Analytical and Numerical Solutions for Nonlinear Equations
©Available online at https://ansne.du.ac.ir/
Online ISSN: 3060–785X
2027, Volume 12, Issue 1, pp. 50–67
Research article

A Constrained FEM-LSSVR Framework with B-Spline Basis for Data-Driven Solution of Nonlinear PDEs

Sima Naraghi[1], Kourosh Parand[2,3,*]

[1] Department of Applied Mathematics, Faculty of Mathematical Sciences, Shahid Beheshti University, Tehran, Iran

[2] Department of Computer and Data Sciences, Faculty of Mathematical Sciences, Shahid Beheshti University, Tehran, Iran

[3] Institute for Cognitive and Brain Sciences, Shahid Beheshti University, Tehran, Iran

* **Corresponding author(s):** k_parand@sbu.ac.ir

Received: 25/08/2026 **Revised:** 08/09/2026 **Accepted:** 19/09/2026 **Published:** 02/11/2026 10.22128/ansne.2026.3438.1237

Abstract

This paper presents a constrained hybrid FEM–LSSVR framework for the numerical solution of nonlinear partial differential equations, with a particular focus on FitzHugh–Nagumo (FHN)-type model variants. In the proposed approach, Least Squares Support Vector Regression (LSSVR) is not employed as a post-processing or data-fitting tool; instead, it is formulated as a local primal optimization problem in which the governing PDE, boundary conditions, and inter-element continuity are explicitly enforced as constraints. The finite element method (FEM) is incorporated to provide structural consistency, embedding boundary and continuity information directly into the learning formulation rather than using FEM solutions as training data. The resulting method establishes a tight interaction between numerical discretization and constrained learning, yielding a flexible yet physically consistent approximation framework. The performance of the proposed approach is investigated through numerical experiments on several FHN-type nonlinear PDE variants, including studies with varying mesh resolutions, different final time horizons, and data-driven settings. Comparisons with standard FEM solutions demonstrate improved robustness and accuracy, particularly in scenarios involving limited or noisy data. These results indicate that the constrained FEM–LSSVR formulation provides a viable alternative to traditional numerical solvers for nonlinear PDEs in data-driven contexts.

Keywords: Finite element method, Least squares support vector regression, Data-driven methods, Nonlinear partial differential equations.

Mathematics Subject Classification (2020): 65M60, 68T07, 65D07, 35K57

1 Introduction

Differential equations provide a fundamental mathematical framework for modeling physical and mathematical laws governing complex systems, a field that requires innovative problem formulation and analytical insight [1]. With advances in science and technology, increasingly complex models have emerged that often lack closed-form analytical solutions and therefore necessitate the use of numerical methods [2]. Differential equations arise as mathematical models in a wide range of applications, including industrial processes [3],

COPYRIGHTS: ©2027, Analytical and Numerical Solutions for Nonlinear Equations published by Damghan University.
This is an open access article published under the *CC BY 4 DEED* license

healthcare systems [4], aerospace science [5], and more sophisticated settings such as disease dynamics and control strategies [6, 7]. The development of reliable and accurate numerical techniques for nonlinear partial differential equations remains an active research topic in computational applied mathematics [8].

Mathematical and physical models formulated as ordinary or partial differential equations are commonly solved approximately using numerical methods [9]. Among the most widely used approaches are the finite difference method, finite element method , finite volume method, and isogeometric analysis [10–13]. These techniques play a central role in obtaining approximate solutions while respecting the underlying structure of the governing equations. Depending on the problem geometry, complexity, and computational efficiency requirements, each method can be effectively employed for different classes of problems [14].

A fundamental limitation of traditional numerical methods for partial differential equations is their inability to directly incorporate experimental or empirical data, as the governing equations themselves do not possess learning capability from real-world observations [15]. In recent years, data-driven and machine learning-based approaches have gained increasing attention as alternative tools for solving differential equations, particularly in complex scenarios where classical methods face challenges [16]. A prominent example is the Physics-Informed Neural Networks (PINNs) framework introduced by Raissi et al. [17], which embeds physical laws directly into neural network training. PINNs have been successfully applied to a wide range of physical and scientific problems requiring adherence to governing equations. Furthermore, enhancements based on Chebyshev polynomials, radial basis functions, and collocation strategies have been proposed to improve the accuracy and efficiency of physics-informed learning methods [18, 19]. In this direction, residual-based architectures such as the Residual Integral Solver Network (RISN) have been developed for solving complex integral and integro-differential equations [20].

Another effective machine learning approach for solving differential equations is the Least Squares Support Vector Machine framework. This methodology was initially applied to differential equation problems by Mehrkanoon and Suykens using radial basis function kernels [21]. Subsequent studies have extended this approach by designing problem-adapted kernels and multi-term bases, including Legendre, Chebyshev, and Jacobi polynomials, leading to significant improvements in accuracy and convergence properties [14, 22, 23]. These developments have enabled high-accuracy solutions for challenging problems such as optimal control and inverse modeling [24, 25]. More recently, symbolic and evolutionary extensions of Least Squares Support Vector Regression, such as the GELS–SVR framework, have demonstrated the effectiveness of combining grammatical evolution with least squares support vector regression for solving boundary-layer type equations, including the Falkner–Skan equation [26]. Along the same line, a double deep Q-network-powered approach has been proposed for discovering symbolic solutions of nonlinear integral equations [27], and physics-informed neural networks combined with distributed particle swarm optimization have been employed to obtain symbolic solutions of fractional Volterra population models [28]. Compared to PINNs, LSSVR-based approaches are particularly attractive for problems requiring faster implementation and lower computational cost, whereas PINNs are better suited for high-dimensional and strongly nonlinear problems involving deep network architectures [17, 29, 30].

To further enhance the capabilities of machine learning methods for differential equations, several studies have proposed hybrid strategies that bridge classical numerical discretization techniques with data-driven models. In particular, hybrid formulations combining FEM with learning-based approaches have been shown to improve solution accuracy, alleviate high-dimensional learning difficulties, and reduce computational cost [31]. Related developments include the Deep Finite Volume Method (DFVM), which exploits weak formulations and derivative reduction strategies to efficiently solve higher-order partial differential equations [32].

Despite these advances, many existing LSSVR-based and hybrid frameworks treat numerical discretization and learning as separate components or rely primarily on collocation-based formulations. In particular, boundary conditions and inter-element continuity are often enforced weakly or indirectly, which may compromise numerical stability and physical consistency in data-driven settings.

In this study, we propose a constrained hybrid algorithm that combines the Finite Element Method with B-spline-based Least Squares Support Vector Regression. Unlike conventional approaches that employ FEM solely as a source of training data, the proposed method embeds FEM-derived boundary conditions and inter-element continuity constraints directly into the LSSVR optimization problem. This formulation establishes a tight coupling between numerical discretization and constrained learning, enabling robust and physically consistent solutions for data-driven nonlinear partial differential equations.

The main contributions of this paper can be summarized as follows:

- A constrained hybrid FEM–LSSVR framework for solving nonlinear partial differential equations.

- A data-driven formulation in which FEM-based boundary and continuity information is incorporated explicitly as constraints rather than training data.
- A comprehensive comparative study with standard and noisy FEM solutions for neuro-cognitive models of FitzHugh–Nagumo type.

The remainder of this paper is organized as follows. Section 2 reviews the fundamental concepts of the Finite Element Method and Least Squares Support Vector Regression. Section 3 presents the proposed constrained FEM–LSSVR formulation and the associated numerical algorithm. Section 4 reports numerical results and comparative studies. Section 5 discusses the obtained results and analyzes the performance of the proposed framework. Finally, Section 6 concludes the paper and outlines possible directions for future research.

2 Preliminaries

2.1 Least Squares Support Vector Regression

Support Vector Regression is a machine learning technique designed for regression problems, where the goal is to predict continuous outcomes based on input data. SVR is based on the concepts of Support Vector Machines, which are commonly applied to classification tasks. The key advantage of SVR lies in its capability to model non-linear relationships between inputs and outputs using kernel functions, while also maintaining robustness to noise in the data. Least Squares Support Vector Regression is a modified version of SVR that simplifies the optimization process and reduces computational demands. Instead of solving a conventional quadratic programming problem as in standard SVR, LSSVR employs a least squares formulation, making it more efficient and easier to implement.

The core concept of SVR is to find a function $f(x)$ that models the relationship between the input vector x and the output y, ensuring that the error margin is minimized. The goal is to position most data points within a predefined margin ε, while minimizing the number of support vectors.

