The First International Conference on
Machine Learning and Knowledge Discovery (MLKD 2024)
Amirkabir University of Technology, December 18-19, 2024

A Hybrid Method Based on Least Squares Support Vector Regression and Orthonormal Bernoulli Polynomials for Solving Fredholm Integral Equations

Marzieh Pourbabaee*

Abstract

In this work, we propose a machine learning approach utilizing Least Squares Support Vector Regression (LS-SVR) to numerically solve Fredholm integral equations. The suggested approach utilizes a combination of LS-SVR alongside orthonormal Bernoulli polynomials, as well as Galerkin and collocation spectral techniques. An optimization problem is derived and converted into the solution of a system of algebraic equations. Finally, we present two numerical results that demonstrate the efficiency of the proposed method.

Keywords: Least squares support vector machines, Fredholm integral equation, and orthonormal Bernoulli polynomials.

1 Introduction

Integral equations have recently found applications in machine learning techniques. Many problems in physics and engineering can be modeled using integral equations. For example, they are used in the prediction of multiple crack propagations in elastic media [1], heat transfer problems [2], option pricing [3], and etc. Integro-differential equations are typically challenging to solve analytically, creating a need for efficient approximate solutions [4, 5]. Various approaches have been suggested for addressing integro-differential equations; nonetheless, many of these methods have constraints, such as unrealistic premises, linear approximations, slow convergence rates, and results that diverge. Among these methods are Adomain decomposition method[6], Runge-Kutta [7], Wavelet-Galerkin method[8], least squares [9] and Chebyshev polynomial [10].

In recent years, machine learning algorithms have gained significant popularity across various fields, such as pattern recognition, recommendation systems, healthcare, theorem proving, cybersecurity, and financial services. These methods aim to uncover the underlying structure of a problem by analyzing incoming data. The use of machine learning techniques has now expanded into both engineering and mathematical challenges. These techniques have been designed for the numerical simulation of differential and integral equations [11, 12, 13]. Each year, new techniques are introduced that surpass the current leading algorithms. Some of these represent minor improvements or combinations of existing methods, while others are entirely new developments that result in impressive advancements. They have also been utilized as a means to approximate the solutions of ordinary differential equations (ODEs), partial differential equations (PDEs), and integral equations (IEs). Support Vector Machines (SVMs) are a robust methodology for addressing pattern recognition and function estimation problems [14, 15]. SVM algorithms [16], introduced by Vapnik within the framework of statistical learning theory, map input data into a high-dimensional feature space using a feature map. These algorithms are capable of achieving a global optimum by solving a convex quadratic programming problem. SVMs are models designed for supervised learning tasks, such as classification and regression, introduced by Cortes et al. [17]. They include a notable innovation that emphasizes improving the model's generalization capabilities rather than solely focusing on minimizing empirical risk on training inputs. The least squares support vector machine (LS-SVM) is a modification of the traditional SVM formulation for machine learning tasks, originally proposed by Suykens and Vandewalle in 1999. The LS-SVM replaces the inequality constraints of the SVM's primal model with equality constraints. Additionally, the loss function associated with the slack variables is changed to a squared error loss function. These modifications allow the dual problem to be expressed as a system of linear equations, which can be more computationally efficient to solve than a quadratic programming problem in certain cases. To name a few, Mall et al. [18] presented Chebyshev neural networks for solving Lane–Emden type equations. Golbabai et al. [19] utilized radial basis function networks to address second-kind integral equations. A least-squares support vector regression scheme was developed by Parand et al. [12] for solving Fredholm integral equations. Lu et al. [20] addressed higher-order nonlinear ordinary differential equations using a method based on least-squares support vector machines. In addition, LS-SVM algorithms have been successfully applied to solving dif-

*Department of Applied Mathematics, Faculty of Mathematical Sciences, University of Kashan, Kashan 87317-53153, Iran, m.pourbabaee@kashanu.ac.ir

ferential equations [21, 22], differential algebraic equations [24, 23], and integral equations [25]. Ye et al. presented an orthogonal Chebyshev kernel for SVMs in 2006. Leake et al. compared the application of connection theory using LS-SVMs in 2019. Baymani et al. developed a new technique utilizing LS-SVMs to achieve analytical solutions for ODEs in 2016. Additionally, Chu et al. introduced an improved method for the numerical solution of LS-SVMs. In this paper, the least-squares support vector algorithm is utilized to develop a numerical algorithm for solving Fredholm integral equations using orthonormal Bernoulli polynomials. In this study, we examine the integral equation involving the unknown function $y(t)$ [12]:

$$\rho y(\mathbf{t}) = g(\mathbf{t}) + \mu \int_C k(\mathbf{t}, \mathbf{v}) \lambda(y(\mathbf{v})) d\mathbf{v}, \ (\mathbf{t}, \mathbf{v}) \in C \subset \mathbb{R}^n \tag{1}$$

where, $\mathbf{t} = (t_1, t_2, \ldots, t_n), \mathbf{v} = (v_1, v_2, \ldots, v_n)$, μ is a constant, referred to as the eigenvalue of the integral operator, C is a compact set and a square-integrable kernel of the integral equation denote by $k(\mathbf{t}, \mathbf{v}) \in L^2(C)$. If $\rho = 0$, it is referred to as an integral equation of the first kind; otherwise, it is classified as an integral equation of the second kind. Based on the function $\lambda(.)$, Equation (1) can be classified as either a linear or nonlinear integral equation. For simplicity, we express Equation (1) in operator form as follows:

$$L(y) = g, \tag{2}$$

where L defined by

$$L(y) = \rho y(t) - \mu \int_C k(\mathbf{t}, \mathbf{v}) \lambda(y(\mathbf{v})) d\mathbf{v}, \tag{3}$$

The main goal of this paper is to construct an efficient learning-based method using least squares support vector machines for the numerical simulation of integral equation (1). This method utilizes two different training approaches: Galerkin least squares support vector regression (GLS-SVR) and collocation least squares support vector regression (CLS-SVR). We employ orthonormal Bernoulli polynomials as the kernel of the network. The approach leads to an optimization problem that can be reduced to solving a system of algebraic equations, facilitating an efficient resolution of the integral equation.

2 Preliminaries and initial definitions

In this section, we compile essential results concerning orthogonal polynomials, numerical integration, and LS-SVR, which will be necessary for the formulation of the proposed method. These foundational concepts will underpin the development of our learning-based approach and ensure a thorough understanding of the mathematical tools used in the numerical simulation of the integral equation. We will first review the properties of orthogonal polynomials, focusing on their recursive relations and applications in approximation theory. Next, we will discuss numerical integration techniques, particularly emphasizing their relevance in the context of the proposed kernel-based methods. Finally, we will outline the principles of least squares support vector regression, illustrating how these principles can be applied in our framework for solving integral equations.

2.1 Bernoulli's Polynomial

The Bernoulli polynomials specifically defined on the interval $[0, 1]$ of degree n are defined by the following generating function [26]

$$\frac{t}{exp(t) - 1} = \sum_{n=0}^{\infty} B_n \frac{t^n}{n!},$$

where, B_n represents the n-th Bernoulli polynomial. These polynomials can also be expressed explicitly for non-negative integers n using the following recursive relation:

$$B_0(t) = 1, \quad B_n(t) = \sum_{j=0}^{n} \binom{n}{j} b_{n-j} t^n, \tag{4}$$

where $b_j, \ \ j = 0, 1, \ldots, n$, are Bernoulli's numbers. The first few Bernoulli polynomials for $n = 4$ are

$$\begin{aligned} B_0(t) &= 1, \\ B_1(t) &= t - \frac{1}{2}, \\ B_2(t) &= t^2 - t + \frac{1}{6}, \\ B_3(t) &= t^3 - \frac{3}{2}t^2 + \frac{1}{2}t, \\ B_4(t) &= t^4 - 2t^3 + t^2 - \frac{1}{30}. \end{aligned}$$

However, the name "Bernoulli Polynomials" was coined by J. L. Raabe in 1851. A thorough study of these polynomials for arbitrary values of their variable was first conducted by Leonhard Euler in 1755. In his book "Foundations of Differential Calculus," Euler demonstrated that Bernoulli polynomials satisfy a finite difference relation, establishing a key connection between these polynomials and finite difference calculus. Euler's work laid the groundwork for many subsequent developments in the theory and applications of Bernoulli polynomials. Their significance extends beyond pure mathematics into various fields, including numerical analysis, combinatorics, and number theory, where they are utilized for approximations and error analysis in numerical integration. The properties of Bernoulli's polynomials

$B_n(t)$ and Bernoulli's numbers b_n are fundamental in various areas of mathematics. Here are some key properties [27]

1. $B_n(1-t) = (-1)^n B_n(t),\ n \in \mathbb{Z}^+$,
2. $B_n'(t) = nB_{n-1}(t),\ n \in \mathbb{Z}^+$,
3. $B_n(t+1) - B_n(t) = nt^{n-1},\ n \geq 1$,
4. $\int_0^1 B_n(t)B_m(t)dt = (-1)^{n-1}\frac{m!n!}{(m+n)!}b_{m+n},\ m,n \geq 1$,
5. $\int_0^1 B_n(t)dt = 0,\ n \geq 1$,
6. $b_{2n+1} = 0,\ b_{2n} = B_{2n}(1)$,
7. $B_n(\frac{1}{2}) = (2^{1-n}-1)b_n$,
8. $b_n = \frac{-1}{1+n}\sum_{i=0}^{n-1}\begin{pmatrix} n+1 \\ i \end{pmatrix} b_i$.

