1 = 4.219926077,$	$x_2 = 4.222042800,$	$x_3 = 4.222041028,$	$x_4 = 4.222041030,$
	$x_5 = 4.222041030.$			

3.2.2. Kung–Traub Method

Kung–Traub method [41]; Equation (11):

$$\left.\begin{array}{c} x_{i+1} = y_i - \frac{F(y_i)}{F'(x_i)} \cdot \frac{1}{\left(1 - \frac{F(y_i)}{F(x_i)}\right)^2} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \end{array}\right\} \quad (11)$$

Example 1:	$x_1 = 5.274511398,$	$x_2 = 5.274511499,$	$x_3 = \text{\#div0!};$	
Example 2:	$x_1 = 4.928450156,$	$x_2 = 4.928634513,$	$x_3 = 4.928634498,$	$x_4 = 4.928634498;$
Example 3:	$x_1 = 4.128359434,$	$x_2 = 4.128359435,$	$x_3 = \text{\#div0!};$	
Example 4:	$x_1 = 7.331277468,$	$x_2 = 7.331277467,$	$x_3 = \text{\#div0!};$	
Example 5:	$x_1 = 4.219864191,$	$x_2 = 4.222042905,$	$x_3 = 4.222041028,$	$x_4 = 4.222041030,$
	$x_5 = 4.222041030.$			

3.2.3. Maheshwari Method

Maheshwari method [42]; Equation (12):

$$\left.\begin{array}{c} x_{i+1} = x_i - \left(\left(\frac{F(y_i)}{F(x_i)}\right)^2 - \frac{F(x_i)}{F(y_i) - F(x_i)}\right) \cdot \frac{F(x_i)}{F'(x_i)} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \end{array}\right\} \quad (12)$$

Example 1:	$x_1 = 5.274511398,$	$x_2 = 5.274511499,$	$x_3 = \text{\#div0!};$	
Example 2:	$x_1 = 4.928446781,$	$x_2 = 4.928634513,$	$x_3 = 4.928634498,$	$x_4 = 4.928634498;$
Example 3:	$x_1 = 4.128359434,$	$x_2 = 4.128359435,$	$x_3 = \text{\#div0!};$	
Example 4:	$x_1 = 7.331277468,$	$x_2 = 7.331277467,$	$x_3 = \text{\#div0!};$	
Example 5:	$x_1 = 4.219731647,$	$x_2 = 4.222043139,$	$x_3 = 4.222041028,$	$x_4 = 4.222041030,$
	$x_5 = 4.222041030.$			

3.2.4. Khattri and Babajee Method

Khattri and Babajee method [45]; Equation (13):

$$\left.\begin{array}{c} x_{i+1} = y_i - \frac{F(x_i) \cdot F(y_i)}{F(x_i) - 2 \cdot F(y_i)} \cdot \left(\frac{3}{F'(x_i) + 0.001 \cdot F(y_i)} - \frac{2}{F'(x_i)}\right) \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \end{array}\right\} \quad (13)$$

Example 1:	$x_1 = 5.274511397,$	$x_2 = 5.274511499,$	$x_3 = \text{\#div0!};$	
Example 2:	$x_1 = 4.928450478,$	$x_2 = 4.928634513,$	$x_3 = 4.928634498,$	$x_4 = 4.928634498;$
Example 3:	$x_1 = 4.128359434,$	$x_2 = 4.128359435,$	$x_3 = \text{\#div0!};$	
Example 4:	$x_1 = 7.331277468,$	$x_2 = 7.331277467,$	$x_3 = \text{\#div0!};$	
Example 5:	$x_1 = 4.219904119,$	$x_2 = 4.222042819,$	$x_3 = 4.222041028,$	$x_4 = 4.222041030,$
	$x_5 = 4.222041030.$			

3.2.5. Hermite Interpolation Based on Jarratt Method

In Equation (14), H is Hermite interpolation polynomial. This method requires an evaluation of the first derivatives in points x, y, and z, but the function F needs to be evaluated only in points x and y.

Hermite Interpolation [43] based on Jarratt Method [44]; Equation (14):

$$\left.\begin{array}{c} x_{i+1} = z_i - \frac{H_i}{F'(z_i)} \\ y_i = x_i - \frac{2}{3}\cdot\frac{F(x_i)}{F'(x_i)} \\ z_i = x_i - \frac{1}{2}\cdot\frac{F(x_i)}{F'(x_i)}\cdot\left(1 + \frac{1}{1+\frac{3}{2}\cdot\left(\frac{F'(y_i)}{F'(x_i)}-1\right)}\right) \\ H_i = F(x_i) + F'(x_i)\cdot\frac{(z_i-x_i)\cdot(z_i-y_i)^2}{(y_i-x_i)\cdot(x_i+2\cdot y_i-3\cdot z_i)} + F'(z_i)\cdot\frac{(z_i-y_i)\cdot(x_i-z_i)}{x_i+2\cdot y_i-3\cdot z_i} - \frac{F(x_i)-F(y_i)}{x_i-y_i}\cdot\frac{(z_i-x_i)^3}{(y_i-x_i)\cdot(x_i+2\cdot y_i-3\cdot z_i)} \end{array}\right\} \quad (14)$$

Example 1:	$x_1 = 5.274466557,$	$x_2 = 5.2745115,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.926606155,$	$x_2 = 4.928636193,$	$x_3 = 4.928634496,$	$x_4 = 4.928634498,$
	$x_5 = 4.928634498;$			
Example 3:	$x_1 = 4.128353373,$	$x_2 = 4.128359436,$	$x_3 = 4.128359435,$	$x_4 =$ #div0!;
Example 4:	$x_1 = 7.331278378,$	$x_2 = 7.331277467,$	$x_3 = 7.331277467;$	
Example 5:	$x_1 = 4.214067429,$	$x_2 = 4.222058401,$	$x_3 = 4.222040992,$	$x_4 = 4.222041030,$
	$x_5 = 4.222041030.$			

3.2.6. Wang–Liu Method Based on Hermit Interpolation

In Equation (15), H is Hermite interpolation polynomial, which is used in the same form as in Equation (14). This method also requires an evaluation of the first derivatives in points x, y, and z, but the function F needs to be evaluated only in points x and y.

