Computational Analysis of Two Numerical Solvers for Functional Fredholm Equations

AZUBUIKE WELI
Rivers State University
Department of Mathematics
Port Harcourt
NIGERIA

CHINEDU NWAIGWE
Rivers State University
Department of Mathematics
Port Harcourt
NIGERIA

Abstract: We construct two numerical methods for solving nonlinear functional Fredholm integral equations and compare their accuracy and computational costs. First, the nonlinear integral is approximated with composite trapezoid rule on the mesh. Then Picard and Newton iterations are designed to linearize the nonlinear system and approximate the solution. The in-built python function, *scipy.optimize.fsolve*, is used to handle the nonlinear system arising from the Newton approach. Several numerical experiments are performed to test the performance of the two methods. The results show that (i) both schemes are convergent and have the correct order of accuracy which is two, (ii) the methods have comparable accuracy, (iii) the Picard scheme is far more efficient than the Newton method, and (iv) the efficiency of the Picard scheme over the Newton method increases as the size of the problem increases. In addition to the above results, the Picard scheme is easier to program. We conclude that if the second order trapezoid rule is used to approximate the integral in a Fredholm equation, then the Picard scheme should be preferred over the Newton scheme. Further study is recommended to ascertain if this conclusion still holds when trapezoid rule of different order or a different quadrature rule is used.

Key–Words: Fredholm Integral equations, Banach Fixed-Point Theorem, Discrete Picard Iteration, Newton iteration, Trapezoid rule, Collocation Method, Experimental Order of Convergence.

Received: June 15, 2022. Revised: December 29, 2022. Accepted: January 22, 2023. Published: February 22, 2023.

1 Introduction

When a nonlinear equation is evaluated at discrete points, it turns from being in an infinite dimensional space to a finite dimensional problem. This finite dimensional problem is nonlinear and, in general, there does not exist any analytical method to find it's exact solution. In this case nonlinear approximation methods, like Newton types and fixed point iterations come in handy. However, each method has its own mathematical limitations and strengths which impart on the robustness of the method. It is then expedient to investigate the performance of some of these methods on specific problems with considerable generality.

Consequently, the present study is aimed at investigating the accuracy and Computational efficiency of the Picard and Newton iterations when used to approximate the solution of nonlinear finite dimensional problem arising from collocating the following nonlinear Functional problem at mesh points in a one-dimensional interval:

$$\begin{aligned} \lambda u(x) &= p(x) + f(x, z(x)), \\ z(x) &= \int_{y=a}^{b} k(x, y, u(y))\, dy,\ x \in [a, b] \subset \mathbb{R}, \end{aligned} \tag{1}$$

where u is the unknown function, $0 = \lambda \in \mathbb{R}$ is a constant, $C[a, b] \ni p : \mathbb{R} \to \mathbb{R}, f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}, k : \mathbb{R}\times\mathbb{R}\times\mathbb{R} \to \mathbb{R}$ and k is nonlinear in u and may not be necessarily separable in x and y. Further assumptions are made later.

These equations have applications in neuron transport modelling [1] economics and optimal control [2, 3, 4, 5], physical sciences [6, 7, 8], epidemiology [9, 10], heat transfer [11], antenna wire modelling [12, 13], and applied mathematical methods [14, 15]. See also [16, 17, 18, 19, 20] for more applications.

These numerous applications have motivated mathematical research in theoretical aspects of integral equation. For example, in [21, 22, 23, 24], integral equations of functional non-mixed types are studied, while [25] used Picard-type iteration to investigate functional Volterra equations. Existence and uniqueness results for nonlinear Volterra equations can be found in [26]. Also, [27] used Banach fixed-point theory to investigate linear mixed Volterra-Fredholm equation, while nonlinear mixed integral equations are studied via Banach fixed point theory in [15].

If $f(x, z) = z, k$ is linear in u and can be separated as $k = h(x)g(y)u(y)$, then analytical solution can be obtained [28]. But in the general case where f is nonlinear in the second argument, or k is nonlinear in the third argument, or both, then closed form analytical solution does not exist. Hence, approaching problem (1) via numerical methods is important. This is why the problem of developing numerical methods to solve (1) has been the subject of much research. Numerical methods for integral equations have been devised using Linear programming [29], Taylor series [30], variational iteration methods [31, 32, 33, 34], CAS wavelets [35, 36], collocation methods [37, 38], and Picard-Trepezoidal rule [15]. Other related works can be found in [39, 40, 9, 10] and [41].