LSSVR simplifies the traditional SVR formulation by replacing the ε-insensitive loss and the inequality constraints of SVR with a least squares loss and equality constraints. Given training data $\{(x_i, y_i)\}_{i=1}^{N}$, the regression model $f(x) = w^T \phi(x) + b$ is sought, where $\phi(\cdot)$ is a (possibly implicit) feature map, w is the weight vector, and b is the bias term. The LSSVR optimization problem in the regression setting reads [21]:

$$\begin{aligned} \min_{w,b,e} \quad & \frac{1}{2} w^T w + \frac{\gamma}{2} \sum_{i=1}^{N} e_i^2, \\ \text{s.t.} \quad & y_i = w^T \phi(x_i) + b + e_i, \quad i = 1, \ldots, N, \end{aligned} \tag{1}$$

where e_i is the regression error at the i-th training point and $\gamma > 0$ is a regularization parameter that balances model smoothness (through $\frac{1}{2} w^T w$) against data fidelity (through the squared errors). The stationarity conditions of the associated Lagrangian reduce the problem to a single linear system, making LSSVR computationally appealing. We emphasize that the equality constraints in (1) are of regression type; the classification-type relation $y_i(w^T \phi(x_i) + b) = 1 - e_i$ used in LS-SVM classifiers is not employed anywhere in this work.

2.1.1 The Kernel Trick in LSSVR

Similar to standard SVR, LSSVR uses the kernel trick to manage non-linear relationships between input features and outputs. This technique allows the model to implicitly operate in a higher-dimensional feature space without explicitly computing the transformation.

Common kernel functions include:

- **Linear kernel:** $K(x, x') = x^T x'$
- **Polynomial kernel:** $K(x, x') = (x^T x' + c)^d$
- **Radial Basis Function kernel:** $K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$
- **Sigmoid kernel:** $K(x, x') = \tanh(\alpha x^T x' + c)$

Using these kernels, LSSVR can model complex and non-linear patterns without explicitly mapping inputs to higher-dimensional spaces.

2.1.2 LSSVR for Solving Ordinary Differential Equations

When applying machine learning techniques to solve Ordinary Differential Equations (ODEs), LSSVR offers a robust and flexible framework. Its main advantage lies in its capacity to approximate solutions to complex problems, especially when analytical solutions are difficult or impossible to derive.

The LSSVR model can be formulated in two equivalent but computationally distinct forms: the primal form and the dual form. Both are essential for understanding LSSVR-based ODE solvers [21].

2.1.3 Primal Form of LSSVR for Solving ODEs

In the primal formulation, the goal is to minimize the least squares error between the predicted solution $f(x)$ and the actual solution $y(x)$ over the domain. The objective is to find a function $f(x)$ that approximates the ODE solution while satisfying the boundary conditions.

Consider a general Ordinary Differential Equation:

$$\frac{d^n y}{dx^n} = g(x), \quad x \in [a,b], \tag{2}$$

where $g(x)$ is the known forcing function. We apply LSSVR by introducing a kernel-based approximation $\tilde{y}(x)$.

The primal LSSVR optimization problem is defined as [22]:

$$\begin{aligned} \min_{w,e} \quad & \frac{1}{2} w^T w + \frac{\gamma}{2} e^T e \\ \text{s.t.} \quad & \frac{d^n \tilde{y}}{dx^n}\Big|_{x_i} - g(x_i) = e_i, \quad i = 0,1,2,\ldots,N \end{aligned} \tag{3}$$

Here, w is the weight vector, γ is the regularization parameter, e is the vector of constraint errors at the training (collocation) points, and $\tilde{y}$ denotes the kernel-based approximation. In contrast to the data-fitting problem (1), where the constraints match the model output to measured targets y_i, the equality constraints in (3) match the *differential-equation residual* to the error variables: the residual of the governing equation, evaluated at each training point, is required to equal e_i, and the sum of squared residual errors is minimized together with the regularization term. Initial and boundary conditions are appended as additional equality constraints of the same type. This is the mechanism by which the differential equation is incorporated into the least squares constraints, and it is exactly the mechanism used in the hybrid method of Section 3, with the FEM discretization supplying the residual equations.

2.1.4 Dual Formulation of LSSVR

The optimization problem (1) can be reformulated using Lagrange multipliers, resulting in the dual form, which is beneficial when the feature map is accessible only implicitly through a kernel function. Introducing a multiplier α_i for each equality constraint and eliminating w and e through the Karush–Kuhn–Tucker (KKT) conditions leads to the linear system

$$\begin{bmatrix} 0 & 1^T \\ 1 & K + \gamma^{-1} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \tag{4}$$

where K is the kernel matrix with entries $K_{ij} = K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ and α is the vector of Lagrange multipliers. Once (α, b) is determined, the prediction at a new point x is $\hat{y}(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i) + b$.

In the present work, however, the approximation space is spanned by an explicit, finite-dimensional B-spline basis, so the primal formulation can be used directly, with the coefficient vector playing the role of the weight vector, and no kernel evaluation is required. The dual form is recalled here only for completeness; the hybrid method of Section 3 is formulated and solved entirely in the primal.

2.2 Finite Element Method Operator Formulation

The Finite Element Method is a numerical technique used to solve complex partial differential equations in various fields such as engineering, physics, and computational science. It is especially useful for problems with complex geometries, boundary conditions, and variable material properties. FEM finds wide applications in structural analysis, fluid dynamics, heat transfer, and electromagnetic field simulations.

The process begins by converting the problem to its variational or weak form, which is then discretized using FEM to transform the problem into a set of algebraic equations. To apply FEM, we first convert the problem to its weak form, which is easier to solve numerically. The weak form is obtained by multiplying the differential equation by a test function $v(x)$ and integrating by parts.

Consider a linear boundary value problem governed by a second-order differential equation [33]:

$$L(u(x)) = f(x), \quad x \in \Omega, \quad \text{with boundary conditions} \quad u(x) = g(x) \quad \text{on} \quad \partial\Omega$$

Starting with the strong form:

$$L(u(x)) = f(x).$$

We multiply both sides by a test function $v(x)$ and integrate over the domain Ω:

$$\int_\Omega v(x)L(u(x))\,dx = \int_\Omega v(x)f(x)\,dx.$$

Assuming L is a differential operator (e.g., $L = -\nabla^2$), we apply integration by parts, resulting in the weak form:

$$\int_\Omega \nabla v(x) \cdot \nabla u(x)\,dx = \int_\Omega v(x)f(x)\,dx.$$

This weak form eliminates the requirement for $u(x)$ to be twice differentiable and allows us to work with less stringent conditions.

Once we have the weak form, we discretize the solution space using FEM. The domain Ω is divided into smaller subdomains (elements), and the solution $u(x)$ is approximated as a linear combination of basis functions (or shape functions) $\varphi_i(x)$ associated with each node of the mesh.

The approximation of $u(x)$ is:

$$u(x) \approx \sum_{i=1}^{N} u_i \varphi_i(x), \tag{5}$$

where u_i are the unknowns at the nodes of the finite element mesh and $\varphi_i(x)$ are the basis (or shape) functions corresponding to each node.

The test functions $v(x)$ are chosen from the same function space as the trial functions, meaning they are also represented as a linear combination of the basis functions:

$$v(x) = \sum_{j=1}^{N} v_j \varphi_j(x).$$

In the Galerkin method, the test functions $v(x)$ are chosen to be the same as the trial functions $\varphi_i(x)$. The weak formulation of FEM becomes:

$$\int_\Omega \nabla \varphi_i(x) \cdot \nabla \varphi_j(x)\,dx = \int_\Omega \varphi_i(x)f(x)\,dx.$$

This leads to a system of linear equations:

$$Ku = f.$$

3 FEM-LSSVR

The combination of the Finite Element Method and Least Squares Support Vector Regression creates a robust hybrid framework designed to address complex computational challenges, especially in the context of solving partial differential equations or performing data-driven analyses.

3.1 Choice of the B-Spline Basis

Throughout this work the finite element space is built from B-spline basis functions. B-splines possess several properties that make them particularly well suited to the present constrained setting: (i) they have compact support and form a non-negative partition of unity, which yields sparse, banded, and well-conditioned mass and stiffness matrices; (ii) a B-spline basis of degree p is globally C^{p-1}-continuous across element interfaces, so the degree of inter-element smoothness is controlled directly by the choice of the basis, which is precisely the type of structural information that the proposed method enforces as constraints; and (iii) in contrast to global kernel expansions (e.g., radial basis functions), the locality of B-splines keeps every constraint in the optimization problem sparse, while in contrast to higher-order nodal

Lagrange polynomials, B-splines avoid oscillatory shape functions and allow the same assembly process to be reused when moving to higher-order, higher-continuity discretizations.