Wang–Liu method [49] based on Hermit Interpolation [43]; Equation (15):

$$\left.\begin{array}{c} x_{i+1} = z_i - \frac{H_i}{F'(z_i)} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i = y_i - \frac{F(y_i)}{F'(x_i)}\cdot\frac{F(x_i)}{F(x_i)-2\cdot F(y_i)} \\ H_i = F(x_i) + F'(x_i)\cdot\frac{(z_i-x_i)\cdot(z_i-y_i)^2}{(y_i-x_i)\cdot(x_i+2\cdot y_i-3\cdot z_i)} + F'(z_i)\cdot\frac{(z_i-y_i)\cdot(x_i-z_i)}{x_i+2\cdot y_i-3\cdot z_i} - \frac{F(x_i)-F(y_i)}{x_i-y_i}\cdot\frac{(z_i-x_i)^3}{(y_i-x_i)\cdot(x_i+2\cdot y_i-3\cdot z_i)} \end{array}\right\} \quad (15)$$

Example 1:	$x_1 = 5.274061596,$	$x_2 = 5.2745116,$	$x_3 = 5.274511499,$	$x_4 = 5.274511499;$
Example 2:	$x_1 = 4.907590959,$	$x_2 = 4.928826193,$	$x_3 = 4.928632752,$	$x_4 = 4.928634513,$
	$x_5 = 4.928634497,$	$x_6 = 4.928634498,$	$x_7 = 4.928634498;$	
Example 3:	$x_1 = 4.128298809,$	$x_2 = 4.128359437,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331286591,$	$x_2 = 7.331277465,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 4.136665933,$	$x_2 = 4.224588276,$	$x_3 = 4.221965174,$	$x_4 = 4.222043289,$
	$x_5 = 4.222040962,$	$x_6 = 4.222041032,$	$x_7 = 4.222041030,$	$x_8 = 4.222041030.$

3.3. Iterative Methods with Three Log-Calls Per Iteration

3.3.1. Neta Method

Neta method [47]; Equation (16):

$$\left.\begin{array}{c} x_{i+1} = z_i - \frac{F(z_i)}{F'(x_i)}\cdot\frac{F(x_i)-F(y_i)}{F(x_i)-3\cdot F(y_i)} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i = y_i - \frac{F(y_i)}{F'(x_i)}\cdot\frac{F(x_i)-\frac{1}{2}\cdot F(y_i)}{F(x_i)-\frac{5}{2}\cdot F(y_i)} \end{array}\right\} \quad (16)$$

Example 1:	$x_1 = 5.274511499,$	$x_2 = 5.274511499;$	
Example 2:	$x_1 = 4.928632954,$	$x_2 = 4.928634498,$	$x_3 = 4.928634498;$
Example 3:	$x_1 = 4.128359435,$	$x_2 = 4.128359435;$	
Example 4:	$x_1 = 7.331277467,$	$x_2 = 7.331277467;$	
Example 5:	$x_1 = 4.221992945,$	$x_2 = 4.222041031,$	$x_3 = 4.222041031.$

3.3.2. Chun–Neta Method

Chun–Neta method [48] based on Kung–Traub [41]; Equation (17):

$$\left.\begin{aligned} x_{i+1} &= z_i - \frac{F(z_i)}{F'(x_i)} \cdot \frac{1}{\left(1-\frac{F(y_i)}{F(x_i)}-\frac{F(z_i)}{F(x_i)}\right)^2} \\ y_i &= x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i &= y_i - \frac{F(y_i)}{F'(x_i)} \cdot \frac{1}{\left(1-\frac{F(y_i)}{F(x_i)}\right)^2} \end{aligned}\right\} \tag{17}$$

Example 1:	$x_1 = 5.274511499,$	$x_2 = 5.274511499;$		
Example 2:	$x_1 = 4.928632854,$	$x_2 = 4.928634498,$	$x_3 = 4.928634498;$	
Example 3:	$x_1 = 4.128359435,$	$x_2 = 4.128359435;$		
Example 4:	$x_1 = 7.331277467,$	$x_2 = 7.331277467;$		
Example 5:	$x_1 = 4.221982464,$	$x_2 = 4.222041031,$	$x_3 = 4.222041030,$	$x_4 = 4.222041030.$

3.3.3. Džunić–Petković–Petković Method

Džunić–Petković–Petković method [57–59]; Equation (18):

$$\left.\begin{aligned} x_{i+1} &= z_i - \frac{F(z_i)}{F'(x_i)\cdot\left[1-2\cdot\frac{F(y_i)}{F(x_i)}-\left(\frac{F(y_i)}{F(x_i)}\right)^2\right]\cdot\left[1-\frac{F(z_i)}{F(y_i)}\right]\cdot\left[1-2\cdot\frac{F(z_i)}{F(x_i)}\right]} \\ y_i &= x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i &= y_i - \frac{F(x_i)}{F(x_i)-2\cdot F(y_i)} \cdot \frac{F(y_i)}{F'(x_i)} \end{aligned}\right\} \tag{18}$$

Example 1:	$x_1 = 5.274511499,$	$x_2 = \text{\#div0};$	
Example 2:	$x_1 = 4.928634483,$	$x_2 = 4.928634498,$	$x_3 = \text{\#div0};$
Example 3:	$x_1 = 4.128359435,$	$x_2 = \text{\#div0};$	
Example 4:	$x_1 = 7.331277467,$	$x_2 = \text{\#div0};$	
Example 5:	$x_1 = 4.222039554,$	$x_2 = 4.222041030,$	$x_3 = 4.222041030.$