The works of Micula [15, 42, 43] and Nwaigwe [44] adopted the idea of fixed point iterations in solving integral equations, while those [45, 25, 46] had adopted Newton type methods. These are two different ways of approximating nonlinear systems associated with collocating integral equations at grid points. Except the work of [44] which tried to compare these two different approaches on Volterra equations, no other study has actually carried out a comparative study of these two approaches on one integral equation. This is the gap the current study fills, to compare fixed point approach and Newton approach on nonlinear functional integral equations. The methods are compared in terms of accuracy, CPU time, and also ease of computer programming.

In this work, we approach the numerical solution of (1) by combining collocation projection and trapezoid quadrature then applying Picard and Newton methods. The paper is organized as follows. In section 2, we discretize the problem and formulate the two nonlinear schemes. Examples are presented and discussed in section 3 while concluding remarks are made in section 4.

2 The Numerical Algorithms

We now formulate the two algorithms to approximate the solution of problem (1). Let us make some further assumptions on the problem. First, we assume that k and f are Lipschitz continuous with respect to their third and second arguments respectively , namely there exists $\alpha_k, \alpha_f \geq 0$ such that

$$|f(x, z_1) - f(x, z_2)| \leq \alpha_f |z_1 - z_2| \quad \forall \quad z_1, z_2,$$
$$|k(x, y, u) - k(x, y, v)| \leq \alpha_k |u - v| \quad \forall \quad u, v.$$

Secondly, we assume that the following inequality holds:

$$\alpha_k \alpha_f (b - a) < 1. \tag{2}$$

This second assumption is a contraction requirement [46, 15]. It allows to satisfy the hypothesis of the Banach Contraction Principle stated below. Without loss of generality, we set $\lambda = 1$ in what follows.

Let us now define the computational mesh. We choose an integer N greater than 1. Then define the mesh size $h = \frac{b-a}{N}$ and grid points $x_i = a + ih, i = 0, 1, \cdots, N$ and the mesh $\Omega_h = \{x_i = a + ih : i = 0, 1, \cdots, N\}$.

Now we collocate problem (1) at the mesh points to get

$$u(x_i) = p(x_i) + f(x_i, I_u(x_i)), \forall x_i \in \Omega_h, \tag{3}$$

where

$$I_u(x) = \int_a^b k(x, y, u(y))\, dy.$$

Next, we approximate $I_u(x)$ with a trapezoid rule [45, 47], namely

$$I_u(x) \approx I_{u,i} = \sum_{j=0}^{N} \xi_j^N k(x_i, x_j, u(x_j)). \tag{4}$$

where $\xi_j^N = 0.5h$ when $j = 0$ or $j = N$ and $\xi_j^N = h$ otherwise, see [46, 47, 45] It is easy to show that this approximation is second order accurate, see [44, 48] for example.

This leads to the following approximation:

$$u(x_i) - p(x_i) - f(x_i, I_{u,i}) = 0, \forall x_i \in \Omega_h. \tag{5}$$

This is a nonlinear system of $N+1$ equations in $N+1$ unknowns. Obviously some form of nonlinear solver is needed to tackle the system. In this study, we consider two approaches, namely the Picard iteration and the Newton method.

2.1 Picard Scheme

Theorem 2.1 (Banach Contraction Principle, see [26]). *Let $\mathbb{B}$ be a Banach space and U is a non-empty closed subset of $\mathbb{B}$, and $T : U \to U$. If T^m is a contraction for some $m > 0$, then T has a unique fixed point in U and the sequence $\{u_n\}_{n \geq 0}$ generated by $u_{n+1} = Tu_n$ converges to the fix point of T, for $u_0 \in U$.*

If we define the operator T by

$$(Tu)(x) = p(x) + f(x, I_u(x)),$$

then it can be shown, under assumption (2), that T is a contraction. Hence by Theorem 2.1, the sequence of vectors $\{u_{n,i}\}_{n\geq 1}$ generated by the iteration

$$u_{n+1,i} = Tu_{n,i} := p(x_i) + f(x_i, I_{u_{n,i}}) \quad \text{for each } x_i \in \Omega_h, \tag{6}$$

where u_0, i are suitably chosen initial values, converges to the exact solution of the nonlinear system (5). We adopt the initial values:

$$u_{0,i} = p(x_i) + f(x_i, I_{g,i}), \forall x_i \in \Omega. \tag{7}$$

The equations (6)-(7) make up the complete Picard iteration for problem (1).

