FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 10 No. 12, June, 2026, pp 120 – 140
DOI: https://doi.org/10.33003/fjs-2026-1012-5570

Machine Learning-Assisted Numerical Integration for Predictive Mathematical Modelling

Opara Okwuchukwu Christabel

Department of Computer Science, Delta State University, Abraka, Delta State, Nigeria.

*Corresponding authors' email: oparachristabel@delsu.edu.ng

Received: 06th July 2026
Accepted: 16th July 2026
Published: 29th July 2026

ABSTRACT
Accurate numerical integration remains a fundamental challenge in predictive mathematical modelling, particularly for nonlinear, high-dimensional, and data-driven systems. Conventional numerical integration techniques, including the Trapezoidal Rule, Simpson's Rule, and Gaussian Quadrature, are widely used to approximate integrals that cannot be solved analytically. However, their performance often depends on integration parameters, data quality, and computational resources, particularly for nonlinear, high-dimensional, and uncertain systems. This study proposes a hybrid system that integrates machine learning with numerical integration to enhance prediction accuracy, computational efficiency, and numerical robustness. The designed system uses numerical integration to model continuous system dynamics while machine learning algorithms learn from historical and simulated datasets to select suitable integration methods, optimize step sizes, and estimate numerical errors. Performance was evaluated using MAE, RMSE, MAPE, convergence rate, computational time, stability analysis, robustness analysis, and statistical significance testing. It reduced approximation errors by over 80%, improved predictive accuracy to approximately 99%, and maintained numerical stability across benchmark problems. The results demonstrate that machine learning-assisted numerical integration significantly improves computational efficiency and predictive performance while preserving the convergence and stability properties of classical numerical methods. The study develops a novel adaptive machine learning-assisted numerical integration algorithm that automatically selects numerical integration strategies while preserving theoretical convergence and stability across science, engineering, finance, and environmental systems.

Keywords: Machine Learning, Numerical Integration, Adaptive Algorithms, Predictive Modeling, Numerical Stability,

INTRODUCTION

Accurate predictive mathematical modelling has become increasingly important in modern science, engineering, finance, healthcare, environmental management, and industrial systems, where complex phenomena must be analysed, simulated, and forecast with high precision. As these systems become more nonlinear, data-intensive, stochastic, and computationally demanding, there is a growing need for mathematical models capable of extracting meaningful information from large datasets while maintaining computational efficiency and numerical reliability. Predictive mathematical modelling provides the mathematical framework for representing real-world systems, describing interactions among variables, analysing system behaviour, and forecasting future outcomes under varying conditions. Consequently, predictive models have become indispensable tools in climate prediction, computational fluid dynamics, financial engineering, epidemiological modelling, energy systems, transportation, and intelligent decision-support systems (Karniadakis *et al*., 2021; Brunton *et al*., 2022). The effectiveness of these models, however, depends fundamentally on the numerical algorithms employed to solve the underlying mathematical equations, particularly the evaluation of definite integrals that frequently arise during model formulation, parameter estimation, optimization, and simulation.

Numerical integration remains one of the fundamental computational techniques in applied mathematics and scientific computing. It provides approximate solutions to definite integrals when analytical solutions are unavailable, computationally intractable, or impossible to obtain. Classical numerical integration methods including the Rectangle Rules, Trapezoidal Rule, Simpson's Rules, Gaussian Quadrature, Romberg Integration, Monte Carlo Integration, and Adaptive Quadrature have been widely adopted because of their mathematical rigor, convergence properties, and computational simplicity (Atkinson, 1989; Burden & Faires, 2016; Quarteroni *et al*., 2023). These techniques constitute the computational backbone of numerous applications, including the numerical solution of ordinary and partial differential equations, uncertainty quantification, Bayesian inference, stochastic simulation, optimization, and engineering design.

Despite their widespread success, conventional numerical integration methods exhibit several limitations when applied to contemporary predictive mathematical models. Their accuracy depends heavily on appropriate selection of integration parameters such as step size, quadrature order, and discretization strategy. Furthermore, approximation errors accumulate rapidly in highly nonlinear systems, oscillatory functions, high-dimensional integrals, and noisy datasets, resulting in increased computational cost and reduced predictive accuracy. Fixed numerical integration schemes are generally unable to adapt dynamically to varying characteristics of the integrand, often requiring repeated computations to achieve acceptable accuracy. Consequently, modern predictive mathematical models involving complex system dynamics frequently experience a trade-off between

numerical accuracy and computational efficiency (Cuomo *et al*., 2022; Kovachki *et al*., 2023).

Simultaneously, the rapid advancement of machine learning has transformed computational modelling by introducing algorithms capable of learning complex nonlinear relationships directly from data. Unlike conventional computational algorithms that follow predetermined numerical procedures, machine learning algorithms automatically identify hidden structures, discover functional relationships, estimate uncertainties, and improve predictive performance through experience. Significant advances in deep learning, probabilistic machine learning, reinforcement learning, and scientific machine learning have enabled data-driven approaches to address computational problems previously considered difficult or computationally prohibitive (Goodfellow *et al*., 2016; Jordan & Mitchell, 2015; Murphy, 2023). These developments have created new opportunities for integrating data-driven intelligence with classical numerical algorithms to improve computational performance without sacrificing mathematical rigor.

