**Global Analysis and Discrete Mathematics**
Volume 7, Issue 1, pp. 33–52
ISSN: 2476-5341 *Research Article*

# Solving Fredholm Integral Equations of the Second Kind Using an Improved Cuckoo Optimization Algorithm

**Hassan Dana Mazraeh · Maryam Kalantari · Seyed Hashem Tabasi · Alireza Afzal Aghaei · Zahra Kalantari · Farhad Fahimi**

Received: 15 September 2021 / Accepted: 6 November 2021

**Abstract** In this paper, we propose an improved cuckoo optimization algorithm (ICOA) to determine unknown function $u(x)$ in the Fredholm integral equations of the second kind. To show utility and capability of the ICOA, we solve some Fredholm integral equations of the second kind using the ICOA and Adomian decomposition method (ADM) and compare results each other. Also, by using the parallelization technique the running time of the algorithm was reduced significantly.

**Keywords** Improved cuckoo optimization algorithm · Fredholm integral equations of the second kind · Integral equations · Adomian decomposition method

H. Dana Mazraeh (Corresponding Author)
Department of Mathematics and Computer Science, Damghan University, Iran.
E-mail: dana@du.ac.ir

M. Kalantari
Department of Mathematics and Computer Science, Damghan University, Iran.
E-mail: maryamkalantari75@yahoo.com

S. H. Tabasi
Department of Mathematics and Computer Science, Damghan University, Iran.
E-mail: tabasi@du.ac.ir

A. Afzal Aghae
Department of Mathematics and Computer Science, Damghan University, Iran.
E-mail: alirezaafzalaghaei@gmail.com

Z. Kalantari
Department of Computer Sciences, Shahid Beheshti University, G.C. Tehran, Iran.
E-mail: zahrakalantari12@yahoo.com

F. Fahimi
Department of Computer Sciences, University of Tabriz, Tabriz, Iran.
E-mail: farhad.fahimi.cs@gmail.com

## 1 Introduction

An integral equation is an equation in which the unknown function $u(x)$ appears under an integral sign. Standard form of an integral equation is something like:

$$u(x) = f(x) + \lambda \int_{g(x)}^{h(x)} k(x,t)\ u(t)\, dt,$$

where $f(x)$ is a known function of $x$ and $k(x,t)$ is a known function of $x$, $t$ and $\lambda$ is a constant factor and $g(x)$ and $h(x)$ are integral bounds which can be constant or variable.

If the limits of integration are fixed, then this integral equation is called the Fredholm integral equation of the second kind (FIESK). Therefore, the form of this equation is as follows:

$$u(x) = f(x) + \lambda \int_{a}^{b} k(x,t)\ u(t)\, dt, \tag{1}$$

where $a$ and $b$ are constants.

In recent years, many methods have been reported to solve Fredholm integral equations such as ADM, variational iteration method, direct computation method, successive approximations method, haar wavelets, hybrid Taylor, block-pulse function method and etc [1–5]. All these methods try to find unknown function $u(x)$ which solves relevant integral equation. However these methods give a solution with high accuracy, investigation of the capability of evolutionary algorithms to solve integral equations might yield more accurate solutions with less execution time. In this paper, we propose an Improved Cuckoo optimization algorithm (COA) to solve FIESKs because of its high speed, accuracy, and its great ability to find global optima. In our proposed method, we assume that the function $u(x)$ is a polynomial and use the ICOA to find the coefficients of this polynomial. In order to increase the precision of our algorithm, a local search method called "Simulated Annealing"[7–10] has been added to the original cuckoo optimization algorithm as a step of its main loop. To improve the execution time of our method, we use the Horner's method to evaluate the polynomials in each step. Since, an evaluation of a definite integral is needed in each step, we use the Simpson's method to do that. Finally, to improve the whole execution time of our algorithm, we implement it in a parallel scheme.

The ADM was introduced by George Adomian [6]. This method has been applied successfully on wide class of equations and problems. Such as: heat conduction equations, wave equations, inverse parabolic problems, inverse hyperbolic problems and integral equations [19–21, 1, 4]. To solve an integral equation by the ADM, unknown $u(x)$ is considered as a series. The components of this series is obtained from a recursive substitution and form a polynomial. Therefore, this method is a good choice for comparison with our proposed method. Also, many authors have compared the ADM with their methods. In the [22], the ADM and the Runge Kutta method have been compared each

other for solutions of the Stefan problem. Bildik and Inc have done a research on a comparison between the Adomian decomposition and Tau methods in the[23]. And the comparison of the ADM and the variational iteration method in solving the moving boundary problem has been investigated in the [24].

The structure of this paper is organized as follows: In the Section 2, we explain the ADM. Section 3 is dedicated to explain the original Cuckoo optimization algorithm. In the Section 4, we present our improved cuckoo optimization algorithm. The Horner's method is presented in the Section 5. In the Section 6 we explain how our ICOA solves the Fredholm integral equation of the second kind. Section 7 discusses about the parallelization of the cost function of the ICOA. Also, in the Section 8, we discuss about a criterion for calculating the accuracy of our algorithms. Section 9 presents some numerical results of the implementation of our algorithm for some examples. Finally, we have a conclusion in the Section 10.