3.3.4. Neta–Johnson Method Based on Jarrat Method

Neta–Johnson method [60] based on Jarrat method [44]; Equation (19):

$$\left.\begin{aligned} x_{i+1} &= z_n - \frac{F(z_i)}{F'(x_i)} \cdot \frac{F'(x_i)+F'(y_i)-F'(\delta_i)}{-2\cdot F'(x_i)+2\cdot F'(y_i)-F'(\delta_i)} \\ y_i &= x_i - \frac{F(x_i)}{F'(x_i)} \\ \delta_i &= x_i - \frac{1}{8}\cdot\frac{F(x_i)}{F'(x_i)} - \frac{3}{8}\cdot\frac{F(x_i)}{F(y_i)} \\ z_i &= x_i - \frac{F(x_i)}{\frac{1}{6}\cdot F'(x_i)+\frac{1}{6}\cdot F'(y_i)+\frac{2}{3}\cdot F'(\delta_i)} \end{aligned}\right\} \tag{19}$$

Auxiliary parameter δ is introduced as described.

Example 1:	$x_1 = 5.272711608,$	$x_2 = 5.27451312,$	$x_3 = 5.274511498,$	$x_4 = 5.274511499,$
	$x_5 = 5.274511499;$			
Example 2:	$x_1 = 4.843943256,$	$x_2 = 4.931741639,$	$x_3 = 4.928520569,$	$x_4 = 4.928638675,$
	$x_5 = 4.928634344,$	$x_6 = 4.928634503,$	$x_7 = 4.928634497,$	$x_8 = 4.928634498,$
	$x_9 = 4.928634498;$			
Example 3:	$x_1 = 4.128116927,$	$x_2 = 4.128359454,$	$x_3 = 4.128359435,$	$x_4 = 4.128359435;$
Example 4:	$x_1 = 7.331313968,$	$x_2 = 7.331277444,$	$x_3 = 7.331277467,$	$x_4 = 7.331277467;$
Example 5:	$x_1 = 3.873979004,$	$x_2 = 4.264698085,$	$x_3 = 4.21685805,$	$x_4 = 4.222671442,$
	$x_5 = 4.221964362,$	$x_6 = 4.222050354,$	$x_7 = 4.222039896,$	$x_8 = 4.222041168,$
	$x_9 = 4.222041013,$	$x_{10} = 4.222041032,$	$x_{11} = 4.222041030,$	$x_{12} = 4.222041030.$

3.3.5. Jain Method Based on Steffensen Method

Jain method [51] based on Steffensen Method [52–55]; Equation (20):

$$\left.\begin{array}{c} x_{i+1} = x_i - \frac{F^3(x_i)}{(F(x_i+F(x_i))-F(x_i))\cdot(F(x_i)-F(y_i))} \\ y_i = x_i - \frac{F^2(x_i)}{F(x_i+F(x_i))-F(x_i)} \end{array}\right\} \quad (20)$$

Jain method is a three-point method with evaluation of the function F in points x, y, but also in point $x_i + F(x_i)$. Serghides explicit approximation of the Colebrook equation is based on Steffensen method [16,65].

Example 1:	$x_1 = 5.274511499,$	$x_2 =5.274511499;$	
Example 2:	$x_1 = 4.928634582,$	$x_2 = 4.928634498,$	$x_3 =$ #div0;
Example 3:	$x_1 = 4.128359435,$	$x_2 =$ #div0;	
Example 4:	$x_1 = 7.331277467,$	$x_2 =$ #div0;	
Example 5:	$x_1 = 4.222058673,$	$x_2 = 4.222041030,$	$x_3 =$ #div0.

3.3.6. Bi–Ren–Wu Method

Bi–Ren–Wu method [50]; Equation (21):

$$\left.\begin{array}{c} x_{i+1} = z_i - \frac{F(z_i)}{(\Diamond_{zy})_i+(\Diamond_{yx})_i-F'(x_i)} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i = y_i - \frac{F(y_i)}{F'(x_i)}\cdot\frac{F(x_i)}{F(x_i)-2\cdot F(y_i)} \\ (\Diamond_{yx})_i = \frac{F(y_i)-F(x_i)}{y_i-x_i} \\ (\Diamond_{zy})_i = \frac{F(z_i)-F(y_i)}{z_i-y_i} \end{array}\right\} \quad (21)$$

Two-parameter function $\Diamond$ is introduced as defined.

Example 1:	$x_1 = 5.274511432,$	$x_2 = 5.274511499,$	$x_3 =$ #div0!;	
Example 2:	$x_1 = 4.928634504,$	$x_2 = 4.928634498,$	$x_3 =$ #div0!;	
Example 3:	$x_1 = 4.128359435,$	$x_2 = 4.128359435;$		
Example 4:	$x_1 = 7.331277468,$	$x_2 = 7.331277467,$	$x_3 =$ #div0!;	
Example 5:	$x_1 = 4.222041809,$	$x_2 = 4.222041029,$	$x_3 = 4.222041030,$	$x_4 = 4.222041030.$

3.3.7. Cordero et al. Method

Cordero et al. method [54,55]; Equation (22):

$$\left.\begin{array}{c} x_{i+1} = z_i - \frac{1+3\cdot\frac{F(z_i)}{F(x_i)}}{1+\frac{F(z_i)}{F(x_i)}}\cdot\frac{F(z_i)}{(\Diamond_{zy})_i+(\Diamond_{zxx})_i\cdot(z_i-y_i)} \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i = y_i - \frac{F(y_i)}{F'(x_i)}\cdot\frac{1}{1-2\cdot\frac{F(y_i)}{F(x_i)}-\left(\frac{F(y_i)}{F(x_i)}\right)^2-\frac{\left(\frac{F(y_i)}{F(x_i)}\right)^3}{2}} \\ (\Diamond_{zy})_i = \frac{F(z_i)-F(y_i)}{z_i-y_i} \\ (\Diamond_{zx})_i = \frac{F(z_i)-F(x_i)}{z_i-x_i} \\ (\Diamond_{zxx})_i = \frac{(\Diamond_{zx})_i-F'(x_i)}{z_i-x_i} \end{array}\right\} \quad (22)$$

Two-parameter and three-parameter functions $\Diamond$ are introduced as defined.