2.2 The Newton Scheme

For the Newton method, we define the vectors $\vec{Q}(\vec{u}) = \{q_0(\vec{u}), q_1(\vec{u}), \cdots, q_N(\vec{u})\}$ with $\vec{u} = \{u_0, u_1, \cdots, u_N\}$ is a vector of approximation solutions at grid the respective grid points. The components of $\vec{Q}$ are

$$q_i(\vec{u}) = u_i - p(x_i) - f(x_i, I_{u_i}). \tag{8}$$

Hence, we consider the system:

$$\vec{Q}(\vec{u}) = 0. \tag{9}$$

Then we require the Jacobian with components

$$J_{i,j}(\vec{u}) = \frac{\partial q_i}{\partial u_j} = \begin{cases} 1 - \frac{\partial I_{u_i}}{\partial u_j}\frac{\partial f(x_i,z)}{z}\Big|_{z=I_{u_i}}, \text{ if } i = j, \\ -\frac{\partial I_{u_i}}{\partial u_j}\frac{\partial f(x_i,z)}{z}\Big|_{z=I_{u_i}}, \text{ otherwise.} \end{cases} \tag{10}$$

The equations (9),(10) and the initialization (7) make up the Newton scheme.

In [44], the Krasnolseskij iteration and Newton method are developed, implemented and compared for nonlinear functional Volterra integral equations. The two methods were written in python without using any python in-built function. The results showed that the Krasnoselkij iteration was more computationally efficient than the Newton method, though both methods have comparable accuracy. Bearing this in mind, we are curious to know if the python routine, scipy.optimize.fsolve, would speed up the Newton method. Hence, we write a full program for the Picard scheme. For the Newton method, we only write a python program that assembles the nonlinear system, then use the python fsolve to complete the solution process.

3 Numerical Experiments

Several numerical experiments are now presented to access the accuracy and computational costs of the two algorithms proposed in the previous section. The methods are implemented in python programming language. The solution of each example is computed on a sequence of grids with $2 \times 2^p, p = 1, 2, \cdots, 8$ mesh points. The numerical solutions are compared with the exact solution and the errors and experimental order of convergence (EOC) are computed. The Picard scheme is taken to converge and terminated whenever the 2-norm, $\|\vec{u}_{n+1} - \vec{u}_n\|_2$, is less than 8×10^{-11}, that is whenever

$$\sqrt{\sum_{i=0}^{N} |u_{n+1,i} - u_{n,i}|^2} \leq 8 \times 10^{-11},$$

whereas the convergence of Newton method is decided by the scipy.optimize.fsolve function. The error is computed in maximum norm, $e_h = \max_i |u(x_i) - u_i|_h$, where u_i is the converged numerical solution computed at mesh point x_i by either the Picard or Newton method on a grid with mesh size, h. Finally, the experimental order of convergence is computed as (see [49, 46, 50])

$$EOC = \frac{log\left(\frac{e_h}{e_{h/2}}\right)}{log(2)}. \tag{11}$$

The computational costs of (average CPU time taken by) the methods are also calculated. Each method is used to repeatedly solve a given problem a number of times while the CPU time is taken for each run, then the average is recorded. We use T^N and T^P to denoted the average CPU times for the Newton and Picard schemes respectively. The efficiency of the Picard scheme over the Newton scheme is calculated as

$$\% \text{ increase by Picard} = \frac{T^N - T^P}{T^N} \times 100\%.$$

3.0.1 Example 1

We consider the Hammerstein equation [51]

$$\begin{aligned} u(x) =& x + \frac{1}{40}(1 - e^1)e^{x^4} \\ &+ \frac{1}{10}\int_0^1 e^{x^4+y^4} u^3(y)dy, \quad x \in [0, 1]. \end{aligned}$$

The exact solution of this problem is $u(x) = x$, [3].

Table 1 displays the results for problem 1. It can be seen that both schemes converge the exact solution as the grid is refined and they do so with second order of convergence which is the correct theoretical order of accuracy of the quadrature rule used in the numerical formulations. This verifies two things, namely (i) that both methods are convergent, and (ii) that we correctly implemented the proposed methods. Further, the Table also shows that both methods have comparable accuracy which means that the method with less computational cost (measured in terms of CPU time) would be taken as the superior method.

Consequently, Table 2 shows the results of multiples (50) runs of each method on the example problems using different grids with 10, 20, 40, 60 and 100 grid points. The results show that when the Newton and Picard schemes were used to solve example 1 for 50 times on a grid of 10 points, the average CPU times used are 0.0069188 and 0.0058953 respectively. This means that the Picard scheme is about 14.8% more efficient than the Newton method for solving problem 1 on a mesh with 10 points. The Table also shows that when this experiment is repeated on grids with 20, 40, 60 and 100 grid points, the efficiency of the Picard scheme over the Newton's are approximately 44.5%, 66.85, 76.5% and 85% respectively. This reveals that the larger the problem size, the more efficient is the Picard scheme over the Newton method.

Table 1: Accuracy Results for Problem 1, $N =$ number of sub-intervals.