The convergence of machine learning and computational mathematics has led to the emergence of Scientific Machine Learning (SciML), an interdisciplinary field that combines mathematical modelling, numerical analysis, scientific computing, and artificial intelligence to improve the solution of complex scientific problems (Karniadakis *et al*., 2021). Rather than replacing established numerical algorithms, Scientific Machine Learning seeks to enhance them by embedding learning capabilities within traditional computational frameworks. Recent developments such as Physics-Informed Neural Networks (PINNs), Neural Operators, Deep Operator Networks (DeepONets), Gaussian Process Regression, and Universal Differential Equations demonstrate that machine learning can effectively complement mathematical models while preserving important physical laws and numerical properties (Raissi *et al.,* 2019; Lu *et al*., 2021; Rackauckas *et al*., 2023).Among the emerging research directions within Scientific Machine Learning is Machine Learning-Assisted Numerical Integration (MLANI), which integrates machine learning algorithms with classical numerical integration techniques to improve approximation accuracy, computational efficiency, and adaptive decision-making during integration. Instead of relying solely on predefined numerical rules, MLANI enables computational systems to learn characteristics of mathematical functions from historical or simulated data, predict optimal integration strategies, estimate local truncation errors, determine adaptive step sizes, and recommend appropriate numerical integration methods based on the complexity of the integrand. By incorporating learning mechanisms into the numerical integration process, the framework transforms integration from a static computational procedure into an adaptive and intelligent numerical process capable of responding dynamically to varying mathematical conditions.

Recent advances in artificial neural networks, Gaussian Process Regression, decision tree learning, ensemble methods, Physics-Informed Neural Networks, and Neural Operators have demonstrated considerable success in approximating nonlinear functions, modelling high-dimensional systems, and accelerating scientific computations (Cuomo *et al*., 2022; Kovachki *et al*., 2023). These algorithms possess strong approximation capabilities that make them particularly suitable for enhancing numerical integration by predicting integration errors, learning correction terms, estimating optimal discretization parameters, and improving convergence behaviour. Consequently, machine learning-assisted numerical integration offers significant potential for solving challenging predictive mathematical modelling problems involving nonlinear differential equations, stochastic systems, uncertainty quantification, financial forecasting, environmental modelling, computational fluid dynamics, and intelligent engineering systems.

Despite these promising developments, a significant research gap remains. Existing studies in Scientific Machine Learning have primarily focused on solving differential equations, surrogate modelling, inverse problems, and physics-informed learning. Comparatively limited attention has been devoted to the systematic integration of machine learning with classical numerical integration methods for predictive mathematical modelling. Existing approaches rarely provide a unified computational framework capable of simultaneously selecting appropriate numerical integration methods, optimizing integration parameters, estimating numerical errors, and preserving the theoretical convergence and stability properties of traditional numerical algorithms. Furthermore, many existing machine learning approaches operate as black-box predictors with limited interpretability and weak integration with established numerical analysis principles. Consequently, there remains a need for a robust, mathematically consistent, and computationally efficient hybrid framework capable of combining the predictive strengths of machine learning with the reliability of classical numerical integration.

Motivated by this research gap, this research proposes a Machine Learning-Assisted Numerical Integration system for Predictive Mathematical Modelling. The proposed system formulates numerical integration as a supervised learning optimization problem in which machine learning algorithms analyse characteristics of mathematical functions to predict suitable numerical integration methods, optimize adaptive step sizes, estimate numerical approximation errors, and improve computational efficiency while maintaining theoretical convergence and numerical stability.

The proposed system is evaluated using both simulated benchmark mathematical functions and real-world predictive modelling applications. Performance is assessed through numerical accuracy, convergence behaviour, computational efficiency, robustness under noisy and high-dimensional conditions, and statistical significance analyses using established performance metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), coefficient of determination (R^2), computational time, and convergence rate. By combining mathematical rigor with data-driven intelligence, the proposed system seeks to improve numerical integration beyond the capabilities of conventional techniques while preserving the theoretical foundations of numerical analysis.

This research contributes to the rapidly growing field of Scientific Machine Learning by developing a novel adaptive machine learning-assisted numerical integration system that bridges applied mathematics, numerical analysis, scientific computing, and artificial intelligence. The study not only advances the theory of intelligent numerical integration but also provides practical computational tools for solving complex predictive mathematical modelling problems across finance, engineering, environmental science, healthcare, and other data-intensive disciplines. It establishes a foundation for next-generation adaptive computational methods capable of supporting intelligent scientific discovery and decision-making in increasingly complex real-world systems.

Theoretical Foundations of Numerical Integration

Numerical integration, commonly referred to as quadrature, is a branch of numerical analysis concerned with approximating definite integrals of the form

$I = \int_a^b f(x)dx$

when closed-form analytical solutions are unavailable or computationally intractable. It forms the computational foundation of numerous applications in scientific computing, optimization, uncertainty quantification, finite element analysis, and predictive mathematical modelling (Quarteroni *et al*., 2023).

The overall idea is to approximate the exact integral value by substituting a weighted sum of function evaluations at specific places within the integration domain for a continuous integral. Several numerical integration techniques have been developed over time, each offering different levels of computational efficiency and accuracy depending on the characteristics of the integrand and the problem domain.

i. Trapezoidal Rule

One of the most basic and popular techniques for numerical integration is the Trapezoidal Rule. By splitting the integration interval into smaller intervals and using a trapezoid to symbolize each segment, it approximates the area under a curve. The total of these trapezoids' areas is then used to approximate the integral. The technique is still useful for smooth functions despite its simplicity, and it serves as the foundation for numerous adaptive integration algorithms that are employed in contemporary scientific computing (Higham & Higham, 2021). In general, the more subintervals there are, the more accurate the trapezoidal rule is. The following is the formula:

Trapezoidal Rule = $^{h}/_{2}$ [f(x_o) + 2 $\sum_{i=1}^{n-i} f(xi) + f(xn)$]

ii. Simpson's Rule

By approximating the integrand with second-degree (quadratic) polynomials instead of straight-line segments, Simpson's Rule improves integration accuracy. For smooth functions with comparatively low computing cost, Simpson's Rule usually provides more precision than the trapezoidal technique by fitting parabolic curves through specific data points. Simpson's Rule is still acknowledged in the numerical analysis literature as an effective method for moderate-dimensional integration problems when function evaluations are not unduly costly (Quarteroni *et al*., 2023). The following is the formula:

Simpson's rule = $\frac{h}{3}$ [f(x_o) + $4\sum_{i=1,3,5}^{n-1} f(xi)$ + $2\sum_{i=2,4,6}^{n-2} f(xi) + f(xn)$]

iii. Gaussian Quadrature

A more complex integration method called Gaussian Quadrature carefully chooses integration sites (nodes) and matching weights to optimize approximation accuracy. In contrast to equally spaced techniques like the trapezoidal and Simpson's rules, Gaussian Quadrature uses orthogonal polynomial theory to establish optimal evaluation locations, enabling precise integration of polynomials up to a certain degree with very few function evaluations. Because of its efficiency, it is especially useful in computational physics applications, finite element analysis, and engineering simulations (Trefethen & Bau, 2022). Gaussian quadrature variations are still among the most precise deterministic integration techniques. The following is the formula:

Gaussian Quadrature = $\sum_{i=1}^{n} Wif(xi)$

Monte Carlo Integration

Monte Carlo Integration is especially helpful for high-dimensional problems where typical deterministic approaches become computationally impractical. It uses random sampling techniques to estimate integral values. By creating random samples from the integration domain and calculating the average function value, the approach approximates integrals. Its convergence rate is essentially independent of dimensionality, which is one of its main features. This makes it ideal for applications including stochastic modelling, financial risk analysis, uncertainty quantification, and Bayesian inference (Owen, 2023). Recent developments have improved Monte Carlo methods even more by using machine learning-based sampling algorithms and variance reduction approaches. The following is the formula:

Monte Carlo integration = $^{h}/_{2}$ $\sum_{i=1}^{n} f(Yi)$

Challenges in Numerical Integration

When applied to modern predictive mathematical models with large datasets, nonlinear relationships, stochastic behaviour, and high-dimensional state spaces, traditional numerical integration techniques frequently face significant difficulties despite being mathematically rigorous and having shown notable success across a wide range of applications. In order to maintain acceptable levels of accuracy, numerical integration processes may need substantial computational resources, a large number of function evaluations, and complex adaptive strategies as model complexity rises (Karniadakis *et al*., 2021; Rackauckas *et al*., 2023).

Furthermore, integration tasks that exceed the practical capabilities of traditional quadrature methods are often encountered in new applications across scientific machine learning, digital twins, computational fluid dynamics, epidemiological forecasting, financial engineering, and climate modeling. Due to these constraints, interest in hybrid computational frameworks—which combine conventional numerical techniques with machine learning algorithms that can learn integration patterns, approximate complex integrands, estimate numerical errors, and optimize computational performance—has grown (Kovachki *et al*., 2023).

As a result, a promising field of study within Scientific Machine Learning (SciML) is Machine Learning-Assisted Numerical Integration. These methods aim to enhance accuracy, scalability, adaptability, and computational efficiency in resolving challenging predictive mathematical modelling issues by utilizing data-driven learning capabilities in conjunction with well-established numerical principles. (Brunton *et al*., 2022; Karniadakis *et al*., 2021).

Literature Review

Leung *et al*. (2013) proposed a novel multi-staged strategy for predicting genotype-phenotype risk patterns in diabetic kidney disease (DKD). They integrated ML and mathematical modelling; the approach involved data collection, ML analysis, and mathematical modelling. The study aimed to enhance predictive accuracy and uncover nuanced relationships. The proposed strategy holds promise for personalized medicine in DKD, offering insights into risk stratification and informing targeted interventions for individuals based on their genetic predispositions and phenotypic characteristics. The multi-disciplinary approach exemplifies a comprehensive strategy for advancing our understanding and prediction capabilities in DKD.

Dutta *et al*. (2018) proposed Mathematical Models for Classification Algorithms in ML: A Comprehensive Review. They systematically explored the landscape of mathematical models underpinning classification algorithms. They

surveyed the theoretical foundations of classification and elucidated the mathematical principles behind algorithms like decision trees, support vector machines, and neural networks. By providing a comprehensive review, their paper facilitated a deeper understanding of the mathematical underpinnings, guiding researchers and practitioners in optimizing and interpreting classification models. It serves as a valuable resource in the pursuit of refining mathematical frameworks, ultimately enhancing the efficacy and interpretability of classification algorithms in the realm of ML.

Göçmen and Erol (2019) studied Transportation Problems in Intermodal Networks: Mathematical Models, Exact and Heuristic Algorithms, and ML Approaches. Their paper provided a comprehensive exploration of addressing transportation challenges in intermodal networks. It integrated mathematical models for problem formulation, exact algorithms like Integer Linear Programming and Branch and Bound, heuristic algorithms including metaheuristic approaches, and ML applications. By combining these methodologies, the study aimed to optimize routing, scheduling, and capacity planning in intermodal transportation. The multi-faceted approach offers a robust framework for efficient and adaptive solutions, contributing to advancements in logistics and transportation management within the dynamic context of intermodal networks.

Bolte and Pauwels (2020) researched the foundations of Automatic Differentiation: A Mathematical Model for Machine Learning. They established a robust framework for understanding automatic differentiation (AD) in the context of Machine Learning (ML). The paper delved into the mathematical foundations of AD, elucidating its role in optimizing models through gradient-based methods. By presenting a comprehensive mathematical model, it demystifies the intricacies of AD, providing a clear pathway for its application in ML algorithms. The paper's insights contribute to enhancing the efficiency and effectiveness of gradient-based optimization, underscoring the pivotal role of mathematical modelling in advancing the foundations of automatic differentiation within the ML domain.

Boso *et al.* (2020) proposed advancements in drug delivery that showcased cutting-edge progress by integrating experiments, mathematical modelling, and ML. This multidisciplinary approach optimizes drug delivery systems, utilizing real-world experimentation, theoretical modelling, and adaptive ML. The synergy of these methodologies promises innovative solutions for precision drug delivery, improving therapeutic outcomes.

Clement *et al*. (2021) surveyed the mathematical, ML, and deep learning models for COVID-19 transmission and diagnosis by synthesizing advancements in these domains. Their paper offered valuable insights into diverse approaches to combating the pandemic. The collective analysis informs ongoing efforts to understand, model, and diagnose COVID-19 for improved public health responses.