## 2 Adomian decomposition method

The ADM is a semi-analytical method to solve wide class of equations. To solve the second kind of the Fredholm integral equations by using this method, unknown function $u(x)$ is considered as a infinite series and components of this series is obtained as follows:

$$u(x) = \sum_{n=0}^{\infty} u_n(x) = u_0(x) + u_1(x) + \cdots + u_n(x) + \cdots , \tag{2}$$

where the components $u_n(x), n \geq 0$ will be determined recurrently. Therefore, substitution of (2) into equation (1) gives following equation:

$$\sum_{n=0}^{\infty} u_n(x) = f(x) + \lambda \int_a^b k(x,t) \left( \sum_{n=0}^{\infty} u_n(t) \right) dt, \tag{3}$$

or equivalently

$$u_0(x)+\cdots+u_n(x)+\cdots = f(x)+\lambda \int_a^b k(x,t)\, u_0(t)\, dt+\lambda \int_a^b k(x,t)\, u_1(t)\, dt+\cdots , \tag{4}$$

therefore, we can obtain an iterative relation to approximate unknown function $u(x)$ as follows:

$$\begin{aligned} u_0(x) &= f(x), \\ u_1(x) &= \lambda \int_a^b k(x,t)\ u_0(t)\, dt, \\ u_2(x) &= \lambda \int_a^b k(x,t)\ u_1(t)\, dt, \\ &\vdots \\ u_k(x) &= \lambda \int_a^b k(x,t)\ u_{k-1}(t)\, dt, \end{aligned} \tag{5}$$

where, as $k$ approaches infinity, $\sum_{n=0}^{\infty} u_n(x)$ approaches $u(x)$.

## 3 Original Cuckoo Optimization Algorithm

Cuckoo optimization algorithm has introduced by Ramin Rajabioun [11] in 2011. This optimization algorithm is inspired from the life of a bird family that called cuckoo. Special lifestyle of these birds and their characteristics in egg laying and breeding has been the basic motivation for development of this new evolutionary optimization algorithm.

COA starts with an initial population which contains mature cuckoos and their eggs. Each mature cuckoo has own egg laying radius (ELR) and it only can lay eggs in its ELR on host bird habitat. Then, some eggs that are less similar to host bird egg will demise.

Survived cuckoo population will migrate to a better place. To migrate, in the first cuckoos based on their position in environment are divided into some clusters and finally all cuckoos will migrate toward the best cuckoo in the best cluster with a $\lambda$ percent of way and with an deviation of $\alpha$ radians. When all of positions of cuckoo became very close, algorithm finishes. Being close means that this environment has maximum food sources and fewer eggs will be destroyed.

**COA pseudo code:**

(0) Start.
(1) Initialize cuckoos.
(2) If cuckoos are not very close to each other.
(3) Calculate cost of each cuckoo.
(4) Assign some eggs between $\max_{num}$ of eggs and $\min_{num}$ of eggs to each cuckoo.
(5) Calculate ELR for each cuckoo and determine the position of each egg of cuckoo according to parent cuckoo.
(6) If number of current cuckoos in population is greater than $\max_{num}$ of cuckoos, then demise weaker cuckoos.
(7) Clustering of the cuckoos on k clusters.
(8) Find the best cuckoo in the best cluster.
(9) Migrate cuckoos toward the best cuckoo with some deviation.
(10) Go to step (2).
(11) The best cuckoo is the best solution.
(12) End.

## 4 Improved Cuckoo Optimization Algorithm

However, the original COA is a capable algorithm to solve a wide range of problems, by changing some of its operators, according to the problem, we

can improve its total performance. In this paper, we added a local search algorithm called "Simulated Annealing" and chanced some operations of the original COA to improve it. The results showed that the modifications applied on the original COA, improved the total performance of the algorithm. The changes are as follows [13–15]:

(1) Adaptive ELR: Since, at the higher iterations, the approximated solutions must be changed less than the previous iterations, the size of the ELR must be reduced. This means, the eggs are not laid far away from their parent at the higher iterations. In this paper, the size of the ELR is reduced when the number of iterations is increased.
(2) Adaptive number of eggs: In the original COA, the number of eggs of each cuckoo is determined randomly, between the maximum and minimum number of eggs. In this scheme, it is possible that a weak cuckoo may lay more eggs than a strong one. This may reduce the speed of convergence. In this paper, we set the number of eggs of each cuckoo according to its cost value as follows:

$$eggNum_i = minEggNum + \left\lfloor (fitness_i - fitness_{min}) \times \frac{maxEggNum - minEggNum}{fitness_{max} - fitness_{min}} \right\rfloor,$$

where,

$$fitness_{min} = \text{The minimum fitness in the current population},$$
$$fitness_{max} = \text{The maximum fitness in the current population},$$
$$fitness_i = \text{the fitness value of the current cuckoo}.$$

Results show that, this improvement increases the speed of convergence at these kind of problems.
(3) Repair the malformed cuckoos: During the steps of the COA, some elements of the habitation vector of each cuckoos may exceed the search range of the problem. In this case, these elements are returned to the search range of the problem as much as the exceeded size. For example, if the search range is $[-M, M]$, an element with value of $M + \alpha$ will be $M - \alpha$ after the repair step.
(4) Apply a local search on the best solutions: In each iteration, to improve the best solutions, we apply the Simulating Annealing algorithm on the best cuckoo of each cluster. When the best cuckoo at each cluster get improved, other cuckoos are affected from this improvement due to the Migration step.