Example 1:	$x_1 = 5.274511398,$	$x_2 = 5.274511499,$	$x_3 =$ #div0!;	
Example 2:	$x_1 = 4.928453453,$	$x_2 = 4.928634512,$	$x_3 = 4.928634498,$	$x_4 = 4.928634498;$
Example 3:	$x_1 = 4.128359434,$	$x_2 = 4.128359435,$	$x_3 =$ #div0!;	
Example 4:	$x_1 = 7.331277468,$	$x_2 = 7.331277467,$	$x_3 =$ #div0!;	
Example 5:	$x_1 = 4.219987477,$	$x_2 = 4.2220427,$	$x_3 = 4.222041028,$	$x_4 = 4.222041030,$
	$x_5 = 4.222041030.$			

3.3.8. Sharma–Arora Method

Sharma–Arora method [56]; Equation (23):

$$\left.\begin{aligned} x_{i+1} &= z_i - \frac{\left(\Diamond_{zy}\right)_i}{\left(\Diamond_{zx}\right)_i} \cdot \frac{F(z_i)}{2\cdot\left(\Diamond_{zy}\right)_i - \left(\Diamond_{zx}\right)_i} \\ y_i &= x_i - \frac{F(x_i)}{F'(x_i)} \\ \left(\Diamond_{yx}\right)_i &= \frac{F(y_i)-F(x_i)}{y_i-x_i} \\ z_i &= y_i - \frac{F(y_i)}{2\cdot\left(\Diamond_{yx}\right)_i - F'(x_i)} \\ \left(\Diamond_{zx}\right)_i &= \frac{F(z_i)-F(x_i)}{z_i-x_i} \\ \left(\Diamond_{zy}\right)_i &= \frac{F(z_i)-F(y_i)}{z_i-y_i} \end{aligned}\right\} \tag{23}$$

Two-parameter function $\Diamond$ is introduced as defined.

Example 1:	$x_1 = 5.274511499,$	$x_2 =$ #div0!;	
Example 2:	$x_1 = 4.928634497,$	$x_2 = 4.928634498,$	$x_3 =$ #div0!;
Example 3:	$x_1 = 4.128359435,$	$x_2 =$ #div0!;	
Example 4:	$x_1 = 7.331277467,$	$x_2 =$ #div0!;	
Example 5:	$x_1 = 4.222040921,$	$x_2 = 4.222041030,$	$x_3 =$ #div0!.

3.3.9. Sharma–Sharma Method

Sharma–Sharma method [61]; Equation (24):

$$\left.\begin{aligned} x_{i+1} &= z_i - w_i \cdot \frac{F(z_i)\cdot\left(\Diamond_{xy}\right)_i}{\left(\Diamond_{xz}\right)_i\cdot\left(\Diamond_{yz}\right)_i} \\ y_i &= x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i &= y_i - \frac{F(y_i)}{F'(x_i)} \cdot \frac{1}{1-2\cdot\frac{F(y_i)}{F(x_i)}} \\ w_i &= 1 + \frac{\frac{F(z_i)}{F(x_i)}}{1+\frac{F(z_i)}{F(x_i)}} \\ \left(\Diamond_{xy}\right)_i &= \frac{F(x_i)-F(y_i)}{x_i-y_i} \\ \left(\Diamond_{xz}\right)_i &= \frac{F(x_i)-F(z_i)}{x_i-z_i} \\ \left(\Diamond_{yz}\right)_i &= \frac{F(y_i)-F(z_i)}{y_i-z_i} \end{aligned}\right\} \tag{24}$$

Two-parameter function $\Diamond$ is introduced as defined.

Example 1:	$x_1 = 5.274511499,$	$x_2 =$ #div0!;	
Example 2:	$x_1 = 4.928634483,$	$x_2 = 4.928634498,$	$x_3 =$ #div0!;
Example 3:	$x_1 = 4.128359435,$	$x_2 =$ #div0!;	
Example 4:	$x_1 = 7.331277467,$	$x_2 =$ #div0!;	
Example 5:	$x_1 = 4.222039549,$	$x_2 = 4.222041030,$	$x_3 = 4.22204103.$

3.3.10. Sharma–Guha–Gupta Method

Sharma–Guha–Gupta method [62]; Equation (25):

$$\left.\begin{array}{c} x_{i+1} = x_i - \frac{P+Q+R}{P\cdot(\Diamond_{zx})_i + Q\cdot F'(x_i) + R\cdot(\Diamond_{yx})_i}\cdot F(x_i) \\ y_i = x_i - \frac{F(x_i)}{F'(x_i)} \\ z_i = y_i - \frac{1}{1-2\cdot\frac{F(y_i)}{F(x_i)}}\cdot\frac{F(y_i)}{F'(x_i)} \\ P = (x_i - y_i)\cdot F(x_i)\cdot F(y_i) \\ Q = (y_i - z_i)\cdot F(y_i)\cdot F(z_i) \\ R = (z_i - x_i)\cdot F(z_i)\cdot F(x_i) \\ (\Diamond_{zx})_i = \frac{F(z_i)-F(x_i)}{z_i-x_i} \\ (\Diamond_{yx})_i = \frac{F(y_i)-F(x_i)}{y_i-x_i} \end{array}\right\} \quad (25)$$

Two-parameter function $\Diamond$ is introduced as defined, as well as auxiliary parameters P, Q and R.