2.2 Orthonormal Bernoulli polynomials

Bernoulli polynomials possess many beneficial characteristics; however, they do not exhibit orthogonality. In certain numerical methods, orthogonality is especially important. As a result, utilizing these polynomials is generally less suitable compared to orthogonal polynomials like Chebyshev and Legendre polynomials. To address this issue, the Gram-Schmidt orthonormalization procedure is applied to sets of Bernoulli polynomials of varying degrees. Specifically, this is applicable on the interval [0, 1] as noted in [28].

$$\begin{aligned} OB_0(t) &= 1, \\ OB_1(t) &= \sqrt{3}(2t-1), \\ OB_2(t) &= \sqrt{5}(6t^2-6t+1), \\ OB_3(t) &= \sqrt{7}(20t^3-30t^2+12t-1), \\ OB_4(t) &= 3(70t^4-140t^3+90t^2-20t+1). \end{aligned}$$

By examining the coefficients of this polynomial, we can identify a pattern and subsequently introduce the shifted Orthogonal Bernoulli Polynomials (OBPs). This leads us to the following definition.

Definition 1 *The OBPs on the interval* $[0,1]$ *are defined as [28, 29]*

$$OB_i(t) = \sqrt{2i+1}\sum_{k=0}^{i}(-1)^k \begin{pmatrix} i \\ k \end{pmatrix}\begin{pmatrix} 2i-k \\ i-k \end{pmatrix} t^{i-k}, \tag{5}$$

Consequently, these polynomials meet the orthogonality condition as outlined in [30].

$$\int_0^1 BP_i(t)BP_j(t)dt = \delta_{ij}, \quad i,j = 0,1,\ldots, \tag{6}$$

where δ_{ij} represents the Kronecker delta function.

Remark 1 *Several advantages of Bernoulli basis functions include:*

1. *As mentioned in [31], Bernoulli basis functions provide more accurate approximations of problem solutions with a fewer number of basis functions compared to some other basis functions.*
2. *The OBPs are simple basis functions, making the implementation of the Bernoulli operational matrices method straightforward.*
3. *According to [32], Bernoulli polynomials have fewer terms than shifted Legendre polynomials (SLP), and the coefficients of individual terms in Bernoulli polynomials are smaller than those in the SLP.*

Let $G = [0,1]$ and $g(t)$ be a square-integrable function defined over G. Then, $g(t)$ may be expressed in an OBP series as follows [29]

$$g(t) = \sum_{j=0}^{\infty} d_j OB_j(t), \tag{7}$$

where

$$d_j = \int_0^1 g(t)OB_j(t), \quad j = 0,1,\ldots. \tag{8}$$

Also, we can consider the following truncated series for $g(t)$ as:

$$g(t) = \sum_{j=0}^{M} d_j OB_j(t). \tag{9}$$

For a given positive integer V, let $x_j,\ j = 1,\ldots,V$ be the set of V distinct roots of Legendre polynomials of degree V. The V-point Legendre-Gauss quadrature rule can be employed to estimate the integral of a function $g(t)$ over the interval (β_1, β_2) as follows:

$$\int_{\beta_1}^{\beta_2} g(t)dt \simeq \frac{\beta_2-\beta_1}{2}\sum_{i=1}^{V} w_i g\left(\frac{\beta_2-\beta_1}{2}x_i + \frac{\beta_2+\beta_1}{2}\right), \tag{10}$$

Here w_i are the weights corresponding to each root x_i of the Legendre polynomial $L_V(t)$, and the transformation adjusts the standard quadrature rule (which typically operates over the interval $[-1,1]$) to the specified interval (β_1, β_2). Also, we can obtain w_i as following [33]

$$w_i = \frac{\beta_2-\beta_1}{(1-x_i^2)(L_V'(x_i))^2} \tag{11}$$

This method is particularly effective for approximating integrals of smooth functions.