In all numerical experiments reported in this paper we employ B-splines of degree $p = 1$ on a uniform knot vector whose knots coincide with the mesh nodes $x_i = a + ih$, $i = 0, \ldots, N$, with $h = (b-a)/N$. For $p = 1$ the B-spline basis coincides with the classical piecewise linear "hat" functions $\varphi_i(x)$, and the mass and stiffness matrices are assembled as

$$M_{ij} = \int_\Omega \varphi_i(x)\, \varphi_j(x)\, dx, \qquad D_{ij} = \int_\Omega \varphi_i'(x)\, \varphi_j'(x)\, dx, \tag{6}$$

which on a uniform mesh reduce to the tridiagonal forms

$$M = \frac{h}{6}\,\mathrm{tridiag}(1, 4, 1), \qquad D = \frac{1}{h}\,\mathrm{tridiag}(-1, 2, -1). \tag{7}$$

Higher-degree B-splines can be substituted without any structural change to the method; the reported experiments use $p = 1$.

3.2 Constrained Formulation

To fix the notation, consider the following model problem on the domain Ω:

$$\begin{aligned} &-u''(x) + u(x) = f(x), x \in \Omega = [a, b], \\ &u(a) = g_a, \qquad u(b) = g_b, \end{aligned} \tag{8}$$

with, for instance, $f(x) = e^x$. Multiplying by a test function φ_i, integrating over Ω, and integrating the second-order term by parts (the boundary contributions vanish for the interior test functions) gives the weak form. Substituting the expansion $u(x) \approx \sum_{j=0}^{N} U_j \varphi_j(x)$ of Eq. (5) then yields

$$\sum_{j=0}^{N} U_j \left[\int_\Omega \varphi_j'(x)\, \varphi_i'(x)\, dx + \int_\Omega \varphi_j(x)\, \varphi_i(x)\, dx \right] = \int_\Omega f(x)\, \varphi_i(x)\, dx, \qquad i = 0, \ldots, N. \tag{9}$$

With the matrices of Eq. (6) and the load vector $F_i = \int_\Omega f\, \varphi_i\, dx$, this reads

$$(D + M)U = F, \tag{10}$$

where the sign of the stiffness contribution follows directly from the integration by parts of $-u''$; Dirichlet data are imposed by replacing the first and last rows of (10) by $U_0 = g_a$ and $U_N = g_b$.

Now, by combining the finite element method with the least squares support vector machine method, System (10) is incorporated as a constraint into the optimization problem of the algorithm, resulting in the following form for the hybrid method:

$$\begin{aligned} &\min_{\mathbf{U}, e} \frac{1}{2}\mathbf{U}^T\mathbf{U} + \frac{\gamma}{2}\sum_{i=0}^{N} e_i^2, \\ &\text{s.t.} \quad \sum_{j=0}^{N} \left(D_{ij} + M_{ij}\right) U_j - F_i = e_i, \qquad i = 0, \ldots, N, \end{aligned} \tag{11}$$

in which the rows associated with boundary (and, for time-dependent problems, initial) conditions are replaced by the corresponding constraints $U_0 - g_a = e_0$ and $U_N - g_b = e_N$.

Two points deserve emphasis. First, the unknown vector $\mathbf{U} \in \mathbb{R}^{N+1}$ is the vector of B-spline coefficients of the approximate solution; because the approximation space is spanned by the explicit feature map $\phi(x) = (\varphi_0(x), \ldots, \varphi_N(x))^T$, the same vector $\mathbf{U}$ simultaneously plays the role of the primal weight vector w in (3): the approximate solution is $u(x) = \mathbf{U}^T\phi(x)$. Problem (11) is therefore an LSSVR primal problem posed in the B-spline feature space—the objective is the standard LSSVR objective, and the equality constraints equate the discrete FEM residuals (rather than data mismatches) to the error variables e_i—and not merely a regularized least squares smoothing of the FEM coefficients. Second, in contrast to a plain FEM solve of (10), the discrete equations are enforced only in the least squares sense controlled by γ: for $\gamma \to \infty$ the FEM solution is recovered, while for finite γ the term $\frac{1}{2}\mathbf{U}^T\mathbf{U}$ biases the coefficients toward small magnitude, which damps the amplification of perturbations (such as noisy initial data) through the discrete operator. This is the mechanism behind the robustness observed in the data-driven experiments of Section 4.

Eliminating e from (11) shows that the problem being solved is equivalent to the unconstrained ridge-type problem

$$\min_{\mathbf{U}} \frac{1}{2}\mathbf{U}^T\mathbf{U} + \frac{\gamma}{2}\|A\mathbf{U} - F\|_2^2 \qquad \Longleftrightarrow \qquad \left(\gamma^{-1}I + A^TA\right)\mathbf{U} = A^TF, \tag{12}$$

with $A = D + M$ (after boundary-row replacement). In the implementation, the constrained form (11) is passed directly to a nonlinear programming solver (Maple's `NLPSolve`), which is convenient when additional constraints are appended and returns the same minimizer as (12).

3.3 Time-Dependent Problems

For an evolution equation $u_t = u_{xx} + r(u)$ with reaction term $r(u)$, the same construction is applied at every time step. Let $\mathbf{U}^n$ collect the B-spline coefficients at time $t_n = nk$, where k is the time step. Discretizing in time with the diffusion term treated implicitly and the nonlinear reaction term evaluated at the previous time level (semi-implicit Euler), the Galerkin system at step $n+1$ reads

$$\underbrace{(M + kD)}_{A}\,\mathbf{U}^{n+1} = \underbrace{M\mathbf{U}^n + kR(\mathbf{U}^n)}_{F^n}, \tag{13}$$

where $R(\mathbf{U}^n)$ denotes the load vector obtained by evaluating the reaction term with the previous solution, and the first and last rows of (13) are replaced by the Dirichlet data $U_0^{n+1} = u(a, t_{n+1})$ and $U_N^{n+1} = u(b, t_{n+1})$. The optimization problem actually solved at each time step is then exactly (11) with the constraint $A\mathbf{U}^{n+1} - F^n = e$:

$$\min_{\mathbf{U}^{n+1},\,e} \frac{1}{2}(\mathbf{U}^{n+1})^T\mathbf{U}^{n+1} + \frac{\gamma}{2}\sum_{i=0}^{N} e_i^2 \qquad \text{s.t.} \qquad A\mathbf{U}^{n+1} - F^n = e. \tag{14}$$

At $n = 0$ the initial coefficients $\mathbf{U}^0$ are obtained by evaluating the initial condition at the mesh nodes; in the data-driven variants, the measured (possibly noisy) samples at $t = 0$ are used directly in place of these values.

The complete procedure for time-dependent equations is summarized in Algorithm 1.

Algorithm 1 Pseudocode for FEM–LSSVR method.

1: Set parameters: regularization γ, number of elements N, element size $h = (b-a)/N$, final time T, number of time steps N_t, and time step $k = T/N_t$.
2: Define the degree-one B-spline basis $\{\varphi_i\}_{i=0}^{N}$ and assemble the mass matrix M and stiffness matrix D of Eq. (7).
3: Derive the weak form of the model and the semi-implicit time-stepping system (13).
4: Initialize $\mathbf{U}^0$ from the initial condition evaluated at the mesh nodes. If the initial condition is unknown but data points have been collected at $t = 0$, adopt the data-driven approach and use the collected (possibly noisy) data as $\mathbf{U}^0$.
5: **for** $n = 0$ to $N_t - 1$ **do**
6: Assemble $F^n = M\mathbf{U}^n + kR(\mathbf{U}^n)$ and impose the boundary values at t_{n+1} by row replacement, cf. Eq. (13).
7: Form the primal LSSVR problem of Eq. (14):

$$\min_{\mathbf{U}^{n+1},\,e} \frac{1}{2}(\mathbf{U}^{n+1})^T\mathbf{U}^{n+1} + \frac{\gamma}{2}\sum_{i=0}^{N} e_i^2,$$
$$\text{s.t.} \quad \sum_{j=0}^{N} A_{ij}U_j^{n+1} - F_i^n = e_i, \qquad i = 0, \ldots, N.$$

8: Solve the optimization problem (e.g., with `NLPSolve`) and obtain $\mathbf{U}^{n+1}$.
9: **end for**
10: Return the coefficient history $\{\mathbf{U}^n\}_{n=0}^{N_t}$; the approximate solution is $u(x, t_n) = \sum_{i=0}^{N} U_i^n \varphi_i(x)$.