Ramchandani *et al*. (2021) stated that COVID-19 has claimed 4.1 million lives, as of July 24, 2021. A variety of ML models have been applied to related data to predict important factors such as the severity of the disease and infection rate, and discover important prognostic factors. Often, the usefulness of the findings from the use of these techniques is reduced due to a lack of method interpretability. Some recent progress made on the interpretability of ML models has the potential to unravel more insights while using conventional ML models. In this work, they analysed COVID-19 blood work data with some of the popular ML models; then they employed state-of-the-art post-hoc local interpretability techniques (e.g., SHAP, LIME), and global interpretability techniques (example is symbolic metamodeling) to the trained black-box models to draw interpretable conclusions. In the gamut of ML algorithms, regressions remain one of the simplest and most explainable models with a clear mathematical formulation. They explored one of the most recent techniques called symbolic metamodeling to find the mathematical expression of the ML models for COVID-19. They identify Acute Kidney Injury (AKI), initial Albumin level (ALB I), Aspartate aminotransferase (AST I), Total Bilirubin initial (TBILI), and D-Dimer initial (DIMER) as major prognostic factors of the disease severity. Their contributions are: (i) uncover the underlying mathematical expression for the black-box models on the COVID-19 severity prediction task, (ii) they are the first to apply symbolic metamodeling to this task, and (iii) discover important features and feature interactions.

Román *et al.* (2021) analysed the temperature and strain-rate-dependent response of polypropylene. A hybrid modelling approach was taken by combining mechanism-based and data-based modelling. The "big data" required for ML is generated using a custom-made robot-assisted testing system. The neural network model was employed in series with a temperature-dependent spring to describe the stress-strain response of polypropylene.

Aarathi and Vasundra (2022) proposed Predicting Arrhythmia Susceptibility. Innovatively merged mathematical model and ML for precise risk forecasts. This fusion combined mechanistic insights with data-driven adaptability, exemplifying a powerful approach in arrhythmia research. Their study aimed to enhance predictive accuracy, showcasing the collaborative potential of mathematical modelling and ML in biomedical applications.

Cai *et al*. (2023) studied the Synergy of Mathematical Modelling and ML: A Comprehensive Review. They navigated the intersection of these disciplines, providing a nuanced exploration of their collaborative potential. Their research emphasized the synergistic integration of mathematical modelling theoretical foundation with Machine Learning data-driven adaptability. Applications across healthcare, finance, and diverse domains showcase the transformative impact of hybrid models. While addressing challenges in interpretability and ethical considerations, the review underscores ongoing interdisciplinary collaboration and educational initiatives. It encapsulated the collective power of mathematical modelling and Machine Learning, offered a comprehensive insight into their combined contributions to advancing knowledge and solving complex real-world problems.

Jacob Colton (2024) in his work 'Integrating Machine Learning with Mathematical Models: A New Paradigm' explained that the convergence of Machine Learning (ML) and mathematical modelling is transforming fields ranging from engineering to environmental sciences. He explores how integrating ML with traditional mathematical models enhances predictive accuracy, uncovers hidden patterns and drives innovation across various domains. However, he stated that the increasing complexity of real-world systems often challenges the limitations of traditional models. Machine learning, with its ability to handle vast datasets and uncover patterns beyond human intuition, offers a promising complement to mathematical modelling. This synergy, often referred to as the integration of ML with mathematical models, promises to advance scientific research and practical applications.

Leitao and R´afales (2025) in their work: Parametric Numerical Integration with (Differential) Machine Learning, introduced a machine/deep learning methodology to solve

parametric integrals. Besides classical machine learning approaches, they considered a differential learning frame work that incorporates derivative information during training, emphasizing its advantageous properties.

MATERIALS AND METHODS

In order to improve predictive mathematical modelling, this study uses a hybrid computational research methodology that combines cutting-edge machine learning algorithms with conventional numerical integration techniques. A key concept in the field of Scientific Machine Learning (SciML) is the hybridization of data-driven learning techniques with traditional numerical methods, which offers enhanced accuracy, computational efficiency, and adaptability in solving challenging mathematical and engineering problems (Karniadakis *et al*., 2021; Brunton *et al*., 2022).

Using a quantitative and experimental research strategy, this paper evaluates traditional numerical integration methods through computational experiments and adds machine learning models. The goal of the experimental framework is to evaluate how well machine learning-assisted integration techniques can enhance prediction accuracy, lower numerical errors, maximize computational efficiency, and adjust to intricate nonlinear datasets. In computer science and artificial intelligence research, these experimental methods are frequently employed to verify the efficacy of hybrid algorithms under various data and modelling settings. (Murphy, 2023; Goodfellow *et al.*, 2023).

In particular, the approach compares the performance of machine learning-enhanced integration frameworks versus conventional numerical integration techniques using predetermined assessment metrics. In order to increase approximation accuracy, evaluate integration errors, optimize integration parameters, and discover underlying functional correlations, machine learning models are trained. The efficacy of the generated predictive models in managing nonlinear dynamics and complex system behaviors is then evaluated using simulated and benchmark datasets (Rackauckas *et al*., 2023; Kovachki *et al*., 2023).

Recent developments in Scientific Machine Learning, which emphasize the integration of mathematical principles, numerical analysis, and artificial intelligence to develop next-generation predictive modelling systems capable of addressing challenges associated with large-scale data, uncertainty, and computational complexity, are driving the adoption of this hybrid methodology (Karniadakis *et al*., 2021; Cuomo *et al.*, 2022). Consequently, the methodological framework provides a systematic basis for evaluating the potential of machine learning-assisted numerical integration as a tool for enhancing predictive mathematical modelling across diverse application domains.

Predictive Mathematical Modelling

The practice of creating mathematical models of real-world systems to explain observed events and predict future outcomes is known as predictive mathematical modelling. To forecast how complex systems will behave under various circumstances, it integrates mathematical theory, statistical analysis, computational methods, and increasingly, machine learning algorithms. To aid in decision-making and the creation of policies, predictive models are extensively used in the fields of economics, engineering, healthcare, climate science, finance, epidemiology, transportation, and industrial automation (Murphy, 2023; Karniadakis *et al*., 2021).