N	Error-Newton	EOC-Newton	Error-Picard	EOC-Picard
4	0.08215397405559721	-	0.0821539682451089	-
8	0.01263272597740328	2.7011643772453855	0.012632724267867212	2.701164470442306
16	0.0026861432241733763	2.2335578515808603	0.0026861422466735174	2.23355818135017
32	0.0006258181125600171	2.101720911707286	0.0006258174492788093	2.1017219157631555
64	0.00015135534026722297	2.0478038353261097	0.00015135473500205165	2.047808075569715
128	3.7235133191426684e-05	2.02320315539687	3.7234541182540326e-05	2.0232203239754862
256	9.235298133125625e-06	2.011434078098783	9.235131938734042e-06	2.01143710256069
512	2.299758743884439e-06	2.0056760116890477	2.299592787746718e-06	2.005754161455908

3.0.2 Example 2

Next, we consider the Hammerstein equation [51, 3]

$$u(x) = x + \frac{1}{20}\left(cos(x+e^1) - cos(1+x)\right) + \frac{1}{20}\int_0^1 \left(e^{u(y)} sin(x+e^y)\right) dy, \quad x \in [0,1].$$

The exact solution is also $u(x) = x$, [3]. The results of approximating the solution of this problem using the two methods are shown in Table 3 which shows the accuracy results and Table 4 which shows the efficiency of the methods. Table 3 shows that both methods converge to the exact solution and at the correct theoretical order of convergence which is 2. Again, both methods have comparable accuracy like in problem 1.

Table 2: CPU times for Example 1

Method	No. of Runs	Average CPU Time	Error
Results for 10 **Mesh Points**			
Newton	50	0.006918811798095703	0.007548990805491762
Picard	50	0.0058935308456420895	0.007548988498649489
Percentage increase: 14.81874319425492			
Results for 20 **Mesh Points**			
Newton	50	0.03925801753997803	0.001670124652474625
Picard	50	0.021760010719299318	0.001670123843565685
Percentage increase: 44.57180448009068			
Results for 40 **Mesh Points**			
Newton	50	0.25215151309967043	0.0003951871519729533
Picard	50	0.08373754024505616	0.00039518651748804423
Percentage increase: 66.79078415367019			
Results for 60 **Mesh Points**			
Newton	50	0.7878385829925537	0.0001725825435245909
Picard	50	0.18556668281555175	0.00017258193576474845
Percentage increase: 76.4461037043034			
Results for 100 **Mesh Points**			
Newton	50	3.4391810274124146	6.127946007650209e-05
Picard	50	0.5113987255096436	6.127886529649906e-05
Percentage increase: 85.13021787938823			

However, Table 4 shows that on using the Newton and Picard schemes to solve problem 2 fifty times on grids of 10, 20, 40, 60 and 100 grid points, the Picard scheme is more efficient than the Newton method by 54.4%, 71.4%, 83.1%, 88.2% and 92.6% respectively. This also re-establishes the efficiency of the Picard scheme over the Newton scheme.

Table 3: Accuracy Results for Problem 2, $N =$ number of sub-intervals.

N	Error-Newton	EOC-Newton	Error-Picard	EOC-Picard
4	0.004224576766553079	-	0.004224576781223677	-
8	0.000777530887556499	2.4418349213860946	0.0007775309116082596	2.441834881768488
16	0.00016929568236101478	2.1993548078316434	0.00016929570760759738	2.1993546373142827
32	3.964218274898901e-05	2.094436873426475	3.964220891017334e-05	2.094436136489716
64	9.598161710755448e-06	2.0462063762979414	9.598187859172214e-06	2.0462033980291707
128	2.361936844996748e-06	2.0227877303915958	2.3619631505100713e-06	2.0227755931565734
256	5.858622871279806e-07	2.0113369000221515	5.858886649168227e-07	2.0112880133525324
512	1.458930138920067e-07	2.0056507831479955	1.4589430841205342e-07	2.0057029363147616

3.0.3 Example 3

This example is constructed via the method of manufacture solution (MMS), see [52, 53, 54, 55, 56] for more applications of MMS. The problem is

$$u(x) = \frac{x^2(x^2(In2)^2 - In4 + 4)}{x^2(In2)^2 + 4} + f(x,z)$$

where $f(x,z) = \frac{x^2 z}{1+z^2}$, $z(x) = \int_0^1 \frac{2xyu}{1+u^2} dy$ The exact solution is $u(x) = x^2$.

Tables 5 and 6 display the accuracy and efficiency results of the methods on this problem. It is can be seen that the convergence comparable accuracy of the methods are ascertained, see Table 5.