2.3 LS-SVR Formulation

LS-SVR is a regression technique based on the SVM framework that seeks to model the relationship between input features and target outputs. It is assumed that the function representing the relationship between input points and output points can be expressed in the following way:

$$y(t) = \sum_{p=0}^{M} r_p OB_p(t) + \zeta = R^T \Psi(t) + \zeta, \tag{12}$$

In this context, $r_p, \quad p = 0, \ldots, M$ and ζ are the model parameters that need to be identified, while $OB_p, \quad p = 0, \ldots, M$ represents the nonlinear feature mapping that transforms the input space into a higher-dimensional feature space. Next, the goal is to find the optimal solution in that space by reducing the difference between the model outputs and the actual measurements [34]. To achieve this, the LS-SVM model in its primal form is expressed as the following optimization problem [35, 36]

$$\min_{R,e} J(R, e) = \frac{1}{2} R^T R + \frac{1}{2} \rho e^T e, \tag{13}$$

subject to

$$y_j = R^T \Psi(x_j) + e_j + \zeta, \quad j = 0, 1, \ldots, K, \tag{14}$$

Here, ρ represents a positive regularization parameter, and e_j denotes the error associated with the jth input data. The first term serves as a regularization component, whereas the second term focuses on minimizing training errors. The optimization problem with equality constraints (13) can be addressed using the method of Lagrange multipliers.

$$\mathcal{L} = \frac{1}{2} R^T R + \frac{1}{2} \rho e^T e - \sum_{j=0}^{K} \lambda_j \left(R^T \Psi(x_j) + e_j + \zeta - y_j \right) \tag{15}$$

In the LS-SVM formulation, λ_j (where $j = 1, \ldots, K$) are the Lagrange multipliers, which may take on positive or negative values. The problem is solved by utilizing Lagrange multipliers and incorporating the KKT conditions.

3 Least squares support vector regression for Fredholm integral equations

In this subsection, we introduce OB-LS-SVR, a novel hybrid method designed to solve Equation (1). Initially, we approximate $y(t)$ using OB polynomials as follows [12]

$$y(t) \simeq \tilde{y}(t) = \sum_{j=0}^{M} d_j OB_j(t), \tag{16}$$

where $d_j, \ j = 0, \ldots, M$ are unknown. The residual function (Res_M) for the mentioned problem can be reformulated as follows:

$$Res_M = L(\tilde{y}(t)) - g(t). \tag{17}$$

By utilizing the LS-SVR method, we derive the unknown coefficients. This leads us to the subsequent constrained optimization problems. The technique presented here serves as an interpolation that balances the collocation method and the least squares method through Tikhonov regularization.

$$\begin{aligned} \min_{R,e} \quad & J(R, e) \quad = \frac{1}{2} R^T R + \frac{1}{2} \rho e^T e, \\ & \text{s. t.} \quad < L(\tilde{y}(t)) - g(t), \Theta_j >= e_j, \ j = 0, \ldots, K. \end{aligned} \tag{18}$$

In the provided equation, the notation $< . >$ signifies the inner product of two functions, K indicates the number of training points, and $\Theta_j, \ j = 0, \ldots, K$ refers to a collection of test functions situated within the test space for an Equation (1). Now, We will examine the Lagrangian function related to the constrained problem (21) as outlined below:

$$L = \frac{1}{2} R^T R + \frac{1}{2} \rho e^T e - \sum_{j=0}^{K} \lambda_j \left(< L(\tilde{y}(t)) - g(t), \Theta_j > - e_j \right) \tag{19}$$

The necessary conditions for the minimum of Equation (22) are provided by:

$$\frac{\partial L}{\partial \lambda_j} = 0, \quad \frac{\partial L}{\partial e_j} = 0, \quad j = 0, \ldots, K, \tag{20}$$

and

$$\frac{\partial L}{\partial R_j} = 0, \quad j = 0, \ldots, M.$$

By solving the above equations, the unknown coefficients can be obtained. Functions Θ_j, known as test functions in weighted residual methods (WRMs), are particular types of functions that include special cases such as the Dirac delta function, basis functions, and others. These functions play a crucial role in formulating the weak form of differential equations by allowing for the construction of residuals that can be minimized or solved in various contexts.