The pseudo code of the ICOA is as follows:

(1) Start.
(2) Initialization
(3) Calculate the cost of each cuckoo.

(4) Assign some eggs between $\max_{num}$ and $\min_{num}$ of eggs to each cuckoo.
(5) Calculate the ELR for each cuckoo and determine the position of each egg of cuckoos according to the parent cuckoo.
(6) If some elements of the habitation vector of a cuckoo exceeds from the predefined range then repair those elements.
(7) If number of current cuckoos in the population is greater than $\max_{num}$, then demise some weaker cuckoos.
(8) Determine cuckoo societies using $k$-maens clustering and find the best cuckoo in each cluster.
(9) Apply a local search algorithm on the best cuckoo.
(10) Move all cuckoos of each cluster toward to the best cuckoo in that cluster.
(11) Determine egg laying radius for each cuckoo.
(12) If the termination criteria is not satisfied then go to 3
(13) The best cuckoo is the best solution.
(14) End.

Fig. 1 presents flowchart of the ICOA.

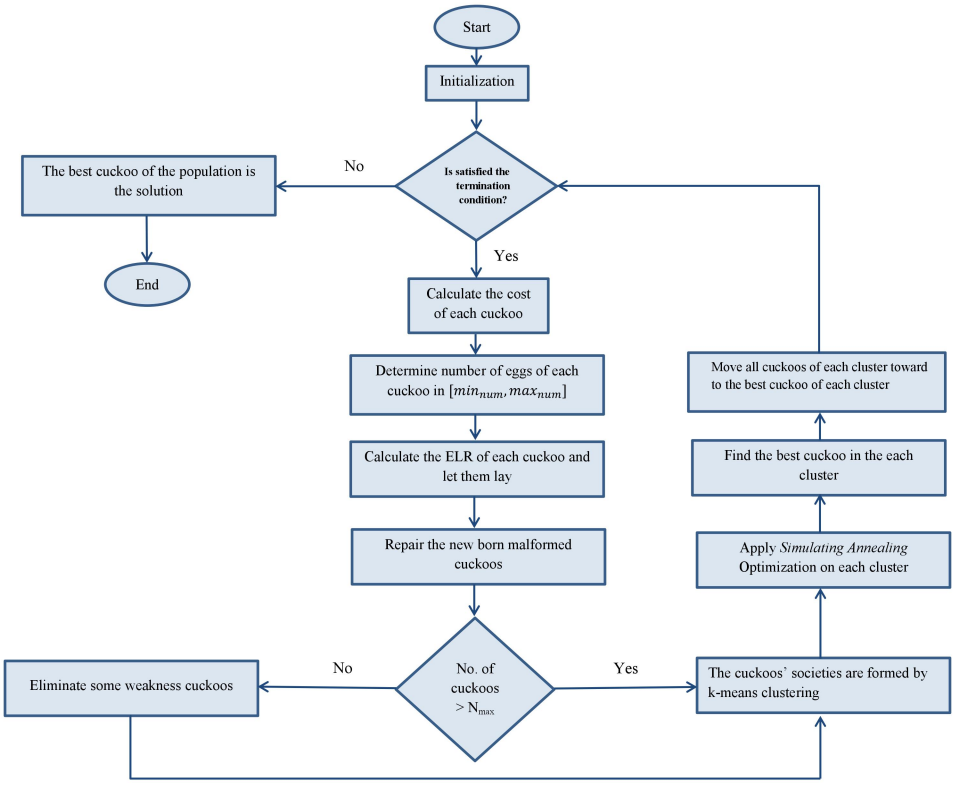


Fig. 1: flowchart of the improved cuckoo optimization algorithm.

## 5 The horner method

Whenever a cuckoo is evaluated, a relevant polynomial which is considered as a possible solution of problem, should be evaluated. Therefore, using an optimized method to evaluate the risen polynomials, could have significant effect on the execution time of the algorithm. One of the best methods to do this, is Horner's method. This method, requires just $n$ multiplications and $n$ additions to evaluate a $n$th-degree polynomial at $x = \alpha$. In fact, less computation in this method has two advantages. The first one is the reducing of rounding error and as it mentioned, the second one is an improvement in the execution time. The horner's method is defined as follows [16].
Suppose the polynomial $P(x)$ is as follows:

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0.$$

The main purpose is to evaluate $P(x)$. To do this, we put $a_n = b_n$ and for $k = n-1, n-2, \ldots, 0$, we calculate $b_k$ from the following equation

$$b_k = a_k + b_{k+1}x_0,$$

then

$$P(x_0) = b_0,$$

and $P(x)$ can be written as

$$P(x) = (b_n x^{n-1} + b_{n-1}x^{n-2} + \cdots + b_2 x + b_1)(x - x_0) + b_0.$$

## 6 Solving the Fredholm integral equations of the second kind using ICOA

In our method, each cuckoo's habitat represents a polynomial which its elements are coefficients of that polynomial. For a cuckoo's habitat such as follows:

$$\text{cuckoo habitat} = [g_0, g_1, g_2, \ldots, g_n],$$

we have the following polynomial:

$$u(x) = g_0 + g_1 x + \cdots + g_n x^n.$$

By substituting this polynomial in a FIESK, we have following equation

$$(g_0 + g_1 x + \cdots + g_n x^n) = f(x) + \int_a^b k(x,t)(g_0 + g_1 x + \cdots + g_n x^n)dt.$$

We can now calculate the right side using Simpson's method. Also, the equation above can be rewritten as follows:

$$h(x) = g(x),$$

where

$$h(x) = (g_0 + g_1 x + \cdots + g_n x^n) - f(x),$$
$$g(x) = \int_a^b k(x,t)(g_0 + g_1 x + \cdots + g_n x^n)dt.$$

Whenever the approximated $u(x)$ approaches to the exact $u(x)$, value of the $|h(x) - g(x)|$ approaches to zero. Therefore, we can define the cost function as follows:

$$\text{cost function} = \sum_{i=0}^{n} \|g(x_i) - h(x_i)\|.$$

In this scheme, the lower value of above equation implies the better approximation for unknown $u(x)$.

## 7 Parallelization of the cost function

Generally, the most time-consuming part of the evolutionary algorithms is the evaluation of the cost function. Therefore, in this paper, to optimize the running time of the algorithm, we calculate the cost function in a parallel scheme. In our algorithm, we are facing with a $M \times N$ matrix in the step of the calculating cost function. Where, $M$ is the number of cuckoos and $N$ is the degree of the polynomials. In fact,to implement this algorithm, a main "loop" is needed to calculate this matrix. Since, calculating of the cost functions can be done independently, we can easily use a parallel scheme.

Fig. 2 shows how a big program is divided to smaller parts and executed in a parallel scheme. We can implement the main loop of the algorithm using this scheme. In general, there are various ways to parallelize the main loop of a problem. In this paper, we use multi processing (openMP) method to do it [18].

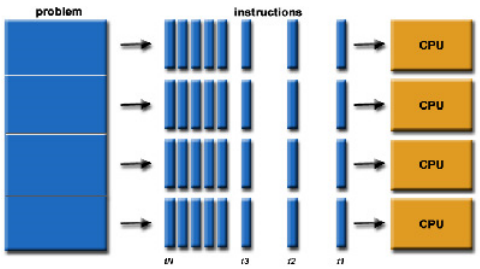


Fig. 2: Parallelization of a problem

## 8 Calculate the accuracy of the algorithm

8.1 Compare the results of our method with the results obtained by the least square method

For all examples in this paper, we have the exact solution of the integral equations. Therefore, we can compare the result with the exact solution to evaluate the efficiency of our proposed method.

8.2 Calculate the accuracy of the algorithm by using $S - Factor$:

$S - Factor$ is defined as follows [12]:

$$s = \left[ \frac{1}{N-1} \sum_{i=1}^{N} (\hat{q}_i - q_i)^2 \right]^{\frac{1}{2}}, \tag{6}$$

where $N$ is the number of estimated variables. $q_i, i = 1, 2, \ldots, N$ are the values of the result function at the point $x_i, i = 1, 2, \ldots, N$ and $\hat{q}_i, i = 1, 2, \ldots, N$ are the exact values of function at the point $x_i, i = 1, 2, \ldots, N$. In this paper, we use $S - Factor$ to evaluate the efficiency of a possible solution.

## 9 Numerical results

In this section, to have a convergence study for the our proposed method, some Fredholm integral equations of the second kind have been considered. All equations have been taken from [4].

*Example 1*

$$u(x) = 1 - \frac{19}{15}x^2 + \int_{-1}^{1} (xt + x^2t^2)u(t)\,dt.$$

*Example 2*

$$u(x) = 1 + \int_{0}^{\frac{\pi}{4}} (\frac{1}{2}(\sec^2(x)))u(t)\,dt.$$

*Example 3*

$$u(x) = -x^4 + \int_{-1}^{1} (xt^2 - x^2t)u(t)\,dt.$$

Table 1: Input table for Example 1.