Algorithm 1 advances the solution in time exactly as a classical FEM time-stepper would, except that the linear solve at each step is replaced by the small constrained optimization problem (14). At every step, the data of the problem enter through F^n (previous solution,

reaction term, and boundary values), the physical structure of the problem enters through the constraint matrix A (mass and stiffness contributions), and the learning component enters through the LSSVR objective, which trades off exact satisfaction of the discrete equations against the magnitude of the coefficient vector. The regularization parameter γ is the only tuning parameter of the method; the values used in the experiments, together with all data-driven and noise settings, are collected in Table 1.

4 Numerical Results

In this section, we will tackle examples utilizing the proposed approach for the nonlinear FitzHugh–Nagumo equation under various boundary conditions, and we will present the results obtained.

Throughout this section, the accuracy of a numerical solution u_h is measured against the exact solution u as follows. Let x_i, $i = 0,\dots,N$, denote the mesh nodes, t_n, $n = 0,\dots,N_t$, the time levels, and let $\varepsilon_i = |u_h(x_i,T) - u(x_i,T)|$ be the nodal errors at the final time T. We report

$$\begin{aligned} \mathrm{MAE} &= \frac{1}{(N+1)(N_t+1)} \sum_{n=0}^{N_t} \sum_{i=0}^{N} \big|u_h(x_i,t_n) - u(x_i,t_n)\big|, \qquad L_2 = \Big(\sum_{i=0}^{N} \varepsilon_i^2\Big)^{1/2}, \\ \mathrm{RMS} &= \frac{L_2}{\sqrt{N+1}}, \qquad L_\infty = \max_{0\le i\le N} \varepsilon_i, \qquad \mathrm{RE} = \frac{L_2}{\big(\sum_{i=0}^{N} u(x_i,T)^2\big)^{1/2}}, \end{aligned} \tag{15}$$

i.e., the mean absolute error is accumulated over the whole space–time grid, whereas the four norm-based metrics are all computed from one and the same final-time nodal error vector. With these definitions the inequalities $\mathrm{RMS} \le L_\infty \le L_2$ hold automatically, which provides a built-in consistency check of the reported values. The convergence indicators reported below are computed as $\log_2(E_{k-1}/E_k)$ from the mean absolute errors of the corresponding refinement tables.

The settings of all data-driven and noisy experiments are collected in Table 1. In every data-driven run, the $N+1$ nodal values of the initial state at $t = 0$ are treated as measured data; in the noisy variants these values are additionally perturbed by independent zero-mean random noise whose empirical standard deviation σ, measured against the exact initial profile, is listed in the table. All computations were carried out in Maple with `Digits` $= 15$, and the constrained problems were solved with `NLPSolve`.

Table 1. Parameter settings of the data-driven and noisy experiments.

	Example 1	**Example 2**	**Example 3**	**Example 4**
Domain $[a,b]$	$[0,5]$	$[0,5]$	$[0,10]$	$[0,5]$
Final time T	1	1	1	1
Elements N (data-driven runs)	4–64	4–64	64	4, 8
Element size $h = (b-a)/N$	$5/N$	$5/N$	$10/N$	$5/N$
Time step $k = T/N$	$1/N$	$1/N$	$1/N$	$1/N$
Initial data points	$N+1$	$N+1$	$N+1$	$N+1$
Noise std. σ (noisy runs)	$\approx 1.1\times10^{-2}$	$\approx 1.3\times10^{-2}$	$\approx 1\times10^{-2}$	$\approx 1.5\times10^{-4}$
Regularization γ	10^5	10^8	10^8	10^8

Noise values are the empirical standard deviations of the perturbed initial samples measured against the exact initial profiles; the maximum perturbation never exceeded 0.03.

Example 1. In this example, we consider the nonlinear standard FitzHugh–Nagumo partial differential equation [34]:

$$u_t = u_{xx} - u(1-u)(\rho - u), \quad 0 \le x \le 5,\ t \ge 0, \tag{16}$$

with the initial and boundary conditions defined as follows:

$$u(x,0) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x}{2\sqrt{2}}\right), \tag{17}$$

$$u(0,t) = \frac{1}{2} + \frac{1}{2}\tanh\left(-\frac{(2\rho-1)t}{4}\right), \tag{18}$$

$$u(5,t) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{1}{2\sqrt{2}}\left(5 - \frac{(2\rho-1)}{\sqrt{2}}t\right)\right). \tag{19}$$

The exact solution is given by:

$$u(x,t) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{1}{2\sqrt{2}}\left(x - \frac{2\rho-1}{\sqrt{2}}t\right)\right). \tag{20}$$

The initial and boundary conditions (17)–(19) are the restrictions of the exact solution (20) to $t = 0$, $x = 0$, and $x = 5$, respectively, and it can be verified by direct substitution that (20) satisfies Eq. (16). For the value $\rho = 1/2$ used in the experiments below, the wave speed $(2\rho - 1)/\sqrt{2}$ vanishes, so the exact solution is a stationary front.

Table 2 shows the mean absolute error obtained for $\rho = 0.5$ using FEM–LSSVR, data-driven FEM–LSSVR, and data-driven FEM methods across different mesh resolutions. Table 3 presents the estimated convergence rates for the three methods, computed from the mean absolute errors of Table 2. Table 4 provides a comparison of norm-based error metrics (RMS, L_2, L_∞, and RE) for data-driven FEM and Hybrid methods using 32 elements, evaluated at the final time $T = 1$ according to Eq. (15). Figures 1 and 2 illustrate the numerical solutions and absolute errors for both the clean and noisy versions. Noisy initial data were also taken into account in the analysis, and the corresponding results are presented in Figure 2.

Table 2. Example 1 ($\rho = 0.5$): mean absolute error over the space–time grid for the three methods under mesh refinement.

Number of Elements	FEM–LSSVR	data-driven FEM–LSSVR	data-driven FEM
4	8.90814×10^{-4}	3.0361×10^{-3}	1.1228×10^{-1}
8	2.57967×10^{-4}	2.3846×10^{-3}	1.2403×10^{-1}
16	6.83955×10^{-5}	2.1661×10^{-3}	1.2278×10^{-1}
32	2.37424×10^{-5}	5.6166×10^{-4}	6.9168×10^{-2}

MAE as defined in Eq. (15); the data-driven columns use the noisy initial data described in Table 1.

Table 3. Example 1: convergence indicators $\log_2(E_{k-1}/E_k)$ computed from the mean absolute errors of Table 2.

FEM–LSSVR	data-driven FEM–LSSVR	data-driven FEM
1.7882e + 00	3.5030e − 01	−1.4350e − 01
1.9112e + 00	1.4020e − 01	1.4600e − 02
1.5238e + 00	1.9464e + 00	8.2890e − 01

Each row corresponds to one refinement step of Table 2 ($4 \to 8$, $8 \to 16$, $16 \to 32$ elements); E_k denotes the MAE at refinement level k.

Table 4. Example 1 (data-driven setting, 32 elements): error norms at the final time $T = 1$, computed according to Eq. (15).

Method	RMS Error	L_2 Norm	L_∞ Norm	Relative Error (RE)
FEM	1.3842×10^{-1}	7.9516×10^{-1}	1.8603×10^{-1}	1.6834×10^{-1}
Hybrid	7.0320×10^{-3}	4.0396×10^{-2}	9.2180×10^{-3}	8.5519×10^{-3}

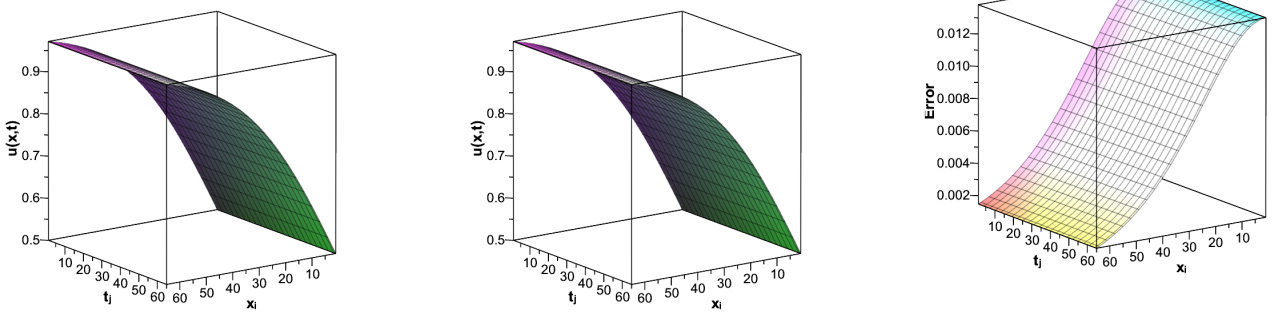
All four metrics are computed from the same final-time nodal error vector, cf. Eq. (15).