Remark 2 *By utilizing the Dirac delta function as $\Theta_j = \delta(t - t_j)$, which has the property $< y(t), \delta(t - t_j) >= y(t_j)$, we can construct a collocation model for LS-SVR (CLS-SVR). This approach allows us to enforce that the solution $y(t)$ accurately fits the training data at specific points t_j effectively transforming the problem into one of minimizing the residuals at those collocation points. This method is particularly useful in ensuring that the regression captures the behavior of the underlying function at the selected points in the domain.*

So, we have

$$\begin{aligned} \min_{R,e} \quad & J(R,e) = \frac{1}{2}R^T R + \frac{1}{2}\rho e^T e, \\ \text{s. t.} \quad & L(\tilde{y}(t_j)) - g(t_j), \Theta_j = e_j,\ j = 0, \ldots, K. \end{aligned} \tag{21}$$

Remark 3 *If we consider $\Theta_j = P_j$ we get the optimization problem name as Galerkin LS-SVR (GLS-SVR). The optimization problem for GLS-SVR involves formulating a method that combines the principles of the Galerkin approach with least squares support vector regression. In GLS-SVR, test functions are typically chosen to be basis functions, and the residuals are minimized in a weighted least squares sense over the entire domain.*

So, we have

$$\begin{aligned} \min_{R,e} \quad & J(R,e) = \frac{1}{2}R^T R + \frac{1}{2}\rho e^T e, \\ \text{s. t.} \quad & < L(\tilde{y}(t)) - g(t), P_j >= e_j,\ j = 0, \ldots, K. \end{aligned} \tag{22}$$

4 Numerical examples

In this section, we conduct a series of numerical experiments to demonstrate the reliability and effectiveness of the improved LS-SVR algorithms. These experiments will assess the algorithms' performance across a range of tasks and highlight their advantages over standard LS-SVR methods. We executed the suggested method using Maple programming on a personal computer equipped with a 2.20 GHz Core i7 processor and 8 GB of RAM.

example 1 *We consider the following linear Fredholm integral equation as the first problem [12]*

$$y(t) - \int_0^1 \frac{\cosh(t+\tau+\pi)}{\sqrt{t^2+\tau^2+1}} y(\tau)d\tau = g(t). \tag{23}$$

Now we have a specific function $g(t)$ along with the exact solution $y(t) = \dfrac{1}{\sqrt{t^2+1}}$.

Table 1 denote the maximum absolute error with $\rho = 10^{20}$ and $M = 4, 7$ for CLS-SVR and $M = 5, 8$ for GLS-SVR methods. Also, the absolute error for $\rho = 10^{20}, M = 9$ for CLS-SVR (above) and GLS-SVR (below) display in Figure 1. Moreover, the report of CPU time for CLS-SVR and GLS-SVR methods with various values of M for Example 1 displayed in Table 2.

example 2 *Now, we consider the second kind linear Fredholm integral equation as [12]:*

$$y(t) - \int_0^1 (t+\tau)^2 y(\tau)d\tau = g(t), \tag{24}$$

we have a specific function $g(t)$ along with the exact solution $y(t) = \sin(10t)$.

Table 1: Maximum absolute error with $\rho = 10^{20}$, for Example 1.

t	CLS-SVR		GLS-SVR	
	$M=4$	$M=7$	$M=5$	$M=8$
0.1	2.48×10^{-4}	1.20×10^{-7}	4.94×10^{-5}	1.92×10^{-7}
0.2	8.85×10^{-5}	1.47×10^{-6}	2.27×10^{-5}	3.32×10^{-8}
0.3	1.43×10^{-4}	1.75×10^{-6}	3.96×10^{-5}	2.21×10^{-7}
0.4	1.47×10^{-4}	2.43×10^{-7}	1.09×10^{-5}	2.80×10^{-7}
0.5	4.83×10^{-9}	1.55×10^{-6}	4.22×10^{-5}	5.02×10^{-9}
0.6	1.16×10^{-4}	2.05×10^{-7}	1.06×10^{-5}	3.04×10^{-7}
0.7	8.94×10^{-5}	1.26×10^{-6}	3.76×10^{-5}	2.27×10^{-7}
0.8	4.38×10^{-5}	9.07×10^{-7}	2.13×10^{-5}	6.63×10^{-8}
0.9	9.92×10^{-5}	6.44×10^{-8}	4.67×10^{-5}	2.20×10^{-7}

Table 2: CPU time for CLS-SVR and GLS-SVR methods with various values of M for Example 1.