| Parameters of the improved cuckoo optimization algorithm | |
|---|---|
| Radius coeff | 3.4975 |
| Minimum number of eggs | 2 |
| Maximum number of eggs | 10 |
| Uper limit of the search interval | 2 |
| Lower limit of the search interval | $-2$ |
| The number of iterations | 200 |
| The number of clusters | 5 |
| The number of variables | 5 |
| Maximum number of cuckoos | 50 |
| The radius of egg laying | 0.314606 |
| The radius of motion | 4 |
| The number of discrete points for calculation of the cost function | 20 |
| Parameters of the integral equation | |
| $f(x)$ | $1-\frac{19}{15}x^2$ |
| $\lambda k(x,t)$ | $xt+x^2t^2$ |
| Parameters of the simpson method | |
| Beginning of the integration interval | $-1$ |
| The end of the integration interval | 1 |
| The number of discrete points in the simpson method | 12 |
| Parameters of the simulated annealing | |
| Minimum temperature | $4.5\times10^{-7}$ |
| Maximum temperature | 0.00095 |
| The number of iterations | 2300000 |

Table 2: The results for Example 1.

| Results of the improved cuckoo optimization algorithm | |
|---|---|
| Execution time | |
| Serial time | 7 $min$ |
| Parallel time | 5 $min$ |
| The cost of the final cuckoo | $3.15845 \times 10^{-6}$ |
| The cost of the best result | 0.001516 |
| Difference | $-0.001512$ |
| The approximation polynomial of the best cuckoo | $(-9.99376 \times 10^{-7})x^4 + (4.24538 \times 10^{-7})x^3 - (1.00034)x^2 - (8.41502 \times 10^{-8})x + (0.99999)$ |
| The result of the least square method (the best result) | $(-1.38861 \times 10^{-15})x^4 + (2.28090 \times 10^{-16})x^3 - (0.99999)x^2 + (1.57926 \times 10^{-17})x + (0.99999)$ |
| Difference | $[9.99376 \times 10^{-7}, 4.24538 \times 10^{-7}, 0.00034, 8.41502 \times 10^{-8}, 1.68604 \times 10^{-7}]$ |
| The S-Factor function for the best polynomial obtained by least square method | $4.20903 \times 10^{-16}$ |
| The S-Factor function for the best polynomial obtained by improved cuckoo optimization algorithm | 0.00016 |
| Difference | 0.00016 |
| Compare the exact function with the polynomial derived from the least square method and ICOA method in 60 points | |
| The greatest difference between the real answer and the approximate answer of the improved cuckoo optimization algorithm | 0.00034 |
| The least difference between the real answer and the approximate answer of the improved cuckoo optimization algorithm | $2.65587 \times 10^{-7}$ |
| The greatest difference between the real answer and the approximate answer of the least square method | $9.16045 \times 10^{-13}$ |
| The least difference between the real answer and the approximate answer of the least square method | $5.55111 \times 10^{-16}$ |

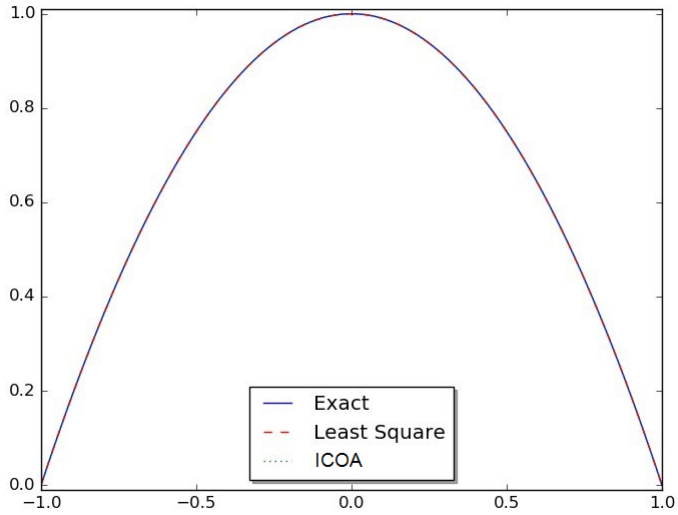


Fig. 3: Comparison study for Example 1.

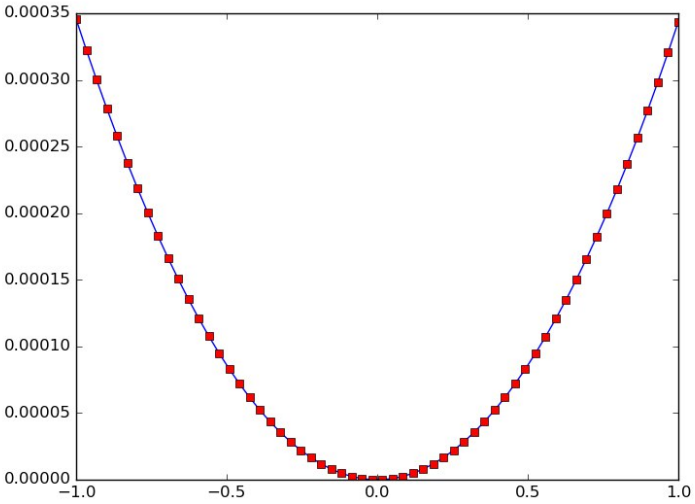


Fig. 4: Error study for Example 1.