Establishing correlations between variables and using these relationships to anticipate future states of a system is the core goal of predictive mathematical modelling. Predictive models have historically mostly depended on statistical methods and deterministic mathematical formulas. However, the emergence of machine learning has significantly expanded the predictive capabilities of mathematical models by enabling them to learn complex nonlinear relationships directly from data while incorporating scientific and physical knowledge (Brunton *et al*., 2022).

Since many real-world systems are governed by differential equations whose solutions require evaluating integrals, numerical integration is essential to predictive modelling. By identifying underlying patterns in the data, improving integration parameters, predicting numerical errors, and modifying computing techniques in response to shifting system dynamics, machine learning-assisted numerical integration enhances forecasting. These hybrid approaches have shown impressive results in energy demand modelling, financial risk assessment, disease outbreak forecasting, climate prediction, and engineering simulations (Willard *et al*., 2022; Rackauckas *et al.*, 2023).

For instance, by integrating real-time data and lowering numerical approximation errors, machine learning-enhanced integration techniques might improve disease transmission rate forecasts in epidemiological modelling. They aid in the simulation of oceanic and atmospheric phenomena with highly nonlinear interactions in climate science. In finance, hybrid models enhance the prediction of asset prices, market volatility, and portfolio risks by combining mathematical models with data-driven learning techniques (Rolnick *et al*., 2022; Murphy, 2023).

Recent developments in Machine Learning have further strengthened predictive mathematical modelling by integrating machine learning algorithms with physical laws and mathematical constraints. Physics-Informed Neural Networks (PINNs), Deep Operator Networks (DeepONets), Neural Operators, and Universal Differential Equations are examples of modern approaches that combine data-driven learning with numerical computation to improve predictive accuracy and computational efficiency (Karniadakis *et al*., 2021; Lu *et al*., 2021; Kovachki *et al.*, 2023).

The Mathematical

General Predictive Mathematical Model

Assume a differential equation represents a system:

$(dy(t))/dt = f(t, y, \theta)$

Where:

t = time

y(t) = state variable

θ = model parameters

f(·) = governing physical or mathematical process

Initially,

$y(t0) = y0$

Putting equation (1) into equation (2), we will get:

$y(t) = y0 + \int_{to}^{t} f(s,y(s),\theta)\, ds$

Numerical Integration Formulation

Numerical methods are approximated as:

$I = \int_a^b f(x)dx$

Trapezoidal Rule

Trapezoidal Rule = $h/2\ [f(xo) + 2 \sum_{(i=1)}^{(n-i)} [f(x_i)+f(x_n)]]$

Where:

h = (b-a)/n

Simpson's Rule

Simpson's rule = h/3 [f(xo) + 4∑_(i=1,3,5)^(n-1)▒f(x_i) +2∑_(i=2,4,6)^(n-2)▒[f(x_i)+f(x_n)]]

Gaussian Quadrature
Gaussian Quadrature = ∑_(i=1)^n▒Wif(xi)
Where
wi = weights
xi = Gaussian nodes

Numerical Error
Traditional integration produces
E = I − Ih
Where:
I = exact integral
Ih = numerical approximation

Machine Learning-Assisted Error Prediction
Instead of accepting this numerical error, machine learning estimates it. Then defines error as:
$\hat{E} = ML(X)$
where
X ={h, f(x), f′(x), f′′(x), curvature, interval size}
The machine learning model predicts $\hat{E}$
The corrected integral becomes
$I_{ML} = I_h + \hat{E}$
This is the main contribution of this study.

Predictive Mathematical Model
$\hat{y} = F(x, \theta)$
Where F may be a neural network, random forest, XGBoost or support vector regression.
The major equation becomes:
$\frac{dy}{dt} = F(t, y, \theta)$
This will give us:
$y(t) = y_0 + \int_{to}^{t} F(s, y, \theta)\, ds$

Hybrid Machine Learning Numerical Integration
This study creates:
$I = I_h + M\,L\,(I_h, h, f)$
Generally,
$I = N\,(f, h) + M\,(X)$
Where:
N = classical numerical integration method and
M = machine learning correction model.

Neural Network Formulation
Input:
X = [h, f(x), f′(x), f′′(x)]′(x)]
Hidden layers
$z^{(1)} = W^{(1)}X + b^{(1)}$
This will result in:
$a^{(1)}=\sigma(z^{(1)})$
Output:
$\hat{E} = W^{(2)}a^{(1)} + b^{(2)}$
$\hat{E} = ML(X)$
Where
X ={h, f(x), f′(x), f′′(x), curvature, interval size}
The machine learning model predicts $\hat{E}$
The corrected integral becomes
$I_{ML} = I_h + \hat{E}$
This is the main contribution of this study.

Predictive Mathematical Model
$\hat{y} = F(x, \theta)$
Where F may be a neural network, random forest, XGBoost or support vector regression.
The major equation becomes:
$\frac{dy}{dt} = F(t, y, \theta)$
This will give us:
$y(t) = y_0 + \int_{to}^{t} F(s, y, \theta)\, ds$

Hybrid Machine Learning Numerical Integration
This study creates:
$I = I_h + M\,L\,(I_h, h, f)$
Generally,
$I = N\,(f, h) + M\,(X)$
Where:
N = classical numerical integration method and
M = machine learning correction model.