Table 6 shows that on using the methods o solve problem 3 fifty times on grids of 10, 20, 40, 60 and

Table 4: CPU times for Example 2

Method	**No. of Runs**	**Average CPU Time**	**Error**
Results for 10 **Mesh Points**			
Newton	50	0.009309396743774415	0.0004701870720190682
Picard	50	0.004241676330566406 5	0.0004701870973055078
Percentage increase: 54.436614451919304			
Results for 20 **Mesh Points**			
Newton	50	0.05688747882843018	0.00010553111642674917
Picard	50	0.016275830268859863	0.00010553114237565886
Percentage increase: 71.38943295773933			
Results for 40 **Mesh Points**			
Newton	50	0.36586283683776855	2.5045781609334128e-05
Picard	50	0.06172049522399902	2.504580762352493e-05
Percentage increase: 83.13015452526894			
Results for 60 **Mesh Points**			
Newton	50	1.1574162149429321	1.0943794029660836e-05
Picard	50	0.136169753074646	1.0943820425768358e-05
Percentage increase: 88.23502286242292			
Results for 100 **Mesh Points**			
Newton	50	4.975657024383545	3.886881525083652e-06
Picard	50	0.3691722440719605	3.886907744776735e-06
Percentage increase: 92.58043224718249			

100 grid points, the Picard scheme demonstrates superior efficiency by by 47.7%, 65.9%, 79.6%, 85.8% and 90.8% respectively. This also re-affirms the efficiency of the Picard scheme over the Newton scheme.

Table 5: Accuracy Results for Problem 3, N = number of sub-intervals.

N	**Error-Newton**	**EOC-Newton**	**Error-Picard**	**EOC-Picard**
4	0.007035915687580774	-	0.007035915470433141	-
8	0.0013013783137267865	2.434697777027481	0.0013013782499045057	2.4346978032547058
16	0.00028357548921409403	2.1982356738117668	0.000283575408928316	2.1982360115143154
32	6.640097989030203e-05	2.0944564020007768	6.640089565124185e-05	2.094457823809909
64	1.607782339063668e-05	2.0461324235669562	1.6077738207886938e-05	2.0461382369401773
128	3.956429290630226e-06	2.022801229555495	3.95634387628796e-06	2.022824732229111
256	9.814624188919652e-07	2.0111940418568612	9.812806749387448e-07	2.011430073763261
512	2.444672335677467e-07	2.0052919157820166	2.4429676548365364e-07	2.0060310861088446

Table 6: CPU times for Example 3

Method	**No. of Runs**	**Average CPU Time**	**Error**
Results for 10 **Mesh Points**			
Newton	50	0.002906074523925781	0.0007874964197842615
Picard	50	0.0015208387374877929	0.0007874963480500874
Percentage increase: 47.66690513382602			
Results for 20 **Mesh Points**			
Newton	50	0.012638144493103028	0.00017675318829701858
Picard	50	0.0043107843399047855	0.0001767531060890004
Percentage increase: 65.89068638788473			
Results for 40 **Mesh Points**			
Newton	50	0.07177528858184815	4.195403804851949e-05
Picard	50	0.014618129730224609	4.195395335271357e-05
Percentage increase: 79.6334782916895			
Results for 60 **Mesh Points**			
Newton	50	0.2107717752456665	1.8331747347666294e-05
Picard	50	0.030028929710388185	1.833166220732707e-05
Percentage increase: 85.75286957877175			
Results for 100 **Mesh Points**			
Newton	50	0.8688078927993774	6.510883697963266e-06
Picard	50	0.07958637714385987	6.510798332914902e-06
Percentage increase: 90.83958861291818			

3.0.4 Example 4

Finally, we consider the nonlinear functional equation:

$$u(x) = -0.11\left[x^{16}\left(-72\sqrt{3}tan^{-1}\left(\frac{\sqrt{3}}{6}\right)+35\right)^4 + 486\right]^{\frac{1}{2}} + cos^{-1}(x^2) + f(x,z),$$

where, $f(x,z) = \sqrt{6+z^4}$, $z(x) = \int_0^1 \frac{x^4y^2cos(u)}{12+y^2}dy$. The exact solution is $u(x) = cos^-1(x^2)$.

Table 7 shows that both methods compute accurate results and have comparable accuracy, while Table 8 shows that the Picard scheme is more efficient than the Newton scheme.

Table 7: Accuracy Results for Problem 4, N = number of sub-intervals.

N	**Error-Newton**	**EOC-Newton**	**Error-Picard**	**EOC-Picard**
4	1.1146974633510867e-08	-	1.1146974454590008e-08	-
8	1.676186014726016e-09	2.733398039695799	1.676185856780421e-09	2.7333981524829443
16	3.5208659535372973e-10	2.251180053937738	3.5208636006700544e-10	2.251180882094979
32	8.180157623342013e-11	2.1057297541209055	8.180123245438153e-11	2.105734853097954
64	1.9772975804176246e-11	2.0485986317872236	1.9772627979364188e-11	2.0486179472635424
128	4.866242116614986e-12	2.0226500034185118	4.865441383117286e-12	2.02286203789714
256	1.2091836980419395e-12	2.0087746682859793	1.2088108292118704e-12	2.008982199201384
512	3.036896641723089e-13	1.9933637205853214	3.028688411177427e-13	1.9968234224798223