Example 2. In this example, we consider the nonlinear standard FitzHugh–Nagumo partial differential equation [34]:

$$u_t = u_{xx} - u(1-u)(\rho - u), \quad 0 \le x \le 5, \quad t \ge 0, \tag{21}$$

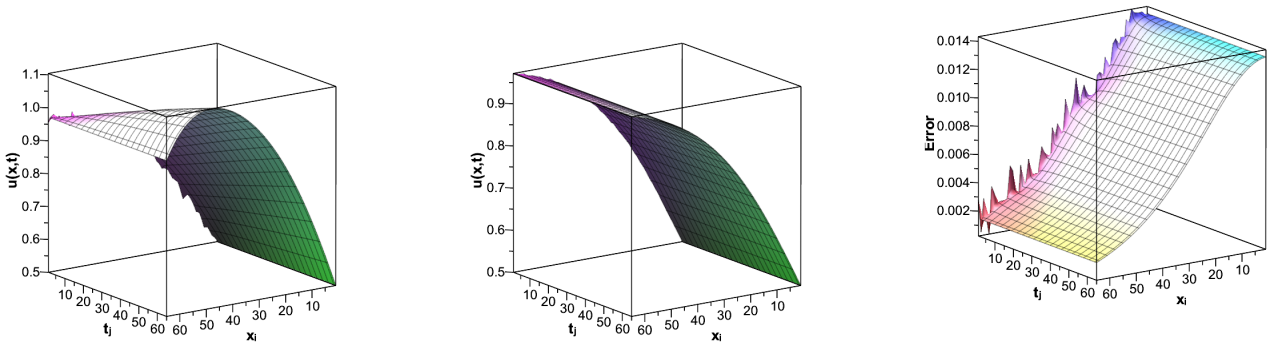
with the initial and boundary conditions:

$$u(x,0) = \frac{1}{2}(1+\rho) + \left(\frac{1}{2} - \frac{1}{2}\rho\right)\tanh\left(\sqrt{2}(1-\rho)\frac{x}{4}\right), \tag{22}$$



(a) FEM–LSSVR method **(b)** Exact solution **(c)** Absolute error for FEM–LSSVR method

Figure 1. Numerical results for Example 1 with 64 elements: $x_i \in [0,5],\ i = 0,\ldots,64;\quad t_j \in [0,1],\ j = 0,\ldots,64.$



(a) Data-driven FEM method **(b)** Data-driven FEM–LSSVR method **(c)** Absolute error for Data-driven FEM–LSSVR method

Figure 2. Comparison of numerical results for Example 1 in the noisy setting with 64 elements: $x_i \in [0,5],\ i = 0,\ldots,64;\quad t_j \in [0,1],\ j = 0,\ldots,64.$

Table 5. Example 2 ($\rho = 0.5$): mean absolute error over the space–time grid for the three methods under mesh refinement.

Number of Elements	FEM–LSSVR	data-driven FEM–LSSVR	data-driven FEM
4	3.5651×10^{-3}	1.4167×10^{-3}	7.7952×10^{-3}
8	1.8140×10^{-3}	1.3776×10^{-3}	5.9263×10^{-3}
16	9.3177×10^{-4}	6.6089×10^{-4}	3.8596×10^{-3}
32	4.7001×10^{-4}	4.9331×10^{-4}	3.9007×10^{-3}

MAE as defined in Eq. (15); the data-driven columns use the noisy initial data described in Table 1.

$$u(0,t) = \frac{1}{2}(1+\rho) + \left(\frac{1}{2} - \frac{1}{2}\rho\right) \tanh\left(\frac{(1-\rho^2)}{4}t\right), \tag{23}$$

$$u(5,t) = \frac{1}{2}(1+\rho) + \left(\frac{1}{2} - \frac{1}{2}\rho\right) \tanh\left(\sqrt{2}(1-\rho)\frac{5}{4} + \frac{(1-\rho^2)}{4}t\right), \tag{24}$$

Table 6. Example 2: convergence indicators $\log_2(E_{k-1}/E_k)$ computed from the mean absolute errors of Table 5.

FEM–LSSVR	data-driven FEM–LSSVR	data-driven FEM
$9.7474e-01$	$4.0377e-02$	$3.9545e-01$
$9.6116e-01$	$1.0597e+00$	$6.1868e-01$
$9.8729e-01$	$4.2192e-01$	$-1.5282e-02$

Each row corresponds to one refinement step of Table 5 ($4\to8$, $8\to16$, $16\to32$ elements); E_k denotes the MAE at refinement level k.

Table 7. Example 2 (data-driven setting, 32 elements): error norms at the final time $T=1$, computed according to Eq. (15).

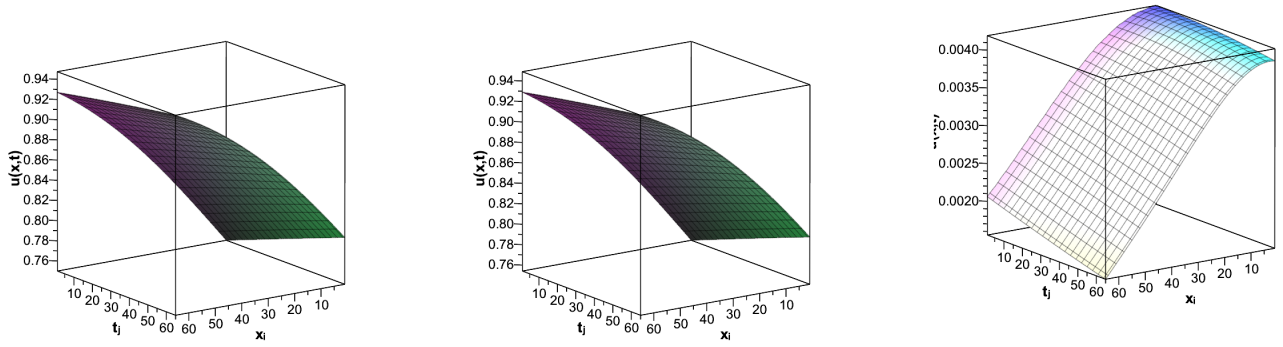
Method	RMS Error	L_2 Norm	L_∞ Norm	Relative Error (RE)
FEM	4.9749×10^{-3}	2.8579×10^{-2}	9.1454×10^{-3}	5.6264×10^{-3}
Hybrid	3.4534×10^{-4}	1.9838×10^{-3}	4.7719×10^{-4}	3.9055×10^{-4}

All four metrics are computed from the same final-time nodal error vector, cf. Eq. (15).

The exact solution is given by:

$$u(x,t)=\frac{1}{2}(1+\rho)+\left(\frac{1}{2}-\frac{1}{2}\rho\right)\tanh\left(\sqrt{2}(1-\rho)\frac{x}{4}+\frac{(1-\rho^2)}{4}t\right). \tag{25}$$

Table 5 presents the mean absolute error for Example 2 with $\rho=0.5$ using FEM, FEM–LSSVR, and their data-driven counterparts for different numbers of elements. Table 6 shows the estimated convergence behavior of these methods, computed from the mean absolute errors of Table 5. Table 7 provides a comparison of norm-based error metrics (RMS, L_2, L_∞, and RE) for data-driven FEM and Hybrid methods using 32 elements, evaluated at the final time $T=1$ according to Eq. (15). Figures 3 and 4 show the numerical solution and the corresponding absolute error in both noise-free and noisy settings.



(a) FEM–LSSVR method **(b)** Exact solution **(c)** Absolute error for FEM–LSSVR method

Figure 3. Numerical results for Example 2 with 64 elements: $x_i\in[0,5]$, $i=0,\ldots,64$; $t_j\in[0,1]$, $j=0,\ldots,64$.