Neural Network Formulation
Input:
X = [h, f(x), f′(x), f′′(x)]′(x)]
Hidden layers
$z^{(1)} = W^{(1)}X + b^{(1)}$
This will result in:
$a^{(1)}=\sigma(z^{(1)})$
Output:
$\hat{E} = W^{(2)}a^{(1)} + b^{(2)}$
Corrected integral
$I_{ML} = I_h + \hat{E}$

Loss Function
Mean Squared Error
$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}(E_i - \hat{E}_i)^2$
Where:
Ei = true numerical error
Êi = predicted error

Optimization
Parameters are updated using Gradient Descent
$\theta_{k+1} = \theta_k - \eta\nabla_\theta L$
or
Adam optimizer
$m_t = \beta_{1mt-1} + (1-\beta_1)_{gt}$
$v_t = \beta_{2vt-1} + (1-\beta_2)_{g2t}$
$\theta_{t+1} = \theta_t - \eta\,\frac{mt}{\sqrt{\hat{v}t}+\varepsilon}$

Prediction Framework
The overall predictive system can be summarized as
Data → Feature Extraction → Machine Learning → Estimated Function → Numerical Integration → Prediction
Mathematically, it is represented as
$X \rightarrow F_\theta(X) \rightarrow \hat{f}(x) \rightarrow \int_a^b \hat{f}(x)\,dx \rightarrow \hat{y}$

Performance Metrics
The model can be evaluated using:

Root Mean Square Error (RMSE)
$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$

Mean Absolute Error (MAE)
$MAE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)$

Coefficient of Determination
$R^2 = 1 - \sum(y_i - \hat{y}_i)^2 / \sum(y_i - \bar{y})^2$

Relative Integration Error

RE = $\frac{[I-IML]}{[I]}$ ×100%

Proposed Novel Mathematical Framework

A concise formulation that captures this research is:

$\hat{y}(t) = y_o + \int_{to}^{t} F(s,y,\theta)\, ds + M(X;\, \emptyset)$

Where:

f(s, y, θ) is the underlying mathematical model,

$\int_{to}^{t} F\, ds$ is approximated using a classical numerical integration method,

M(X; ϕ) is a trained machine learning model that estimates and corrects the integration error,

Φ denotes the machine learning parameters learned from data.

The Hybrid Algorithm

i. Input Mathematical Model
ii. Data Preprocessing
iii. Feature Extraction
iv. Machine Learning Predictor
v. Predict Best Numerical Integrator
vi. Adaptive Numerical Integration
vii. Error Estimation
viii. Error Acceptable?
ix. If yes
x. Prediction Output
xi. If no
xii. Retrain ML & Refine Step Size

Algorithmic Workflow

Input differential equation
Extract numerical features
Normalize features
Feed features into trained neural network
Predict:
Best numerical integrator
Initial step size
Expected numerical error
Execute selected numerical integration algorithm
Compute actual numerical error
If error > tolerance:
Refine step size
Repeat integration
Else:
Accept numerical solution
Store:
Features
Selected integrator
Step size
Actual error
Periodically retrain machine learning model
Return optimized numerical solution

Input Mathematical Model

Assume the predictive model is

$\frac{dy}{dt}$ = f (t, y, θ)

Where

y = dependent variable
t = independent variable
θ = model parameters

Instead of integrating immediately, the algorithm first analyses the equation. Examples of problems include population growth, heat transfer, disease prediction, financial forecasting, fluid dynamics, and neural dynamics.

Data Preprocessing

Here you preprocess the data.

Feature Extraction

Here you extract characteristics of the problem.

X = h, $\frac{\partial f}{\partial y}$, $\frac{\partial f}{\partial y} * 2$, stiffness, smoothness, oscillation, condition number

These describe the behaviour of the differential equation.

Machine Learning Predictor

Train any of these ML model: Random Forest, XGBoost, Artificial Neural Network, Deep Neural Network or Transformer.

Input
X
Output
Optimal integration method
The ML model learns
M: X → I
Where
I is the predicted numerical integration algorithm.

Predict Best Numerical Integration

Among the fundamental numerical integration methods, you will predict the best one to use.

Adaptive Numerical Integration

Suppose ML predicts the Trapezoidal Rule

$^{h}/_{2}$ [f(x_o) + 2 $\sum_{i=1}^{n-i} f(x_i) + f(x_n)$]

Where

h = $\frac{b-a}{n}$

If Simpson's rule is selected:

$\frac{h}{3}$ [f(x_o) + 4$\sum_{i=1,3,5}^{n-1} f(x_i)$ +2$\sum_{i=2,4,6}^{n-2} f(x_i) + f(x_n)$]

If Gaussian Quadrature is selected:

$\int_a^b f\,(x)dx \approx \sum_{i=1}^{n} w_i f(x_i)$

Where

w_i are Gaussian weights.

Error Estimation

Instead of estimating error only numerically, a second ML model predicts integration error.

Input
(X, I)
Output
Predicted error
$\hat{e}$ = g(X, I)
Where g is another neural network.
The new numerical error becomes additional training data.
Dataset grows
D = D ∪ (X, I, e)
The ML model continuously improves.
This creates
Numerical Integration → ML Learning → Better Numerical Integration

Refine Step Size

If
$\hat{e} > \varepsilon$
Reduce step size
$h_{new} = 0.9h(\frac{\varepsilon}{\hat{e}})^{1/5}$
Otherwise
Increase
$h_{new} = 1.1h$

Prediction Output

The hybrid algorithm solves
$Yn+_1 = N (M (X), f, h, y_n)$
Where
M (X) predicts the optimal numerical integration method.
N performs the selected integration.

Pseudocode

i. Input:
Differential equation
Initial condition
Tolerance ε
ii. Generate problem features, X
iii. Load trained ML model
iv. Predict best numerical integrator I
While t < T
v. Integrate using I
vi. Estimate numerical error e
vii. Predict error ê if ê > ε
viii. Reduce step size
ix. Repeat integration else
x. Accept solution
xi. Store (X, I, e)
xii. Update ML model periodically end while
xiii. Return numerical solution

The Proposed Hybrid Algorithm has the following uniqueness

Intelligent Integrator Selection

Machine learning automatically selects the most appropriate numerical integration method based on the characteristics of the mathematical model, rather than using a fixed method.

Machine Learning-Based Error Prediction

i. A trained model estimates numerical integration error before excessive computation, enabling proactive control of accuracy.

ii.

iii. ***Adaptive Step-Size Optimization***

iv. Step sizes are adjusted using both classical error-control formulas and ML predictions, improving computational efficiency while maintaining accuracy.

v.

vi. ***Continuous Self-Learning***

vii. The algorithm expands its training dataset with new integration results, allowing performance to improve over time across different classes of predictive mathematical models.