Example 1:	$x_1 = 5.274511499$,	$x_2 =$ #div0!;	
Example 2:	$x_1 = 4.928634483$,	$x_2 = 4.928634498$,	$x_3 =$ #div0!;
Example 3:	$x_1 = 4.128359435$,	$x_2 =$ #div0!;	
Example 4:	$x_1 = 7.331277467$,	$x_2 =$ #div0!;	
Example 5:	$x_1 = 4.222039558$,	$x_2 = 4.222041030$,	$x_3 = 4.222041030$.

4. Summary—Discussion and Analysis

To summarize, the highest required number of iterations to reach the final accurate solution of the Colebrook equation in respect to eight decimal places, for the examined methods are:

3.1 One log-call per iteration (one-point methods)

- 3.1.1 Fixed-point [30,31]; Equation (3): 7 iterations
- 3.1.2 Newton–Raphson [32,33]; Equation (4): 7 iterations
- 3.1.3 Hansen–Patrick [34]:
  - 3.1.3a Halley [35], Equation (5): 7 iterations
  - 3.1.3b Euler–Chebyshev [36,37], Equation (6): 7 iterations
  - 3.1.3c Basto–Semiao–Calheiros method [38]; Equation (7): 7 iterations
- 3.1.4 Super Halley [39]; Equation (8): 7 iterations
- 3.1.5 Murakami [46]; Equation (9): 12 iterations

3.2 Two log-calls per iteration (two-point methods):

- 3.2.1 Ostrowski method of the King family [36,37,40]; Equation (10): 4 iterations
- 3.2.2 Kung–Traub [41]; Equation (11): 4 iterations
- 3.2.3 Maheshwari [42]; Equation (12): 4 iterations
- 3.2.4 Khattri and Babajee [45]; Equation (13): 4 iterations
- 3.2.5 Hermite interpolation [43] based on Jarratt method [44]; Equation (14): 4 iterations
- 3.2.6 Wang–Liu method [49] based on Hermit interpolation [43]; Equation (15): 7 iterations

3.3 Three log-calls per iteration (three-point methods):

- 3.3.1 Neta [47]; Equation (16): 2 iterations
- 3.3.2 Chun–Neta [48] based on Kung–Traub [41]; Equation (17): 2 iterations (3 in rare cases)
- 3.3.3 Džunić–Petković–Petković method [57–59]; Equation (18): 2 iterations

3.3.4 Neta–Johnson [60] based on Jarrat method [44]; Equation (19): 11 iterations
3.3.5 Jain [51] based on Steffensen method [52–55]; Equation (20): 2 iterations
3.3.6 Bi–Ren–Wu [50]; Equation (21): 3 iterations
3.3.7 Cordero et al. [54,55]; Equation (22): 4 iterations
3.3.8 Sharma–Arora [56]; Equation (23): 2 iterations
3.3.9 Sharma–Sharma [61]; Equation (24): 2 iterations
3.3.10 Sharma–Guha–Gupta method [62]; Equation (25): 2 iterations

After the conducted analysis the following methods should not be used for the Colebrook equation: (Section 3.1.5) Murakami [46]; Equation (9), (Section 3.2.6) Wang–Liu method [49] based on Hermit interpolation [43]; Equation (15), Neta–Johnson [60] based on Jarrat method [44]; Equation (19), and Cordero et al. [54,55]; Equation (22).

Most one-point methods that require one log-call per iteration need in our case seven iterations to reach the final solution. Among them as the simplest, (Section 3.1.1) Fixed-point [30,31]; Equation (3) can be recommended. Following procedure from Praks and Brkić [29], only one log-call is required in respect to the whole procedure and in particular only in the first iteration, whereas in all subsequent iterations log-calls are replaced by computationally inexpensive Padé polynomials.

Two log-calls iterative methods (two-point methods) require up to four iterations to reach the final solution. Those simpler can be used: (Section 3.2.1) Ostrowski method of the King family [36,37,40]; Equation (10), (Section 3.2.2) Kung–Traub [41]; Equation (11), and (Section 3.2.3) Maheshwari [42]; Equation (12).

Three log-calls methods (three-point methods) are the most powerful among the presented procedures. The simplest, but accurate, can be recommended for use: (Section 3.3.1) Neta [47]; Equation (16), (Section 3.3.2) Chun–Neta [48] based on Kung–Traub [41]; Equation (17), (Section 3.3.3) Džunić-Petković-Petković method [57–59]; Equation (18), and (Section 3.3.5) Jain [51] based on Steffensen method [52–55]; Equation (20). Those which are accurate, but have two-parameter functions $\Diamond$, can be used. However, for the Colebrook equation they seem to be too complex and their use should be limited to some rare very well elaborated cases and calculations: Sharma–Arora [56]; Equation (23), (Section 3.3.9) Sharma–Sharma [61]; Equation (24), and (Section 3.3.10) Sharma–Guha–Gupta method [62]; Equation (25).

Moreover, for three-point methods, choosing a different starting point x_0 compared with that selected for the shown numerical validation does not alter the number of required iterations significantly [29,66]. Thus, the most powerful three-point methods remain recommended even if we changed the initial starting point.

It should be noted that the Colebrook equation is empirical, and that today also many alternative equations exist. These equations are based on various experiments, where the most known are from Princeton or Oregon facilities [3,67,68]. The Princeton facility uses compressed air, while Oregon uses helium, oxygen, nitrogen, carbon dioxide, and sulfur hexafluoride. The Princeton device weights approximately 25 tons, while the Oregon device weighs approximately 0.7 kg.