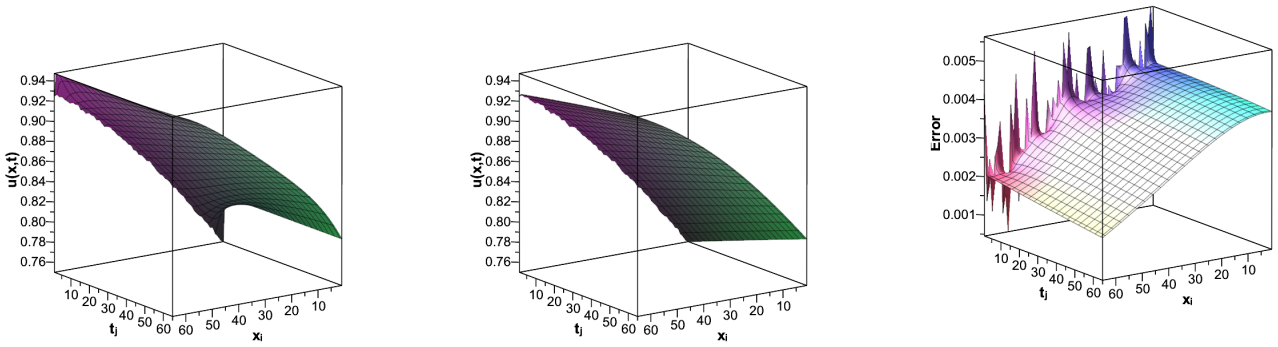
Example 3. In this instance, a nonlinear FitzHugh–Nagumo equation with time-dependent coefficients has been considered [34].

$$u_t+\cos(t)u_x-\cos(t)u_{xx}+2\cos(t)\,(u(1-u)(\rho-u))=0;\quad 0\le x\le 10,\quad t\ge 0, \tag{26}$$

The initial and boundary conditions for the model are defined as follows:

$$u(x,0)=\frac{\rho}{2}+\frac{\rho}{2}\tanh\left(\frac{\rho x}{2}\right), \tag{27}$$

$$u(A,t)=\frac{\rho}{2}+\frac{\rho}{2}\tanh\left(\frac{\rho}{2}\,(A-(3-\rho)\sin(t))\right), \tag{28}$$



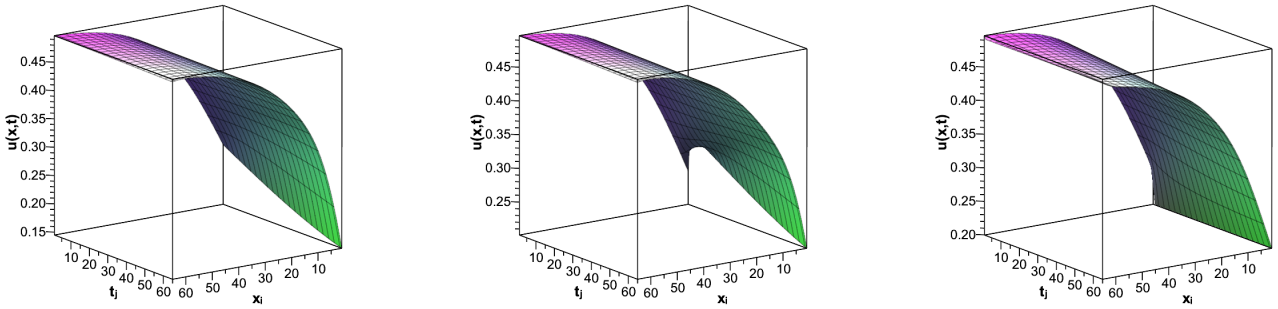
(a) Data-driven FEM method **(b)** Data-driven FEM–LSSVR method **(c)** Absolute error for Data-driven FEM–LSSVR method

Figure 4. Comparison of numerical results for Example 2 in the noisy setting with 64 elements: $x_i \in [0,5],\ i = 0,\ldots,64; \quad t_j \in [0,1],\ j = 0,\ldots,64.$

Table 8. Example 3 ($\rho = 0.5$): hybrid data-driven FEM–LSSVR solution versus the closed-form solution (30) at time $t = 0.9$.

x	Hybrid data driven FEM–LSSVR	Closed-form Solution
1.0	0.194171	0.191225
3.0	0.315193	0.313672
5.0	0.417573	0.410331
7.0	0.467231	0.462795
9.0	0.486202	0.485637

Values of both solutions at the interior nodes $x = 1,3,5,7,9$ of $[0,10]$ at $t = 0.9$.



(a) FEM–LSSVR method **(b)** data-driven FEM method **(c)** data-driven FEM–LSSVR method

Figure 5. Comparison of numerical results for Example 3 with 64 elements: $x_i \in [0,10],\ i = 0,\ldots,64; \quad t_j \in [0,1],\ j = 0,\ldots,64.$

$$u(B,t) = \frac{\rho}{2} + \frac{\rho}{2}\tanh\left(\frac{\rho}{2}\left(B - (3-\rho)\sin(t)\right)\right), \tag{29}$$

where $A = 0$ and $B = 10$ denote the left and right endpoints of the spatial domain. The associated closed-form solution is

$$u(x,t) = \frac{\rho}{2} + \frac{\rho}{2}\tanh\left(\frac{\rho}{2}\left(x - (3-\rho)\sin t\right)\right), \tag{30}$$

whose restrictions to $t = 0$, $x = A$, and $x = B$ reproduce (27)–(29); substitution of (30) into Eq. (26) verifies that it is an exact solution.

Table 8 presents a comparison between the hybrid method and the closed-form solution (30) for $\rho = 0.5$, evaluated at time $t = 0.9$ at interior points of the domain $[A,B] = [0,10]$. Furthermore, Figure 5 illustrates the results obtained using FEM, hybrid FEM–LSSVR, and data-driven hybrid methods.

Table 9. Example 4 (Huxley equation): mean absolute error over the space–time grid for the two tested mesh resolutions.

Number of Elements	FEM–LSSVR	data-driven FEM–LSSVR	data-driven FEM
4	1.4303×10^{-2}	8.5191×10^{-3}	1.5107×10^{-2}
8	8.5928×10^{-3}	7.6993×10^{-3}	6.9297×10^{-2}

MAE as defined in Eq. (15); the data-driven columns use the sampled initial data described in Table 1.

Table 10. Example 4 (data-driven setting, 8 elements): error norms at the final time $T = 1$, computed according to Eq. (15).

Method	RMS Error	L_2 Norm	L_∞ Norm	Relative Error (RE)
FEM	8.3295×10^{-2}	2.4989×10^{-1}	1.2325×10^{-1}	1.1043×10^{-1}
Hybrid	1.1092×10^{-2}	3.3276×10^{-2}	2.4894×10^{-2}	1.4705×10^{-2}

All four metrics are computed from the same final-time nodal error vector, cf. Eq. (15).

Example 4. In this instance, the nonlinear Huxley partial differential equation has been considered [35].

$$u_t = u_{xx} + u(1-u)(u-1); \quad 0 \le x \le 5, \quad t \ge 0, \tag{31}$$

with the initial condition:

$$u(x,0) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x}{2\sqrt{2}}\right), \tag{32}$$

and Dirichlet boundary conditions obtained from the exact solution,

$$u(0,t) = \frac{1}{2} + \frac{1}{2}\tanh\left(-\frac{t}{4}\right), \qquad u(5,t) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{1}{2\sqrt{2}}\left(5 - \frac{t}{\sqrt{2}}\right)\right). \tag{33}$$

The exact solution of this equation is given by:

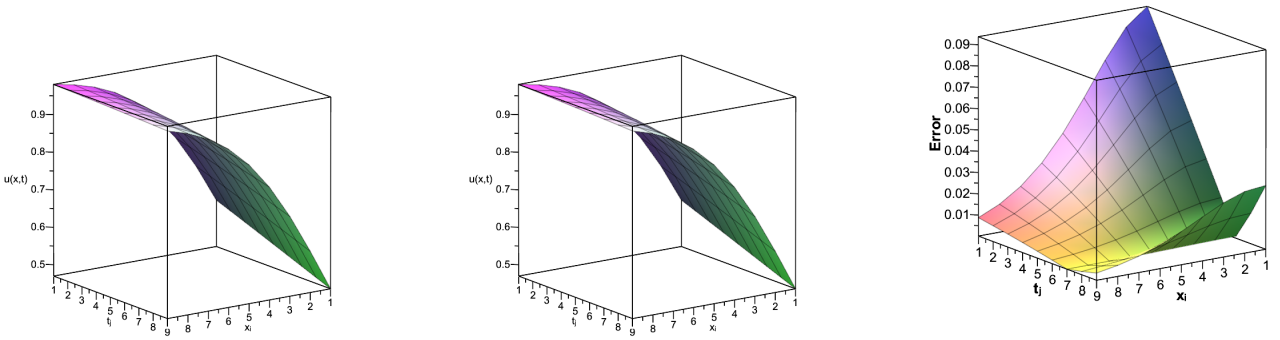
$$u(x,t) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{1}{2\sqrt{2}}\left(x - \frac{t}{\sqrt{2}}\right)\right). \tag{34}$$

Equation (31) is the Huxley equation $u_t = u_{xx} + u(1-u)(u-\gamma_h)$ with $\gamma_h = 1$; direct substitution confirms that (34) satisfies (31) together with (32) and (33).