Table 3: Input table for Example 2.

| Parameters of the improved cuckoo optimization algorithm | |
|---|---|
| Radius coeff | 6 |
| Minimum number of eggs | 2 |
| Maximum number of eggs | 10 |
| Uper limit of the search interval | 3 |
| Lower limit of the search interval | $-3$ |
| The number of iterations | 500 |
| The number of clusters | 5 |
| The number of variables | 7 |
| Maximum number of cuckoos | 200 |
| The radius of egg laying | 0.1746031746031746 |
| The radius of motion | 5.499 |
| The number of discrete points for calculation of the cost function | 50 |
| Parameters of the integral equation | |
| $f(x)$ | 1 |
| $\lambda k(x,t)$ | $\frac{1}{2}(\sec^2(x))$ |
| Parameters of the simpson method | |
| Beginning of the integration interval | 0 |
| The end of the integration interval | $\frac{\pi}{4}$ |
| The number of discrete points in the simpson method | 42 |
| Parameters of the simulated annealing | |
| Minimum temperature | $1 \times 10^{-43}$ |
| Maximum temperature | 0.00049 |
| The number of iterations | 5400000 |

Table 4: The results for Example 2.

| Results of the improved cuckoo optimization algorithm | |
|---|---|
| Execution time | |
| Serial time | 80 $min$ |
| Parallel time | 8 $min$ |
| The cost of the final cuckoo | 0.0091 |
| The cost of the best result | 0.127387 |
| Difference | $-0.00363857$ |
| The approximation polynomial of the best cuckoo | $(-0.438314)x^6 + (2.491752)x^5 - (1.881152)x^4 + (0.95981)x^3 + (0.626791)x^2 + (0.008157)x + (1.785388)$ |
| The result of the least square method (the best result) | $(1.997413)x^6 + (2.819566)x^5 - (2.405886)x^4 + (0.605745)x^3 + (0.876960)x^2 + (0.005490)x + (1.785070)$ |
| Difference | $[2.435727, 5.311319, 4.287039, 1.565558, 0.250169, 0.013647, 0.000318]$ |
| The S-Factor function for the best polynomial obtained by least square method | $3.943776 \times 10^{-5}$ |
| The S-Factor function for the best polynomial obtained by improved cuckoo optimization algorithm | 0.000562 |
| Difference | 0.000522 |
| Compare the actual function of answer with the polynomial derived from the least square method and ICOA method in 60 points | |
| The greatest difference between the real answer and the approximate answer of the improved cuckoo optimization algorithm | 0.001092 |
| The least difference between the real answer and the approximate answer of the improved cuckoo optimization algorithm | 0.000230 |
| The greatest difference between the real answer and the approximate answer of the least square method | 0.000104 |
| The least difference between the real answer and the approximate answer of the least square method | $4.993500 \times 10^{-6}$ |

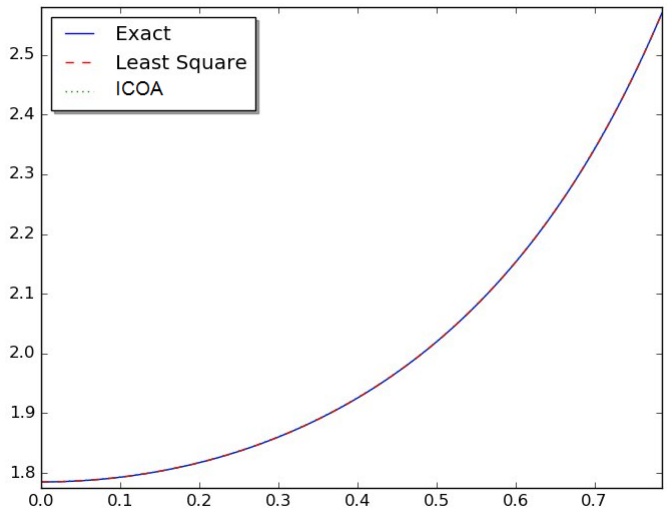


Fig. 5: Comparison study for Example 2.

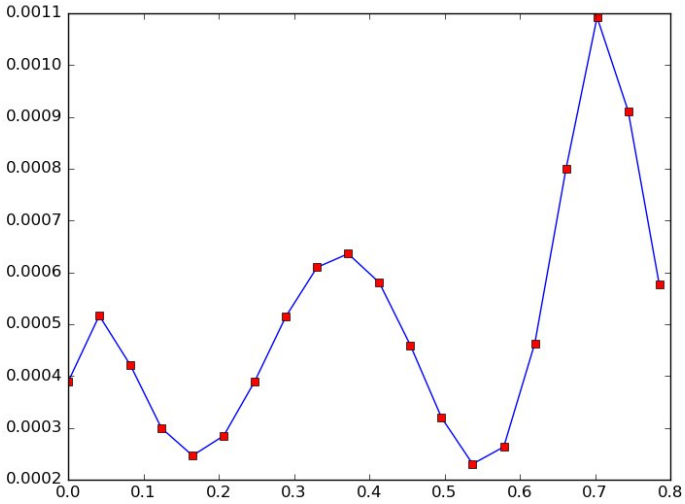


Fig. 6: Error study for Example 2.