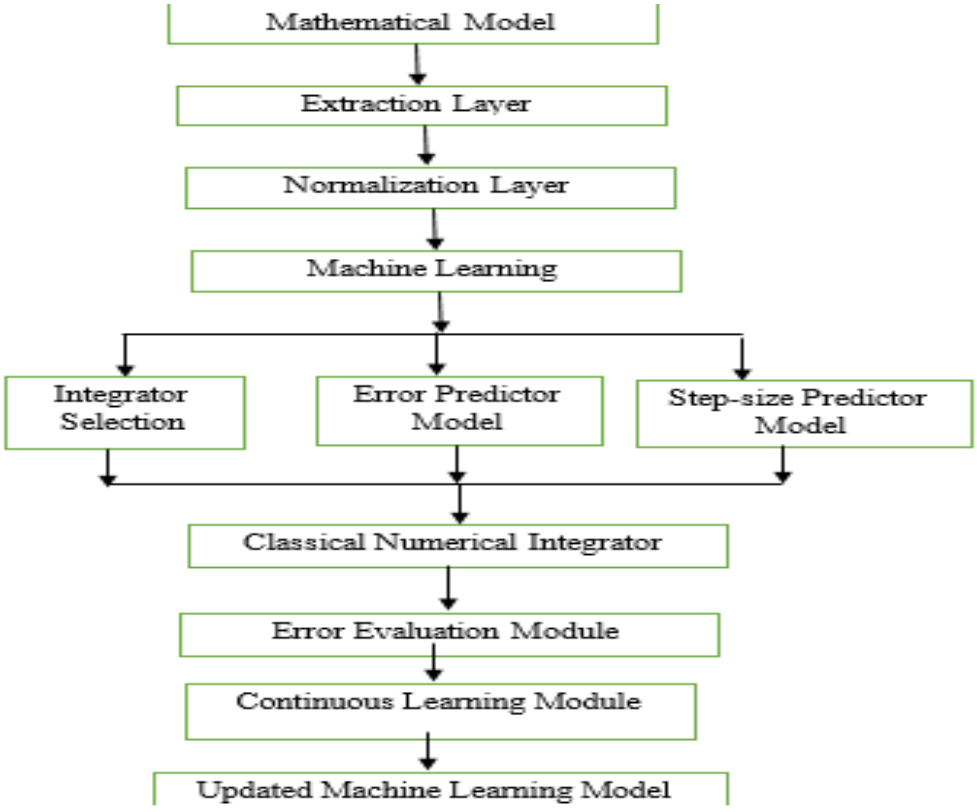
viii.

ix. ***Preserved Mathematical Reliability***

x. Classical numerical integration performs the actual integration, ensuring convergence, consistency, and stability, while machine learning acts as an intelligent decision-support layer rather than replacing established numerical methods.

Learning Architecture of the Proposed Hybrid Algorithm

The learning architecture is the intelligent core of the hybrid algorithm. Rather than directly performing numerical integration, it learns from previously solved mathematical problems to predict the most suitable numerical integration strategy, estimate the expected integration error, and recommend an optimal step size. This architecture transforms numerical integration into a data-driven, adaptive process while preserving the accuracy of classical numerical methods. The learning architecture consists of eight interconnected layers.



The Learning Architecture has the Following Uniqueness

The proposed learning architecture is novel because it uses multi-task learning to jointly optimize three tightly related decisions: selecting the numerical integration algorithm, predicting integration error, and recommending an adaptive step size. Unlike traditional adaptive numerical integrators that rely solely on local error estimators, this framework leverages patterns learned from previously solved mathematical models. By combining a shared neural network with classical numerical integration, the architecture can improve computational efficiency while preserving numerical stability, convergence, and accuracy. This integration of data-driven decision-making with established numerical analysis provides a strong methodological contribution suitable for doctoral research and opens opportunities for applications in predictive modelling across engineering, physics, finance, epidemiology, and computational science.

Model Training Procedure

The quality of the model training process has a major impact on how well machine learning-assisted prediction models work. To enable a machine learning algorithm to generate precise predictions on unobserved data, model training entails teaching it to recognize patterns and relationships within a dataset. A methodical approach comprising data preparation, model training, optimization, validation, and testing stages is advised by machine learning research (James *et al.*, 2021; Murphy, 2023).

Data Splitting

The first stage involves dividing the dataset into separate subsets:

i. Training Set: Used for learning model parameters.
ii. Validation Set: Used for tuning model settings and preventing overfitting.
iii. Testing Set: Used for evaluating final model performance.

A common practice is to allocate approximately 70% of the data for training, 15% for validation, and 15% for testing. Data splitting ensures that the model is evaluated on previously unseen data, thereby providing a realistic assessment of predictive performance (James *et al.*, 2021).

Model Training

During training, the machine learning algorithm learns relationships between input variables and target outputs. The objective is to minimize prediction errors through iterative optimization procedures.

Depending on the application, various algorithms may be employed, including:

i. Artificial Neural Networks (ANNs)
ii. Deep Neural Networks (DNNs)
iii. Random Forests
iv. Gradient Boosting Machines
v. Support Vector Machines
vi. Gaussian Process Regression
vii. Physics-Informed Neural Networks

The selected model learns patterns embedded in historical or simulated data and uses this knowledge to generate predictions for future observations (Murphy, 2023).

Hyperparameter Optimization

Hyperparameters are configuration settings that control how a machine learning algorithm learns. Examples include learning rates, tree depth, batch size, and network architecture.

To identify optimal hyperparameter values, researchers employ several search strategies:

Grid Search

Grid Search systematically evaluates all possible combinations of predefined parameter values.

Random Search

Random Search selects parameter combinations randomly from the search space, often achieving comparable performance with lower computational cost.