Table 8: CPU times for Example 4

Method	**No. of Runs**	**Average CPU Time**	**Error**
Results for 10 **Mesh Points**			
Newton	50	0.006128149032592773	9.956891196578677e-10
Picard	50	0.0011171436309814454	9.956893087803564e-10
Percentage increase: 81.77029270926869			
Results for 20 **Mesh Points**			
Newton	50	0.030853209495544435	2.186125519041151e-10
Picard	50	0.0037441015243530273	2.1861223942210017e-10
Percentage increase: 87.86479077681135			
Results for 40 **Mesh Points**			
Newton	50	0.20467841148376464	5.1640339883071304e-11
Picard	50	0.013835945129394532	5.163958149978498e-11
Percentage increase: 93.24015413785249			
Results for 60 **Mesh Points**			
Newton	50	0.6568456077575684	2.254631241006722e-11
Picard	50	0.030254077911376954	2.254596509487783e-11
Percentage increase: 95.39403513488314			
Results for 100 **Mesh Points**			
Newton	50	2.8504409313201906	8.006390329098463e-12
Picard	50	0.08123322486877442	8.006040275176929e-12
Percentage increase: 97.15015231586817			

4 Conclusion

Two nonlinear equation solvers have been formulated, implemented and verified for approximating the solution of nonlinear Functional Fredholm integral equations. The two methods are Picard and Newton methods and were compared for accuracy and CPU time. The results show that (i) both formulations converge to the exact solution with the same order of convergence as the quadrature rule used in the formulations, (ii) that the Picard scheme is more efficient than the Newton scheme, and (iii) the percentage increase in the efficiency of the Picard scheme over the Newton scheme increases as the grid is refined. We conclude that if the second order trapezoid rule is used to approximate the integral in a Fredholm equation, then the Picard scheme should be preferred over Newton scheme. We also caution that this conclusion may not be valid when the quadrature rule is not second order trapezoid; the method and the order are emphasized. Further study is therefore recommended in that direction, to ascertain if this conclusion still holds when trapezoid rule of different order or a different quadrature rule is used.

References:

[1] I. K. Argyros, On a class of nonlinear integral equations arising in neutron transport, *aequationes mathematicae* 36, 1988, pp. 99–111.

[2] S. Hu, M. Khavanin, and W. Zhuang, Integral equations arising in the kinetic theory of gases, *Applicable Analysis* 34(3-4), 1989, pp. 261–266.

[3] A. Jerri, *Introduction to integral equations with applications,* John Wiley & Sons 1999.

[4] D. Oregan, Existence results for nonlinear integral equations, *Jour- nal of Mathematical Analysis and Applications* 192(3), 1995, pp. 705–726.

[5] J. Prss, *Evolutionary integral equations and applications,* 87 Birkhuser, 2013.

[6] M. A. Abdou, On a symptotic methods for fredholmvolterra integral equation of the second kind in contact problems, *Journal of Computational and Applied Mathematics* 154(2), 2003, pp. 431–446.

[7] T. D. Le, C. Moyne, M. A. Murad, and S. A. Lima, A two- scale non-local model of swelling porous media incorporating ion size correlation effects, *Journal of the Mechanics and Physics of Solids* 61(12), 2013, pp. 2493–2521.

[8] A. C. Rocha, M. A. Murad, C. Moyne, S. P. Oliveira, and T. D. Le, A new methodology for computing ionic profiles and dis- joining pressure in swelling porous media, *Computational Geosciences* 20(5), 2016, pp. 975–996.

[9] K. Maleknejad and M. Hadizadeh, A new computational method for volterra-fredholm integral equations, *Computers & Mathematics with Applications* 37(9), 1999, pp. 1–8.

[10] A. M. Wazwaz, A reliable treatment for mixed volterra fredholm integral equations, *Applied Mathematics and Computation* 127(2-3), 2002, pp. 405–414.

[11] W. E. Olmstead and R. A. Handelsman, Asymptotic solution to a class of nonlinear volterra integral equations II, *SIAM Journal on Applied Mathematics* 30(1), 1976, pp. 180–189.

[12] A. Alipanah, Solution of hallens integral equation by using radial basis functions, *MATHEMATICAL REPORTS* 15(3), 2013, pp. 211–220.

[13] K. Mei, On the integral equations of thin wire antennas, *IEEE Trans- actions on Antennas and Propagation* 13(3), 1965, pp. 374–378.

[14] A. M. Wazwaz, *First Course In Integral Equations, A,* World Scientific Publishing Company, 2015.