Table 9 shows the mean absolute error obtained for Example 4 using FEM–LSSVR, data-driven FEM–LSSVR, and data-driven FEM methods across two different mesh resolutions. Since only two mesh resolutions are available for this example, we report the refinement behaviour directly rather than an asymptotic convergence order: halving the element size reduces the MAE of FEM–LSSVR by a factor of about 1.66 and that of the data-driven FEM–LSSVR by a factor of about 1.11, whereas the error of the purely data-driven FEM increases under refinement. Table 10 provides a comparison of norm-based error metrics (RMS, L_2, L_∞, and RE) for FEM and Hybrid methods using 8 elements, evaluated at the final time $T = 1$ according to Eq. (15). Figures 6 and 7 illustrate the numerical solutions and absolute errors for both clean and noisy settings in Example 4.

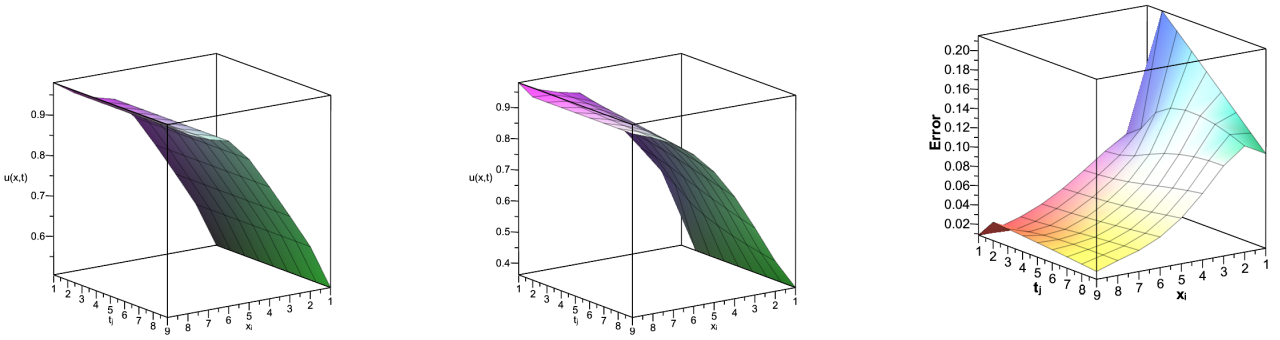
5 Discussion

The proposed FEM–LSSVR framework demonstrates significant improvements in solving FitzHugh-Nagumo model variants, outperforming data-driven FEM approaches. This section provides a detailed discussion of the results across four key examples, highlighting the impact of hyperparameter selection, optimization strategies, and noise handling.



(a) FEM–LSSVR method **(b)** Exact solution **(c)** Absolute error for FEM–LSSVR method

Figure 6. Numerical results for Example 4 with 8 elements: $x_i \in [0,5],\ i = 0,\ldots,8;\quad t_j \in [0,1],\ j = 0,\ldots,8.$



(a) Data-driven FEM method **(b)** Data-driven FEM–LSSVR method **(c)** Absolute error for Data-driven FEM–LSSVR method

Figure 7. Comparison of numerical results for Example 4 in the noisy setting with 8 elements: $x_i \in [0,5],\ i = 0,\ldots,8;\quad t_j \in [0,1],\ j = 0,\ldots,8.$

In Example 1, the FEM–LSSVR method consistently achieves the lowest error and higher convergence rates compared to purely data-driven methods, with its accuracy improving significantly as the mesh is refined. The data-driven FEM–LSSVR method also shows improved performance on finer meshes, reaching a convergence rate close to 2, while the purely data-driven FEM method exhibits unstable behavior and limited sensitivity to mesh refinement. The analysis of various error metrics demonstrates that the Hybrid approach, by simultaneously leveraging the physical structure of the equations and data-driven learning, yields a substantial reduction in error across all norms and maintains strong robustness against noisy initial data. These results confirm that combining the finite element method with machine learning models can enhance both convergence and accuracy, offering a distinct advantage over purely physics-based or purely data-driven approaches, particularly in problems with noise or limited data.

In Example 2, the numerical results presented in Tables 5 to 7 show that all three methods experience error reduction with mesh refinement, but their trends differ significantly. The data-driven FEM–LSSVR method achieves the lowest mean absolute error for most mesh sizes, especially at intermediate resolutions, while FEM–LSSVR exhibits stable convergence rates close to 1.0 (Table 6), indicating predictable improvement with finer meshes. In contrast, the convergence indicator of the purely data-driven FEM method decreases under refinement and becomes slightly negative at the finest level, suggesting that mesh refinement alone does not have a significant impact on its accuracy. The norm-based comparison in Table 7 further highlights the superiority of the Hybrid approach, as it reduces the RMS, L_2,

L_∞, and relative errors by roughly an order of magnitude compared to the data-driven FEM solution. The visual results in Figures 3 and 4 confirm that the Hybrid method preserves solution accuracy and stability in both noise-free and noisy initial data settings, whereas the purely data-driven FEM shows larger localized deviations. These findings further emphasize that integrating FEM with machine learning corrections yields higher accuracy and robustness, particularly when dealing with nonlinear partial differential equations and imperfect data.

In Example 3, the comparison in Table 8 demonstrates that the Hybrid data-driven FEM–LSSVR method produces results in very close agreement with the closed-form solution for $\rho = 0.5$, with deviations remaining small across all tested spatial points. This high level of accuracy indicates that the hybrid approach effectively captures both the spatial and temporal dynamics of the nonlinear FitzHugh–Nagumo equation. The visual comparisons in Figure 5 further confirm the method's robustness: the Hybrid FEM–LSSVR solution maintains smooth profiles and accurately reproduces the reference pattern, whereas the data-driven FEM method exhibits slightly larger deviations, particularly near steep gradients. These results reinforce the conclusion that embedding physical constraints within a learning-based framework enables high-fidelity solutions that remain stable even in challenging nonlinear PDE settings.

Finally, in Example 4, the nonlinear Huxley equation was examined to evaluate the generalizability of the proposed hybrid data-driven FEM–LSSVR method beyond FitzHugh–Nagumo-type models. The numerical results indicate that the data-driven FEM–LSSVR method achieves lower mean absolute errors than the purely data-driven FEM method for both tested mesh resolutions, and in the coarser mesh case even outperforms the standard FEM–LSSVR. Consistent with the refinement comparison reported above, halving the element size reduces the FEM–LSSVR error, whereas the error of the purely data-driven FEM increases, suggesting that mesh refinement alone is insufficient for accuracy gains in the purely data-driven setting. The norm-based comparison highlights the Hybrid method's clear advantage, with substantial reductions across RMS, L_2, L_∞, and relative errors compared to standard FEM. The visual results confirm these trends: the Hybrid approach closely reproduces the exact solution in the clean setting and maintains stability under noisy initial conditions, while the purely data-driven FEM shows larger discrepancies near steep gradients. These findings further reinforce that integrating physical modeling with learning-based corrections leads to improved accuracy and robustness for nonlinear PDEs such as the Huxley equation.

Limitations. The present study has several limitations that should be kept in mind when interpreting the results. All experiments concern one-dimensional scalar problems on uniform meshes, and the reported implementation uses B-splines of degree one; higher-dimensional problems, adaptive meshes, and higher-order bases have not yet been tested. The constrained optimization solved at every time step is more expensive than the single linear solve of a standard FEM step, so no computational-efficiency advantage is claimed for the proposed method. The regularization parameter γ was selected empirically for each example (Table 1), and a systematic strategy for choosing it remains open. The nonlinear reaction terms are treated explicitly in time, which couples the admissible time step to the mesh, and the noise levels considered are moderate (standard deviations up to about 10^{-2}). Finally, the study provides numerical evidence only; a theoretical convergence and stability analysis of the constrained FEM–LSSVR formulation is left for future work.

These results collectively demonstrate that our method achieves consistent convergence and reliable error behavior across all tested PDE scenarios. While a complete theoretical proof of convergence lies beyond the scope of this study, the empirical findings strongly support the stability and effectiveness of the proposed hybrid framework.

6 Conclusion

In this study, we proposed a data-driven finite element method integrated with least squares support vector regression to enhance robustness and predictive accuracy in the numerical simulation of partial differential equations. By leveraging machine learning techniques, the proposed approach effectively captures complex relationships within the dataset while maintaining high precision.

The results on the tested one-dimensional FitzHugh–Nagumo and Huxley problems demonstrate that FEM–LSSVR, even with noisy initial data, outperforms the corresponding data-driven FEM solutions in terms of both accuracy and robustness. We deliberately restrict our claims to what is supported by the reported experiments; in particular, no computational-efficiency or real-time-applicability advantages are claimed, and the behaviour of the method on higher-dimensional and application-scale problems remains to be assessed.