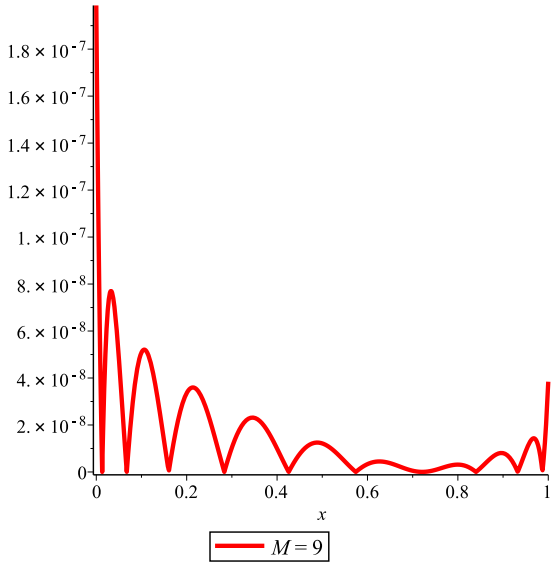
	$M=3$	$M=4$	$M=7$	$M=9$
CLS-SVR	0.312	0.343	0.405	0.452
	$M=5$	$M=6$	$M=7$	$M=8$
GLS-SVR	0.156	0.187	0.234	0.265

Table 3: Maximum absolute error with $\rho = 10^{20}$ and $M = 8, 10$ for Example 2.

t	CLS-SVR		GLS-SVR	
	$M=8$	$M=10$	$M=8$	$M=10$
0.1	8.17×10^{-4}	3.77×10^{-5}	8.16×10^{-4}	3.68×10^{-5}
0.2	5.13×10^{-5}	1.33×10^{-5}	5.29×10^{-5}	1.31×10^{-5}
0.3	4.61×10^{-4}	6.22×10^{-5}	4.59×10^{-4}	6.32×10^{-5}
0.4	1.51×10^{-3}	7.97×10^{-5}	1.51×10^{-3}	7.82×10^{-5}
0.5	1.35×10^{-7}	3.18×10^{-7}	2.82×10^{-6}	1.90×10^{-6}
0.6	3.28×10^{-3}	1.48×10^{-4}	3.28×10^{-3}	1.50×10^{-4}
0.7	3.11×10^{-3}	2.52×10^{-4}	3.11×10^{-3}	2.50×10^{-4}
0.8	8.80×10^{-4}	3.77×10^{-4}	8.84×10^{-4}	2.80×10^{-4}
0.9	3.96×10^{-3}	1.41×10^{-3}	3.95×10^{-3}	3.77×10^{-4}

Table 4: CPU time for CLS-SVR and GLS-SVR methods with various values of M for Example 2.

	$M=7$	$M=8$	$M=9$	$M=10$
CLS-SVR	0.187	0.203	0.218	0.234
GLS-SVR	0.202	0.234	0.281	0.297



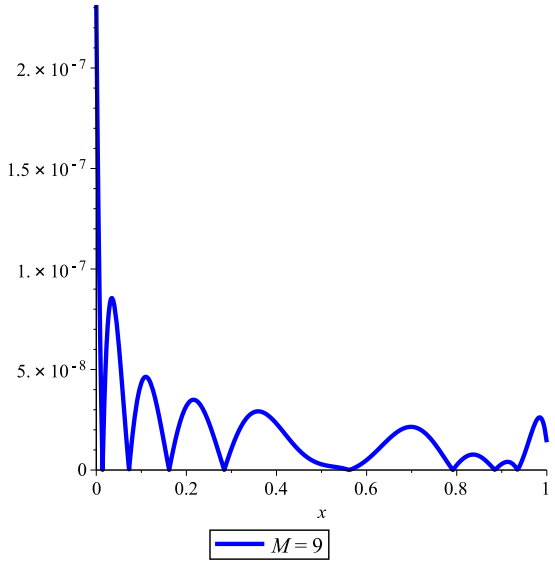
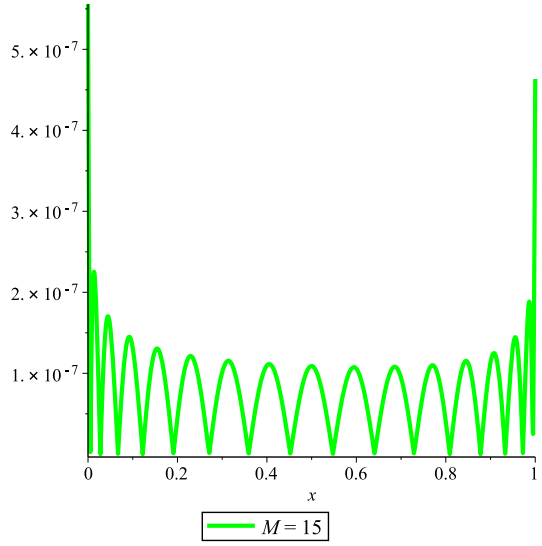


Figure 1: Absolute errors for CLS-SVR (above) and GLS-SVR (below) methods at $M = 9$ and $\rho = 10^{20}$ for Example 1.