Table 5: Input table for Example 3.

| Parameters of the improved cuckoo optimization algorithm | |
|---|---|
| Radius coeff | 4 |
| Minimum number of eggs | 2 |
| Maximum number of eggs | 10 |
| Uper limit of the search interval | 2 |
| Lower limit of the search interval | $-2$ |
| The number of iterations | 100 |
| The number of clusters | 5 |
| The number of variables | 5 |
| Maximum number of cuckoos | 250 |
| The radius of egg laying | 0.05758683729433272 |
| The radius of motion | 3.496 |
| The number of discrete points for calculation of the cost function | 20 |
| Parameters of the integral equation | |
| $f(x)$ | $-x^4$ |
| $\lambda k(x,t)$ | $xt^2 - x^2t$ |
| Parameters of the simpson method | |
| Beginning of the integration interval | $-1$ |
| The end of the integration interval | 1 |
| The number of discrete points in the simpson method | 12 |
| Parameters of the simulated annealing | |
| Minimum temperature | $8.7 \times 10^{-7}$ |
| Maximum temperature | 0.0021 |
| The number of iterations | 36000000 |

Table 6: The results for Example 3.

| Results of the improved cuckoo optimization algorithm | |
|---|---|
| Execution time | |
| Serial time | 69 $min$ |
| Parallel time | 8 $min$ |
| The cost of the final cuckoo | $2.98511 \times 10^{-6}$ |
| The cost of the best result | 0.010288 |
| Difference | $-0.010285$ |
| The approximation polynomial of the best cuckoo | $(-1.000001)x^4 - (1.420742 \times 10^{-7})x^3 + (0.150891)x^2 - (0.226335)x - (9.038432 \times 10^{-8})$ |
| The result of the least square method (the best result) | $(-1.000000)x^4 - (2.28090 \times 10^{-16})x^3 + (0.150375)x^2 - (0.225563)x + (2.220446 \times 10^{-8})$ |
| Difference | $[1.760353 \times 10^{-6}, 1.420742 \times 10^{-7}, 0.000515, 0.000771, 9.038432 \times 10^{-8}]$ |
| The S-Factor function for the best polynomial obtained by least square method | 0.482526 |
| The S-Factor function for the best polynomial obtained by improved cuckoo optimization algorithm | 0.482069 |
| Difference | $-0.000457$ |
| Compare the actual function of answer with the polynomial derived from the least square method and ICOA method in 60 points | |
| The greatest difference between the real answer and the approximate answer of the improved cuckoo optimization algorithm | 1.697974 |
| The least difference between the real answer and the approximate answer of the improved cuckoo optimization algorithm answer | $4.242014 \times 10^{-5}$ |
| The greatest difference between the real answer and the approximate answer of the least square method answer | 1.699260 |
| The least difference between the real answer and the approximate answer of the least square method | $3.124152 \times 10^{-6}$ |

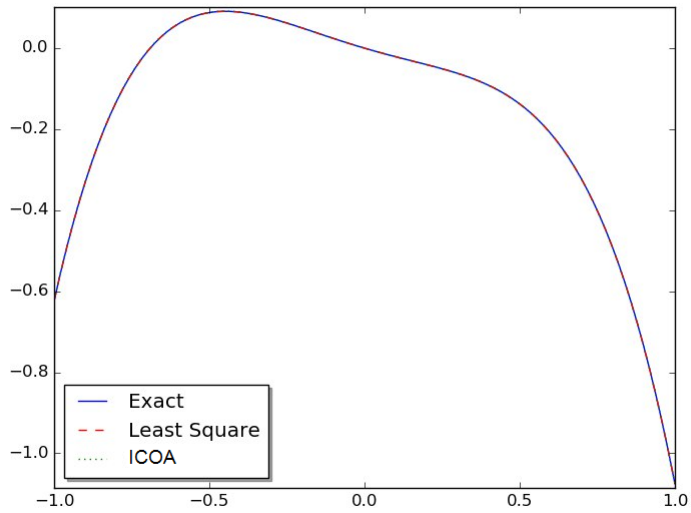


Fig. 7: Comparison study for Example 3.

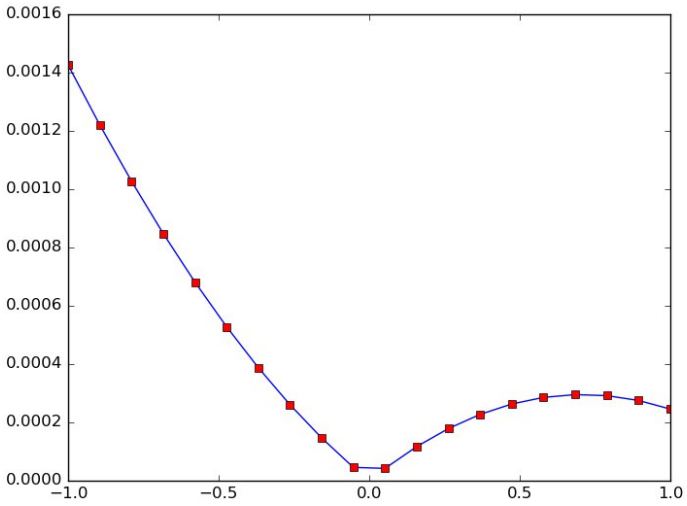


Fig. 8: Error study for Example 3.