5. Conclusions

An iterative solution of the Colebrook equation for flow friction in its domain of applicability represents a challenging task, in order to balance speed and accuracy of the iterative process. The fixed-point iterative methods are commonly used, but they demand up to seven iterations to reach the final accurate solution [30,31]. On the other hand, numerous explicit approximations with different degrees of accuracy are available [12–14]. Similar to the, as here presented, two-point and three-point iterative procedures are very accurate approximations of the Colebrook equation that also contain few internal iterative steps [15–17]. Herein we tested and proposed three-point iterative methods that are very accurate, and therefore can be used in a hybrid way: (1) as very accurate explicit approximations if the calculation is terminated after the first iteration; or (2) for very fine computing, in which a very

high accuracy can be easily reached after one or two more additional iterations. This is of a great significance in computing of flow through miscellaneous pipe-networks of various applications (water, natural gas, air, etc.), in which multiple evaluations of friction factors take place and when certain cases require only rough estimations; while other very accurate computational models need to be compared, repeated, or analyzed in detail [69–71].

Author Contributions: Both authors contributed equally to this study. The paper is a product of the joint efforts of the authors who worked together on models of natural gas distribution networks. P.P. has scientific background in applied mathematics and programming while D.B.'s background is in control and applied computing in mechanical and petroleum engineering. Following the idea by P.P., D.B. tested the methods and controlled the results. D.B. wrote the draft of this paper.

Funding: Resources to cover the Article Processing Charge were provided by the European Commission. This paper is registered in the internal system for publication of the European Commission; Pubsy as JRC112838.

Conflicts of Interest: The authors declare no conflict of interest. The views expressed are those of the authors and may not in any circumstances be regarded as stating an official position of the European Commission or of the VŠB—Technical University of Ostrava.

Abbreviations

The following symbols are used in this paper:

λ	Darcy friction factor (Moody, Darcy–Weisbach or Colebrook); dimensionless
Re	Reynolds number; dimensionless
ε^*	relative roughness of inner pipe surface; dimensionless
i	counter
$x = \frac{1}{\sqrt{\lambda}}$	linearization of friction factor—first point; dimensionless
$y = \frac{1}{\sqrt{\lambda}}$	linearization of friction factor—second point; dimensionless
$z = \frac{1}{\sqrt{\lambda}}$	linearization of friction factor—third point; dimensionless
Auxiliary parameters	
L	used in Super Halley method
ω	used in Murakami method
η	used in Murakami method
H	Hermite interpolation polynomial used in Jarratt method and in Wang-Liu method
δ	used in Neta–Johnson method based on Jarrat method
P, Q and R	Sharma–Guha–Gupta method
$\Diamond$	Two-parameter or three-parameter function used in Bi–Ren–Wu, Cordero et al., Sharma–Arora, Sharma–Sharma, Sharma–Guha–Gupta methods
Functional symbols:	
ζ	Colebrook equation
F	Colebrook equation in form; $F(x) = x - \zeta(x) = 0$
F'	first derivative
F''	second derivative
log_{10}	Briggs logarithm
ln	Napier natural logarithm

References

1. Colebrook, C.F. Turbulent flow in pipes with particular reference to the transition region between the smooth and rough pipe laws. *J. Inst. Civ. Eng. (Lond.)* **1939**, *11*, 133–156. [CrossRef]
2. Colebrook, C.; White, C. Experiments with fluid friction in roughened pipes. *Proc. R. Soc. Lond. Ser. A Math. Phys. Sci.* **1937**, *161*, 367–381. [CrossRef]
3. Vysotskiy, L.I. Guidance for the use of formulas for Darcy coefficient calculating the distribution of averaged velocities. *Sci. J. Russ. Res. Inst. Land Improv. Probl.* **2014**, *4*, 204–212. Available online: http://www.rosniipm-sm.ru/dl_files/udb_files/udb13-rec308-field6.pdf (accessed on 15 August 2018). (In Russian)
4. Moody, L.F. Friction factors for pipe flow. *Trans. ASME* **1944**, *66*, 671–684.