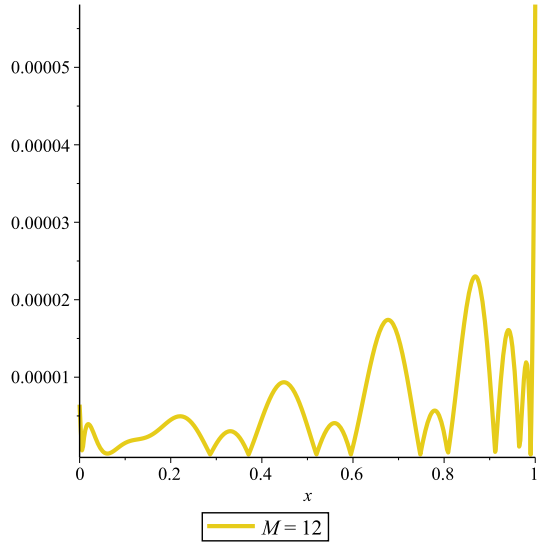


Figure 2: Absolute errors for CLS-SVR method (above) at $M = 15, \rho = 10^{20}$ and GLS-SVR method (below) at $M = 12$ and $\rho = 10^{20}$ for Example 2.

The absolute error for $\rho = 10^{20}, M = 15$ for CLS-SVR (above) and $\rho = 10^{20}, M = 12$ GLS-SVR (below) display in Figure 2. Moreover, we have the report of maximum absolute error with $\rho = 10^{20}$ and $M = 8, 10$ for CLS-SVR and for GLS-SVR methods in Figure 3. Moreover, the report of CPU time for CLS-SVR and GLS-SVR methods with various values of M for Example 2 displayed in Table 4.

5 Discussion

In this paper, a hybrid method based on LS-SVR and with the collocation and Galerkin methods has been presented for solving Fredholm integral equations. For the basis, the orthonormal Bernoulli polynomials have been used to expand the unknown solution in finite dimensions. This process allows you to leverage machine learning techniques to approximate the solution of Fredholm integral equations and transform the optimization problem into a more tractable system of algebraic equations. To demonstrate the efficiency and accuracy of our method, two examples are provided.

References

[1] S. H. Lo, C. Y. Dong and Y. K. Cheung, Integral equation approach for 3D multiple-crack problems, *Engineering fracture mechanics* 72 (2005) 1830–1840.

[2] V. V. Kulish and V. B. Novozhilov, Integral equation for the heat transfer with the moving boundary, *Journal of thermophysics and heat transfer* 17 (2003) 538–540.

[3] J. A. Rad, K. Parand and S. Abbasbandy, Local weak form meshless techniques based on the radial point interpolation (RPI) method and local boundary integral equation (LBIE) method to evaluate european and American options, *Communications in Nonlinear Science and Numerical Simulation* 22 (1–3) (2015) 1178–1200.

[4] P. Darania and A. Ebadian, A method for the numerical solution of the integro differential equations, *Applied Mathematics and Computation* 188 (1) (2007) 657–668.

[5] A. Karamete and M. Sezer, A Taylor collocation method for the solution of linear integro-differential equations, *International Journal of Computer Mathematics* 79 (9) (2002) 987–1000.

[6] S. M. El-Sayed and M. R. Abdel-Aziz, 2003, A comparison of Adomian's decomposition method and Wavelet-Galerkin method for solving integro-differential equations, *Applied Mathematics and Computation* 136 (1) (2003) 151–159.

[7] R. Alexander, Diagonally implicit Runge–Kutta methods for stiff ODE's, *SIAM Journal on Numerical Analysis* 14 (6) (1977) 1006–1021.

[8] A. Avudainayagam and C. Vani, Wavelet-Galerkin method for integro-differential equations, *Applied Numerical Mathematics* 32 (3) (2000) 247–254.

[9] S. A. Ashour, Numerical solution of integral equations with finite part integrals, *International Journal of Mathematics and Mathematical Sciences* 22 (1) (1999) 155–160.

[10] M. Sezer and M. Kaynak, Chebyshev polynomial solutions of linear differential equations, *International Journal of Mathematical Education in Science and Technology* 27 (4) (1996) 607–618.

[11] Z. Hajimohammadi, F. Baharifard and K. Parand, A new numerical learning approach to solve general falkner–skan model, *Engineering with Computers* (2020) 1–17.

[12] K. Parand K., A. A. Aghaei, M. Jani and A Ghodsi, A new approach to the numerical solution of Fredholm integral equations using least squares-support vector regression, *Mathematics and Computers in Simulation* 180 (2021) 114–128.