[15] S. Micula, On some iterative numerical methods for mixed volterra fredholm integral equations, *Symmetry* 11(10), 2019, pp. 1200.

[16] J. Honerkamp and J. Weese, Tikhonovs regularization method for ill-posed problems: A comparison of different methods for the deter- mination of the regularization parameter, *Continuum Mechanics and Thermodynamics* 2, 1990, pp. 17–30.

[17] S. S. Ray, P.K. Sahu, et al, Numerical methods for solving fredholm integral equations of second kind, *In Abstract and Applied Analysis* 2013, 2013.

[18] H. Schfer, E. Sternin, R. Stannarius, M. Arndt, and F. Kremer, Novel ap- proach to the analysis of broadband dielectric spectra, *Physical review letters* 76(12), 1996, pp. 2177.

[19] A. Daddi-Moussa-Ider, B. Kaoui, and H. Lwen, Ax- isymmetric flow due to a stokeslet near a finite-sized elastic membrane, *Journal of the Physical Society of Japan* 88(5), 2019, pp. 054401.

[20] A. Daddi-Moussa-Ider, Asymmetric stokes flow induced by a transverse point force acting near a finite-sized elastic membrane, *Jour- nal of the Physical Society of Japan* 89(12), 2020, pp. 124401.

[21] K. Maleknejad, R. Mollapourasl, and K. Nouri, Study on existence of solutions for some nonlinear functionalintegral equations, *Nonlinear Analysis: Theory, Methods & Applications* 69(8), 2008, pp. 2582–2588.

[22] H. k. Pathak et al, Study on existence of solutions for some nonlinear functional-integral equations with applications, *Mathematical Communications* 18(1), 2013, pp. 97–107.

[23] P.M. Fitzpatrick, Klaus deimling, nonlinear functional analysis, *Bul- letin (New Series) of the American Mathematical Society* 20(2), 1989, pp. 277–280.

[24] J. Bana and B. Rzepka, On existence and asymptotic stability of solutions of a nonlinear integral equation, *Journal of Mathematical Analysis and Applications* 284(1), 2003, pp. 165–173.

[25] S. Bazm, P. Lima, and S. Nemati, Analysis of the eu- ler and trapezoidal discretization methods for the numerical solution of nonlinear functional volterra integral equations of urysohn type, *Jour- nal of Computational and Applied Mathematics*2021, page 113628.

[26] K. Atkinson and W. Han, *Theoretical Numerical Analysis: A Functional Analysis Framework,* 39, Springer 2005.

[27] P. M. Hasan, N. A. Sulaiman, F. Soleymani, and A. Akgl, The existence and uniqueness of solution for linear system of mixed volterra-fredholm integral equations in banach space, *AIMS Mathematics* 5(1), 2020, pp. 226–235.

[28] G. Stephenson, *Mathematical methods for science students,* Courier Dover Publications, 2020.

[29] P. M. A. Hasan and N. A. Suleiman, Numerical solution of mixed volterra-fredholm integral equations using linear programming problem, *Applied Mathematics* 8(3), 2018, pp. 42–45.

[30] Z. Chen and W. Jiang, An approximate solution for a mixed lin- ear volterrafredholm integral equation, *Applied Mathematics Letters* 25(8), 2012, pp. 1131–1134.

[31] L. Xu, Variational iteration method for solving integral equations, *Computers & Mathematics with Applications* 54(7-8), 2007, pp. 1071–1078.

[32] S. S. Sheth, Dr. Singh, et al, An analytical approximate solution of linear, system of linear and non linear volterra integral equations using variational iteration method, *In Proceedings of International Confer- ence on Advancements in Computing & Management (ICACM)* 2019.

[33] S. A. Yousefi, A. Lotfi, and M. Dehghan, Hes variational iteration method for solving nonlinear mixed volterrafredholm inte- gral equations, *Computers & Mathematics with Applications* 58(11-12), 2009, pp. 2172–2176.

[34] A. Hamoud and K. Ghadle, On the numerical solution of nonlinear volterra-fredholm integral equations by variational iteration method, *Int. J. Adv. Sci. Tech. Research* 3, 2016, pp. 45–51.

[35] R. Ezzati and S. Najafalizadeh, Numerical methods for solving linear and nonlinear volterra-fredholm integral equations by using cas wavelets, *World Applied Sciences Journal* 18(12), 2012, pp. 1847–1854.

[36] S.C. Shiralashetti and L. Lamani, Cas wavelets stochastic opera- tional matrix of integration and its application for solving stochastic it-volterra integral equations, *Jordan Journal of Mathematics and Statistics (JJMS)* 14(3), 2021, pp. 555–580.

[37] Y. Ordokhani and M. Razzaghi, Solution of nonlinear volterrafredholmhammerstein integral equations via a collocation method and rationalized haar functions, *Applied Mathematics Letters* 21(1), 2008, pp. 4–9.