5. LaViolette, M. On the history, science, and technology included in the Moody diagram. *J. Fluids Eng.* **2017**, *139*, 030801. [CrossRef]
6. Keady, G. Colebrook-White formula for pipe flows. *J. Hydraul. Eng.* **1998**, *124*, 96–97. [CrossRef]
7. Brkić, D. W solutions of the CW equation for flow friction. *Appl. Math. Lett.* **2011**, *24*, 1379–1383. [CrossRef]
8. Brkić, D. Comparison of the Lambert W-function based solutions to the Colebrook equation. *Eng. Comput.* **2012**, *29*, 617–630. [CrossRef]
9. Corless, R.M.; Gonnet, G.H.; Hare, D.E.; Jeffrey, D.J.; Knuth, D.E. On the LambertW function. *Adv. Comput. Math.* **1996**, *5*, 329–359. [CrossRef]
10. Barry, D.A.; Parlange, J.Y.; Li, L.; Prommer, H.; Cunningham, C.J.; Stagnitti, F. Analytical approximations for real values of the Lambert W-function. *Math. Comput. Simul.* **2000**, *53*, 95–103. [CrossRef]
11. Visser, M. Primes and the Lambert W function. *Mathematics* **2018**, *6*, 56. [CrossRef]
12. Gregory, G.A.; Fogarasi, M. Alternate to standard friction factor equation. *Oil Gas J.* **1985**, *83*, 120–127.
13. Brkić, D. Review of explicit approximations to the Colebrook relation for flow friction. *J. Pet. Sci. Eng.* **2011**, *77*, 34–48. [CrossRef]
14. Winning, H.K.; Coole, T. Improved method of determining friction factor in pipes. *Int. J. Numer. Methods Heat Fluid Flow* **2015**, *25*, 941–949. [CrossRef]
15. Zigrang, D.J.; Sylvester, N.D. Explicit approximations to the solution of Colebrook's friction factor equation. *AIChE J.* **1982**, *28*, 514–515. [CrossRef]
16. Serghides, T.K. Estimate friction factor accurately. *Chem. Eng. (N. Y.)* **1984**, *91*, 63–64.
17. Romeo, E.; Royo, C.; Monzón, A. Improved explicit equations for estimation of the friction factor in rough and smooth pipes. *Chem. Eng. J.* **2002**, *86*, 369–374. [CrossRef]
18. Brkić, D. Iterative methods for looped network pipeline calculation. *Water Resour. Manag.* **2011**, *25*, 2951–2987. [CrossRef]
19. Brkić, D. A gas distribution network hydraulic problem from practice. *Pet. Sci. Technol.* **2011**, *29*, 366–377. [CrossRef]
20. Praks, P.; Kopustinskas, V.; Masera, M. Probabilistic modelling of security of supply in gas networks and evaluation of new infrastructure. *Reliab. Eng. Syst. Saf.* **2015**, *144*, 254–264. [CrossRef]
21. Brkić, D. Spreadsheet-based pipe networks analysis for teaching and learning purpose. *Spreadsheets Educ. (eJSiE)* **2016**, *9*, 4. Available online: https://epublications.bond.edu.au/ejsie/vol9/iss2/4/ (accessed on 15 August 2018).
22. Praks, P.; Kopustinskas, V.; Masera, M. Monte-Carlo-based reliability and vulnerability assessment of a natural gas transmission system due to random network component failures. *Sustain. Resilient Infrastruct.* **2017**, *2*, 97–107. [CrossRef]
23. Brkić, D. Discussion of "Economics and statistical evaluations of using Microsoft Excel solver in pipe network analysis" by I.A. Oke, A. Ismail, S. Lukman, S.O. Ojo, O.O. Adeosun, and M.O. Nwude. *J. Pipeline Syst. Eng. Pract.* **2018**, *9*, 07018002. [CrossRef]
24. Clamond, D. Efficient resolution of the Colebrook equation. *Ind. Eng. Chem. Res.* **2009**, *48*, 3665–3671. [CrossRef]
25. Giustolisi, O.; Berardi, L.; Walski, T.M. Some explicit formulations of Colebrook–White friction factor considering accuracy vs. computational speed. *J. Hydroinformatics* **2011**, *13*, 401–418. [CrossRef]
26. Danish, M.; Kumar, S.; Kumar, S. Approximate explicit analytical expressions of friction factor for flow of Bingham fluids in smooth pipes using Adomian decomposition method. *Commun. Nonlinear Sci. Numer. Simul.* **2011**, *16*, 239–251. [CrossRef]
27. Vatankhah, A.R. Approximate analytical solutions for the Colebrook equation. *J. Hydraul. Eng.* **2018**, *144*, 06018007. [CrossRef]
28. Brkić, D.; Ćojbašić, Ž. Evolutionary optimization of Colebrook's turbulent flow friction approximations. *Fluids* **2017**, *2*, 15. [CrossRef]
29. Praks, P.; Brkić, D. One-log call iterative solution of the Colebrook equation for flow friction based on Padé polynomials. *Energies* **2018**, *11*, 1825. [CrossRef]
30. Brkić, D. Solution of the implicit Colebrook equation for flow friction using Excel. *Spreadsheets Educ. (eJSiE)* **2017**, *10*, 2. Available online: http://epublications.bond.edu.au/ejsie/vol10/iss2/2 (accessed on 15 August 2018).
31. Brkić, D. Determining friction factors in turbulent pipe flow. *Chem. Eng. (N. Y.)* **2012**, *119*, 34–39.