[13] K. Parand, M. Razzaghi, R. Sahleh and M. Jani, Least squares support vector regression for solving volterra integral equations, *Engineering with Computers* 38 (2022) 789–796.

[14] A. J. Smola and B. Schölkopf, *Learning with Kernels*, MIT Press, Cambridge, MA, 2002.

[15] V. Vapnik, *Statistical learning theory*, John Wiley & Sons google schola, (1998) 831-842.

[16] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 1st edn. Springer, New York (1995)

[17] V. Vapnik, Support-vector networks, *Machine learning*, 20 (1995) 273–297.

[18] S. Mall and S. Chakraverty, Chebyshev neural network based model for solving Lane-Emden type equations, *Applied Mathematics and Computation* 247 (2014) 100–114.

[19] A. Golbabai and S. Seifollahi, Numerical solution of the second kind integral equations using radial basis function networks, *Applied Mathematics and computation* 174 (2) (2006) 877–883.

[20] Y. Lu, Q. Yin, H. Li, H. Sun, Y. Yang and M. Hou, Solving higher order nonlinear ordinary differential equations with least squares support vector machines, *Journal of Industrial & Management Optimization* 16 (3) (2020)1481.

[21] S. Mehrkanoon, T. Falck and J. A. K. Suykens, Approximate solutions to ordinary differential equations using least squares support vector, *IEEE transactions on neural networks and learning systems* 23 (9) (2012) 1356–1367.

[22] S. Mehrkanoon and J. A. K. Suykens, Learning solutions to partial differential equations using LS-SVM, *Neurocomputing* 159 (2015) 105–116.

[23] S. Mehrkanoon and J. A. K. Suykens, LS-SVM approximate solution to linear time varying descriptor systems, *Automatica* 48 (10) (2012) 2502–2511.

[24] G. Zhang, S. Wang, Y. Wang and W. Liu, LS-SVM approximate solution for affine nonlinear systems with partially unknown functions, *Journal of Industrial & Management Optimization* 10 (2) (2014) 621–636.

[25] Q. Wang, K. Wang and S. Chen, Least squares approximation method for the solution of Volterra–Fredholm integral equations, *Journal of Computational and Applied Mathematics* 272 (2014) 141–147.

[26] U. P. SINGH, Application of orthonormal bernoulli polynomials for approximate solution of some Volterra integral equations, *Palestine Journal of Mathematics* 11 (2022) 162–170.

[27] P. K. Sahu and B. Mallick, Approximate Solution of Fractional Order Lane–Emden Type Differential Equation by Orthonormal Bernoulli's Polynomials, *International Journal of Applied and Computational Mathematics* 5 (3) (2019) p. 89.

[28] M. Pourbabaee and A. Saadatmandi, New operational matrix of Riemann Liouville fractional derivative of orthonormal Bernoulli polynomials for the numerical solution of some distributed-order time-fractional partial differential equations, *Journal of Applied Analysis & Computation*, **13** (6) (2023) 3352-3373.

[29] M. H. Heydari and Z. Avazzadeh, New formulation of the orthonormal Bernoulli polynomials for solving the variable-order time fractional coupled Boussinesq-Burger's equations, *Engineering with computers*, 37 (2021) 3509–3517.

[30] N. Samadyar and F. Mirzaee, Orthonormal Bernoulli polynomials collocation approach for solving stochastic It^o-Volterra integral equations of Abel type, *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields* 33 (1) (2020) e2688.

[31] S. Bazm, Bernoulli polynomials for the numerical solution of some classes of linear and nonlinear integral equations, *Journal of Computational and Applied Mathematics* 275 (2015) 44–60.

[32] S. Mashayekhi, Y. Ordokhani and M. Razzaghi, Hybrid functions approach for nonlinear constrained optimal control problems, *Communications in Nonlinear Science and Numerical Simulation*, 17 (4) (2012) 1831–1843.

[33] M. Pourbabaee and A. Saadatmandi, A novel Legendre operational matrix for distributed-order fractional differential equations, *Applied Mathematics and Computation* 361 (2019) 215–231.

[34] S. Mehrkanoon and J. A. K. Suykens, Deep hybrid neural–kernel networks using random Fourier features, *Neurocomputing* 298 (2018) 46–54.

[35] Y. Lu, Q. Yin, H. Li, H. Sun, Y. Yang and M. Hou, The LS-SVM algorithms for boundary value problems of high-order ordinary differential equations, *Advances in Difference Equations* 2019 (2019) 1–22.

[36] J. A. K. Suykens and J. Vandewalle, Least squares support vector machine classifiers, *Neural processing letters* 9 (3) (1999) 293–300.