Parameters for Examples 1, 2, and 3 are demonstrated in the Tables 1, 3, and 5, respectively. Also, the results for these examples are depicted in Tables 2, 4, and 6. It is evident from results that the ICOA is capable of obtaining a good approximation for the unknown function in the form of an polynomial by a certain number of iterations in a specified interval. Another important issue is that the execution time of parallel ICOA is much lower than the execution time of the ICOA. Furthermore, it is evident that the polynomials obtained by our proposed method are accurate and close to the exact solution.

## 10 Conclusions

In this paper an improved cuckoo optimization algorithm presented to solve a class of integral equations called Fredholm integral equations of the second kind. The experimental results showed that our presented method has a great capability to solve this class of equations with a high accuracy. Therefore, this method is a good choice to be used in real-world applications. Furthermore, we showed our method has also the capability of being implemented in a parallel scheme. The parallel implementation of our method showed a great improvement in the execution time.

## References

1. K. Atkinson, A survey of numerical methods for the solution of Fredholm integral equations of the second kind, Soc. for Industrial and Applied Mathematics, Philadelphia, (1976).
2. K. E. Atkinson, The numerical solution of integral equations of the second kind, Cambridge university press, (1996).
3. S. Rahbar, E. Hashemizadeh, A computational approach to the Fredholm integral equation of the second kind, In Proceedings of the World Congress on Engineering, 2, 933–937 (2008).
4. A. M. Wazwaz, Linear and nonlinear integral equations: methods and applications, Springer Heidelberg Dordrecht, (2011).
5. J. Y. Xiao, L. H. Wen, D. Zhang, Solving second kind Fredholm integral equations by periodic wavelet Galerkin method, Applied Mathematics and Computation, 175, 508–518 (2006).
6. G. Adomian, A review of the decomposition method and some recent results for nonlinear equation, Mathematical and Computer Modelling, 13, 17–42, (1990).
7. D. Bertsimas, J. Tsitsiklis, simulated annealing, Statistical science, 8, 10–15 (1993).
8. C. R. Hwang, simulated annealing: theory and applications, Acta Applicandae Mathematicae, 12, 108–111, (1988).
9. R. A. Rutenbar, simulated annealing algorithms: An overview, IEEE Circuits and Devices Magazine, 5, 19–26 (1989).
10. E. Aarts, J. Korst, Simulated annealing and Boltzmann machines, A stochastic approach to combinational optimization and neural computing, 1039–1044 (2003).
11. R. Rajabioun, cuckoo optimization algorithm, Applied soft computing, 11, 5508–5518 (2011).
12. R. Pourgholi, H. Dana, S. H. Tabasi, Solving an inverse heat conduction problem using genetic algorithm: Sequential and multi-core parallelization approach, Applied Mathematical Modelling, 38, 1948–1958 (2014).
13. H. Kahramanli, A modified cuckoo optimization algorithm for engineering optimization, International Journal of Future Computer and Communication, 1, 199, (2012).

14. M. Lashkari, M. Moattar, Improved COA with chaotic initialization and intelligent migration for data clustering, Journal of AI and Data Mining, 5, 293–305 (2017).
15. E. Valian, S. Tavakoli, S. Mohanna, A. Haghi, Improved cuckoo search for reliability optimization problems, Computers & Industrial Engineering, 64, 459–468 (2013).
16. T. X. He, P. J.-S. Shiue, A note on horner's method, Journal of concrete and Applicable Mathematics, 10, 53–64 (2012).
17. E. W. Weisstein, simpson's rule, https://mathworld. wolfram. com/ (2003).
18. R. Chandra, L. Dagum, D. Kohr, R. Menon, D. Maydan, J. McDonald, Parallel programming in OpenMP. Morgan kaufmann, (2001).
19. D. Lesnic, The Cauchy problem for the wave equation using the decomposition method, Applied Mathematics Letters, 15, 697 701, (2002).
20. S. Pamuk, An application for linear and nonlinear heat equations by Adomian s decomposition method. Applied Mathematics and Computation, 163, 89 96 (2005).
21. R. Pourgholi, A. Saeedi, A numerical method based on the Adomian decomposition method for identifying an unknown source in non-local initial-boundary value problems, IJMNO, 6, 185–197 (2015).
22. R. Grzymkowski, M. Pleszczynski, D. Slota, Comparing the Adomian decomposition method and the Runge Kutta method for solutions of the Stefan problem, International Journal of Computer Mathematics, 83, 409 417 (2006).
23. N. Bildik, M. Inc, A Comparison between Adomian Decomposition and Tau Methods, Hindawi Publishing Corporation Abstract and Applied Analysis, 2013, (2013).
24. E. Hetmaniok, D. Slota, R. Witula, A. Zielonka, Comparison of the Adomian decomposition method and the variational iteration method in solving the moving boundary problem, Computers & Mathematics with Applications, 61, 1931–1934. (2011).