32. Ypma, T.J. Historical development of the Newton–Raphson method. *SIAM Rev.* **1995**, *37*, 531–551. [CrossRef]
33. Abbasbandy, S. Improving Newton–Raphson method for nonlinear equations by modified Adomian decomposition method. *Appl. Math. Comput.* **2003**, *145*, 887–893. [CrossRef]
34. Hansen, E.; Patrick, M. A family of root finding methods. *Numer. Math.* **1976**, *27*, 257–269. [CrossRef]
35. Gander, W. On Halley's iteration method. *Am. Math. Mon.* **1985**, *92*, 131–134. [CrossRef]
36. Grau, M.; Díaz-Barrero, J.L. An improvement of the Euler–Chebyshev iterative method. *J. Math. Anal. Appl.* **2006**, *315*, 1–7. [CrossRef]
37. Grau, M.; Díaz-Barrero, J.L. An improvement to Ostrowski root-finding method. *Appl. Math. Comput.* **2006**, *173*, 450–456. [CrossRef]
38. Basto, M.; Semiao, V.; Calheiros, F.L. A new iterative method to compute nonlinear equations. *Appl. Math. Comput.* **2006**, *173*, 468–483. [CrossRef]
39. Gutierrez, J.M.; Hernández, M.A. An acceleration of Newton's method: Super-Halley method. *Appl. Math. Comput.* **2001**, *117*, 223–239. [CrossRef]
40. King, R.F. A family of fourth order methods for nonlinear equations. *SIAM J. Numer. Anal.* **1973**, *10*, 876–879. [CrossRef]
41. Kung, H.T.; Traub, J.F. Optimal order of one-point and multipoint iteration. *J. ACM (JACM)* **1974**, *21*, 643–651. [CrossRef]
42. Maheshwari, A.K. A fourth order iterative method for solving nonlinear equations. *Appl. Math. Comput.* **2009**, *211*, 383–391. [CrossRef]
43. De Boor, C.; Höllig, K.; Sabin, M. High accuracy geometric Hermite interpolation. *Comput. Aided Geom. Des.* **1987**, *4*, 269–278. [CrossRef]
44. Kou, J.; Li, Y. An improvement of the Jarratt method. *Appl. Math. Comput.* **2007**, *189*, 1816–1821. [CrossRef]
45. Khattri, S.K.; Babajee, D.K.R. Fourth-order family of iterative methods with four parameters. *Int. J. Math. Comput.* **2013**, *19*, 1–10. Available online: http://ceser.in/ceserp/index.php/ijmc/article/view/2605 (accessed on 15 August 2018).
46. Murakami, T. Some fifth order multipoint iterative formulae for solving equations. *J. Inf. Process.* **1978**, *1*, 138–139. Available online: http://id.nii.ac.jp/1001/00060028/ (accessed on 15 August 2018).
47. Neta, B. A sixth-order family of methods for nonlinear equations. *Int. J. Comput. Math.* **1979**, *7*, 157–161. [CrossRef]
48. Chun, C.; Neta, B. A new sixth-order scheme for nonlinear equations. *Appl. Math. Lett.* **2012**, *25*, 185–189. [CrossRef]
49. Wang, X.; Liu, L. New eighth-order iterative methods for solving nonlinear equations. *J. Comput. Appl. Math.* **2010**, *234*, 1611–1620. [CrossRef]
50. Bi, W.; Ren, H.; Wu, Q. Three-step iterative methods with eighth-order convergence for solving nonlinear equations. *J. Comput. Appl. Math.* **2009**, *225*, 105–112. [CrossRef]
51. Jain, P. Steffensen type methods for solving non-linear equations. *Appl. Math. Comput.* **2007**, *194*, 527–533. [CrossRef]
52. Khattri, S.K.; Agarwal, R.P. Derivative-free optimal iterative methods. *Comput. Methods Appl. Math.* **2010**, *10*, 368–375. [CrossRef]
53. Soleymani, F.; Vanani, S.K. Optimal Steffensen-type methods with eighth order of convergence. *Comput. Math. Appl.* **2011**, *62*, 4619–4626. [CrossRef]
54. Cordero, A.; Hueso, J.L.; Martínez, E.; Torregrosa, J.R. Steffensen type methods for solving nonlinear equations. *J. Comput. Appl. Math.* **2012**, *236*, 3058–3064. [CrossRef]
55. Cordero, A.; Fardi, M.; Ghasemi, M.; Torregrosa, J.R. Accelerated iterative methods for finding solutions of nonlinear equations and their dynamical behavior. *Calcolo* **2014**, *51*, 17–30. [CrossRef]
56. Sharma, J.R.; Arora, H. A new family of optimal eighth order methods with dynamics for nonlinear equations. *Appl. Math. Comput.* **2016**, *273*, 924–933. [CrossRef]
57. Džunić, J.; Petković, M.S.; Petković, L.D. A family of optimal three-point methods for solving nonlinear equations using two parametric functions. *Appl. Math. Comput.* **2011**, *217*, 7612–7619. [CrossRef]
58. Džunić, J.; Petković, M.S.; Petković, L.D. Three-point methods with and without memory for solving nonlinear equations. *Appl. Math. Comput.* **2012**, *218*, 4917–4927. [CrossRef]
59. Brkić, D.; Praks, P. Discussion of "Approximate analytical solutions for the Colebrook equation". *J. Hydraul. Eng.* in press.

60. Neta, B.; Johnson, A.N. High-order nonlinear solver for multiple roots. *Comput. Math. Appl.* **2008**, *55*, 2012–2017. [CrossRef]
61. Sharma, J.R.; Sharma, R. A new family of modified Ostrowski's methods with accelerated eighth order convergence. *Numer. Algorithm.* **2010**, *54*, 445–458. [CrossRef]
62. Sharma, J.R.; Guha, R.K.; Gupta, P. Improved King's methods with optimal order of convergence based on rational approximations. *Appl. Math. Lett.* **2013**, *26*, 473–480. [CrossRef]
63. Petković, M.; Neta, B.; Petković, L.; Džunić, J. *Multipoint Methods for Solving Nonlinear Equations*; Academic Press: Cambridge, MA, USA, 2012. [CrossRef]
64. Roy, D. Global approximation for some functions. *Comput. Phys. Commun.* **2009**, *180*, 1315–1337. [CrossRef]
65. Ćojbašić, Ž.; Brkić, D. Very accurate explicit approximations for calculation of the Colebrook friction factor. *Int. J. Mech. Sci.* **2013**, *67*, 10–13. [CrossRef]
66. Praks, P.; Brkić, D. Advanced iterative procedures for solving the implicit Colebrook equation for fluid flow friction. *Adv. Civ. Eng.* **2018**. [CrossRef]
67. Joseph, D.D.; Yang, B.H. Friction factor correlations for laminar, transition and turbulent flow in smooth pipes. *Phys. D Nonlinear Phenom.* **2010**, *239*, 1318–1328. [CrossRef]
68. Yang, B.H.; Joseph, D.D. Virtual Nikuradse. *J. Turbul.* **2009**, *1*, N11. [CrossRef]
69. Chandrasekhar, S.; Sharma, V.M. Brownian heat transfer enhancement in the turbulent regime. *Facta Univ. Ser. Mech. Eng.* **2016**, *14*, 169–177. [CrossRef]
70. Pambour, K.A.; Cakir Erdener, B.; Bolado-Lavin, R.; Dijkema, G.P. Development of a simulation framework for analyzing security of supply in integrated gas and electric power systems. *Appl. Sci.* **2017**, *7*, 47. [CrossRef]
71. Balokhonov, R.; Romanova, V.; Batukhtina, E.; Sergeev, M.; Emelianova, E. A numerical study of the microscale plastic strain localization in friction stir weld zones. *Facta Univ. Ser. Mech. Eng.* **2018**, *16*, 77–86. [CrossRef]