Future research directions include extending this framework to adaptive finite element methods and multi-scale problems to further improve generalizability and accuracy. Additional work could focus on optimizing computational efficiency through parallel processing and distributed learning. Incorporating theoretical convergence guarantees and reliable error estimators may also strengthen the method's applicability to safety-critical and high-resolution simulations.

Authors' Contributions

The contributions of each author to this study are as follows: **Sima Naraghi:** Conceptualization, Methodology, Software, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Kourosh Parand:** Supervision, Writing – review & editing.

Data Availability

The Maple worksheets used to generate all numerical results in this paper are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there is no conflict of interest.

Ethical Considerations

The authors have diligently addressed ethical concerns, such as informed consent, plagiarism, data fabrication, misconduct, falsification, double publication, redundancy, submission, and other related matters.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] F. Mazhar, C. Bartolucci, F. Regazzoni, M. Paci, L. Dedè, A. Quarteroni, C. Corsi, and S. Severi, A detailed mathematical model of the human atrial cardiomyocyte: Integration of electrophysiology and cardiomechanics, The Journal of Physiology, 602(18), 4543–4583, (2024).

[2] H. Dana Mazraeh, K. Parand, H. Farahani, and S. R. Kheradpisheh, An improved imperialist competitive algorithm for solving an inverse form of the huxley equation, Iranian Journal of Numerical Analysis and Optimization, 14(3), 681–707, (2024).

[3] T. Tanuja, L. Kavitha, S. Varma, V. Raju, C. K. Ganteda, M. Obulesu, W. Jamshed, M. R. Eid, and S. M. Hussain, Heat and mass transfer analysis of casson-based hybrid nanofluid flow in the presence of an aligned magnetic field: An application toward mechanical engineering, Proceedings of the Institution of Mechanical Engineers, Part N: Journal of Nanomaterials, Nanoengineering and Nanosystems, 23977914241248546, (2024).

[4] P. Ferrero, A. Tonini, G. Valenti, M. Chessa, L. Kuthi, P. P. Bassareo, L. Dede, and A. Quarteroni, Appraisal of partial anomalous pulmonary venous drainage through a lumped-parameter mathematical model: a new pathophysiological proof of concept, Interdisciplinary CardioVascular and Thoracic Surgery, 39(5), ivae175, (2024).

[5] T. Lassila and G. Rozza, Parametric free-form shape design with pde models and reduced basis method, Computer Methods in Applied Mechanics and Engineering, 199(23-24), 1583–1592, (2010).

[6] S. W. Teklu, B. B. Terefe, D. K. Mamo, and Y. F. Abebaw, Optimal control strategies on hiv/aids and pneumonia co-infection with mathematical modelling approach, Journal of Biological Dynamics, 18(1), 2288873, (2024).

[7] S. Mohammadi and S. R. Hejazi, Lie symmetry, chaos optimal control in non-linear fractional-order diabetes mellitus, human immunodeficiency virus, migraine parkinsons diseases models: using evolutionary algorithms, Computer Methods in Biomechanics and Biomedical Engineering, 27(5), 651–679, (2024).

[8] T. Eto, H. Garcke, and R. Nürnberg, A parametric finite element method for a degenerate multi-phase Stefan problem with triple junctions, Computational Methods in Applied Mathematics, 26(1), 43–67, (2026).

[9] B. Hooshyarfarzin, M. Abbaszadeh, and M. Dehghan, Numerical simulation and error estimation of the davey-stewartson equations with virtual element method, Applied Mathematics and Computation, 489, 129146, (2025).

[10] K. Soga, Finite difference methods for linear transport equations with sobolev velocity fields, Journal of Mathematical Fluid Mechanics, 27(1), 1–21, (2025).

[11] N. Magome, N. Morita, S. Kaneko, and N. Mitsume, Higher-continuity s-version of finite element method with b-spline functions, 497, Journal of Computational Physics, 497, 112593, (2024).

[12] C.-B. Zhou, Q. Wang, and Y.-X. Ren, Machine learning optimization of compact finite volume methods on unstructured grids, Journal of Computational Physics, 500, 112746, (2024).

[13] M. Reichle, M. Klassen, J. Li, and S. Klinkel, A modified approach for a scaled boundary shell formulation in structural isogeometric analysis, Engineering Analysis with Boundary Elements, 159, 81–94, (2024).

[14] A. Pakniyat, K. Parand, and M. Jani, Least squares support vector regression for differential equations on unbounded domains, Chaos, Solitons & Fractals, 151, 111232, (2021).

[15] C. Pozrikidis, Finite and spectral element methods using matlab, University of California, San Diego, (2005).

[16] R. R. Faria, B. Capron, A. R. Secchi, and M. B. de Souza Jr, A data-driven tracking control framework using physics-informed neural networks and deep reinforcement learning for dynamical systems, Engineering Applications of Artificial Intelligence, 127, 107256, (2024).

[17] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics,378, 686–707, (2019).

[18] Y. Huang, H. Liu, Y. Zhao, and M. Fei, Chebyshev spectral approximation-based physics-informed neural network for solving higher-order nonlinear differential equations, Engineering with Computers, (2024).

[19] D. Stenkin and V. Gorbachenko, Mathematical modeling on a physics-informed radial basis function network, Mathematics, 12(2), 241, (2024).

[20] M. Movahedian Moghaddam, K. Parand, and S. R. Kheradpisheh, Advanced physics-informed neural network with residuals for solving complex integral equations, Analytical and Numerical Solutions for Nonlinear Equations, 11(1), 153–173, (2026).

[21] S. Mehrkanoon, T. Falck, and J. A. Suykens, Approximate solutions to ordinary differential equations using least squares support vector machines, IEEE Transactions on Neural Networks and Learning Systems, 23(9), 1356–1367, (2012).

[22] K. Parand, 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, 114–128, (2021).

[23] J. A. Rad, K. Parand, and S. Chakraverty, Learning with Fractional Orthogonal Kernel Classifiers in Support Vector Machines: Theory, Algorithms and Applications. Cham, Switzerland: Springer, 2023.

[24] J. Sun, C. Liu, and N. Liu, Data-driven adaptive critic approach for nonlinear optimal control via least squares support vector machine, Asian Journal of Control, 20(1), 104–114, (2018).

[25] M. Bolhassani, H. Dana Mazraeh, and K. Parand, A new method based on least-squares support vector regression for solving optimal control problems, Kybernetika, 60(4), 513–534, (2024).

[26] H. Dana Mazraeh and K. Parand, GELS-SVR: a combination of grammatical evolution and least squares support vector regression for symbolic solution to Falkner–Skan equation, Genetic Programming and Evolvable Machines, 27(1), 3, (2026).

[27] M. Movahedian Moghaddam, H. Dana Mazraeh, and K. Parand, A double deep Q network-powered approach to discovering symbolic solutions for nonlinear integral equations, Engineering Applications of Artificial Intelligence, 169, 114113, (2026).

[28] M. Movahedian Moghaddam, H. Dana Mazraeh, and K. Parand, Symbolic solution of fractional Volterra population models via physics-informed neural networks and distributed particle swarm optimization, Applied Soft Computing, 189, 114512, (2026).

[29] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, Physics-informed machine learning, Nature Reviews Physics, 3(6), 422–440, (2021).

[30] K. Pelckmans, J. A. Suykens, T. Van Gestel, J. De Brabanter, L. Lukas, B. Hamers, B. De Moor, and J. Vandewalle, Ls-svmlab: a matlab/c toolbox for least squares support vector machines, Tutorial. KULeuven-ESAT. Leuven, Belgium, 142(1-2), (2002).

[31] S. K. Mitusch, S. W. Funke, and M. Kuchta, Hybrid fem-nn models: Combining artificial neural networks with the finite element method, ournal of Computational Physics, 446, 110651, (2021).

[32] J. Cen and Q. Zou, Deep finite volume method for partial differential equations, Journal of Computational Physics 517, 113307, (2024).

[33] M. Asadzadeh, An introduction to the finite element method for differential equations. John Wiley & Sons: John Wiley & Sons, 2020.

[34] K. M. Mohammadi, M. Babaei, Z. Hajimohammadi, and K. Parand, A new numerical method for solving neuro-cognitive models via chebyshev deep neural network (cdnn), Research Square, (2023).

[35] E. Babolian and J. Saeidian, Analytic approximate solutions to burgers, fisher, huxley equations and two combined forms of these equations, Communications in Nonlinear Science and Numerical Simulation, 14(5), 1984–1992, (2009).