[38] H. Brunner, On the numerical solution of nonlinear volterra fredholm integral equations by collocation methods, *SIAM Journal on Numerical Analysis* 27(4), 1990, pp. 987–1000.

[39] I. Aziz et al, New algorithms for the numerical solution of nonlinear fredholm and volterra integral equations using haar wavelets, *Journal of Computational and Applied Mathematics* 239, 2013, pp. 333–345.

[40] K. E. Atkinson, The numerical solution of integral equations of the second kind, *Cambridge Monographs on Applied and Computational Mathematics*, 1996.

[41] Y. H. Youssri and R. M. Hafez, Chebyshev collocation treatment of volterrafredholm integral equation with error analysis, *Arabian Jour- nal of Mathematics* 9(2), 2020, pp. 471–480.

[42] S. Micula, Numerical solution of two-dimensional fredholmvolterra integral equations of the second kind, *Symmetry* 13(8) 2021, pp. 1326.

[43] S. Micula, Iterative numerical methods for a fredholm hammerstein integral equation with modified argument, *Symmetry* 15(1), 2023, pp. 66.

[44] C. Nwaigwe and S. Micula, Fast and accurate numerical algorithm with per- formance assessment for nonlinear functional volterra equations, *Re- searchgate.net*, 2022. https://www.researchgate.net/publication/3617229 35 Fast and Accurate Numerical Algorithm with Performance

[45] C. Nwaigwe and D. N. Benedict, Generalized ba- nach fixed-point theorem and numerical discretization for nonlinear volterra-fredholm equations *Journal of Computational and Applied Mathematics*, 2022, pp. 115019.

[46] C. Nwaigwe, Solvability and approximation of nonlinear func- tional mixed volterra-fredholm equation in banach space, *Journal of Integral Equations and Applications* 34(4), 2022, pp. 489–500.

[47] R. L. Burden, *Numerical analysis*, 2011.

[48] E. Sli and D. F. Mayers, *An introduction to numerical analysis*, Cambridge university press, 2003.

[49] C. Nwaigwe, *An unconditionally stable scheme for two- dimensional convection-diffusion-reaction equations*, 2022. www.researchgate.net/publication/357606287_An_Unconditionally_Stable_Scheme_for_Two-Dimensional_Convection-Diffusion-Reaction_Equations

[50] C. Nwaigwe and O. D. Makinde, Finite difference in- vestigation of a polluted non-isothermal non-newtonian porous media flow, *Diffusion Foundations* 26(4), 2019, pp. 145–156.

[51] H. O. Bakodah and M. A. Darwish, Solving hammer- stein type integral equation by new discrete adomian decomposition methods, *Mathematical Problems in Engineering* 2013, 2013.

[52] C. Nwaigwe, R. I. Ndu, and A. Weli, Wall motion effects on channel flow with temperature-dependent transport properties, *Applied Mathemat- ics* 9(3), 2019, pp. 162–168.

[53] A. Weli and C. Nwaigwe, Numerical analyses of channel flow with velocity-dependent suction and nonlinear heat source, *Journal of Interdisciplinary Mathematics* 23(5), 2020, pp. 987–1008.

[54] C. Nwaigwe, Analysis and application of a convergent difference scheme to nonlinear transport in a brinkman flow, *International Journal of Numerical Methods for Heat & Fluid Flow* 30(10), 2020, pp. 4453–4473.

[55] C. Nwaigwe, A. Weli, and O. D. Makinde, Computational analysis of porous channel flow with cross-diffusion, *Ameri- can Journal of Computational and Applied Mathematics* 9(5), 2019, pp. 119–132.

[56] C. Nwaigwe and A. Weli, Analysis of two finite differ- ence schemes for a channel flow problem, *Asian Research Journal of Mathematics* 15, 2019, pp. 1–14.

[57] H. Bellout, J. Neustupa and P. Penel, On the Navier-Stokes Equation with Boundary Conditions Based on Vorticity, *Math. Nachr.* 269–270, 2004, pp. 59–72.

[58] V. Girault and P.–A. Raviart, *Finite Element Methods for Navier–Stokes Equations,* Springer –Verlag, Berlin–Heidelberg–New York–Tokyo 1986

[59] E. Hopf, Über die Anfangswertaufgabe für die Hydrodynamischen Grundgleichungen, *Math. Nachr.* 4, 1951, pp. 213–231.

[60] T. Kato, Non–stationary flows of viscous and ideal fluids in $\mathbb{R}^3$, *J. Func. Anal.* 9, 1972, pp. 296–305.

[61] J. Serrin, On the interior regularity of weak solutions of the Navier–Stokes equations, *Arch. Rat. Mech. Anal.* 9, 1962, pp. 187–195.