Bayesian Optimization

Bayesian Optimization utilizes probabilistic models to identify promising parameter combinations while minimizing the number of evaluations required. It is particularly useful in computationally expensive modelling tasks (Snoek *et al.*, 2012).

Hyperparameter optimization significantly improves model accuracy, robustness, and generalization capability.

Validation

The validation phase assesses model performance during training using the validation dataset. This stage helps detect problems such as:

i. Overfitting
ii. Underfitting
iii. Model instability
iv. Poor generalization

Cross-validation techniques are frequently employed to obtain reliable performance estimates and improve model robustness (James *et al.*, 2021).

Testing

The final testing phase evaluates model performance using previously unseen data. This step provides an unbiased assessment of how well the predictive model will perform in real-world applications.

Testing results are typically reported using standardized performance metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and the Coefficient of Determination (R^2).

The Training Equations for the Designed System

The training equations describe how the machine learning model learns from historical numerical integration data to improve its predictions. During training, the neural network receives numerical features extracted from mathematical models, predicts the optimal integration strategy, computes the loss, and updates its parameters using optimization.

Let the training dataset be

$D=\{(X_i, I_i, h_i, E_i)\}_{i=1}^{N}$

Where:

X_i = feature vector,
I_i = optimal numerical integration method,
h_i = optimal step size,
E_i = numerical integration error.

The normalized input vector is

$X=[x_1, x_2, \ldots, x_n]^T$

For hidden layer l,

$Z^{(l)} = W^{(l)}A^{(l-1)} + b^{(l)}$

Where:

$W^{(l)}$ = weight matrix,
$b^{(l)}$ = bias vector,
$A^{(l-1)}$ = previous layer output.

The activation is
$A^{(l)} = \sigma(Z^{(l)})$

Output Layer
The network has three outputs:
Integrator Selection
$\dot{P}$ = Softmax (ZI)

Step-size Prediction
$\hat{h} = W_h A + b_h$

Error Prediction
$\hat{E} = W_E A + b_E$

Loss Function
The total loss is
$L = \alpha L_{int} + \beta L_{step} + \gamma L_{error} + \delta L_{stab} + \lambda L_{cost}$

Dataset Update Equation
Each numerical integration produces a new training sample, (X, I, h, E)
The dataset evolves according to
$D_{t+1} = D_t \cup (X, I, h, E)$
This continual learning process allows the model to improve as more problems are solved.

Convergence Procedure of the Designed System
Convergence demonstrates that the designed hybrid algorithm produces numerical solutions that approach the exact solution as:

i. the integration step size decreases,
ii. the machine learning model becomes increasingly accurate through training, and
iii. the adaptive numerical integration process satisfies prescribed error tolerances.

Unlike classical numerical integration, where convergence depends only on the numerical scheme, the designed system has two interacting convergence processes:

i. Numerical convergence (convergence of the numerical integrator).
ii. Machine Learning convergence (convergence of the machine learning model).

The overall convergence is achieved only when both processes converge simultaneously.

Numerical Convergence
This is represented with I^*
The numerical approximation is
$y_{n+1} = N(I^*, h, y_n)$

Machine Learning Convergence
The neural network predicts
$F_\theta(X) = (\hat{I}, \hat{h}, \hat{E})$

Convergence of the Hybrid Algorithm
The selected numerical integrator satisfies.
$\lim_{n\to\infty} \left(\frac{y(t+h)-N(I,h,y)}{h} = 0\right)$
Since the machine learning model eventually predicts the optimal numerical method,
$P(I^* = I_{opt}) \to 1$
Therefore,
$N(I^*, h) \to N(I_{opt}, h)$

The complete algorithm can be expressed as
$Yn+_1 = N(F_\theta(X), h, y_n)$
As
$\theta \to \theta^*$
And
$h \to 0$
Which will result in
$y_n^{ML} \to y(t_n)$

Error Convergence
The machine learning model predicts
$\hat{E}$
The actual error is
E
The prediction loss is
$L_E = (\hat{E} - E)^2$
Training ensures
$L_E \to 0$
Therefore
$\hat{E} \to E$

The hybrid system is considered converged when all three conditions hold simultaneously:
$\|\theta_{k+1} - \theta_k\| < \varepsilon_1$
$|L_{k+1} - L_k| < \varepsilon_2$
$\|y(t_n) - y_n^{ML}\| < \varepsilon 3$

Convergence Algorithm
Initialize neural network
Initialize numerical integration
Repeat
Extract problem features
Predict numerical integrator
Predict step size
Integrate
Compute numerical error
If error > tolerance
Reduce step size
Else
Accept solution
Update neural network
Until
$||\theta (k+1) - \theta (k)|| < \varepsilon_1$
And
$|L (k+1) - L (k)| < \varepsilon_2$
And
$||y\_exact - y\_n|| < \varepsilon_3$

Optimization Procedure of the Designed Hybrid Algorithm
The optimization procedure is the decision-making mechanism that enables the designed hybrid algorithm to achieve three objectives simultaneously:

i. Maximize numerical accuracy,
ii. Minimize computational cost, and
iii. Maintain numerical stability.

Unlike numerical integration, where the integration method and step size are fixed or determined only by local error estimates, the proposed framework formulates the problem as a multi-objective optimization problem solved with the aid of machine learning.

Objective 1: Maximize Numerical Accuracy
Integration error is
$E(I, h) = y_{exact} - y_{numerical}$
The first optimization objective is to minimize integration error, which is presented as
Min E (I, h)
This ensures the numerical solution remains close to the exact solution.

Objective 2: Minimize Computational Cost
The computational cost is measured by CPU time, Number of function evaluations, and Memory usage. It is represented as: C(I, h)
The second objective becomes
Min C(I, h)

Objective 3: Maintain Numerical Stability
A numerical method must avoid divergence, unstable step sizes, and satisfy:
$R(z) \leq 1$
Where R(z) is the stability function.

The stability objective is
Max S (I, h)
Or equivalently
Min [−S (I, h)]
Where S measures numerical stability.

Combined Multi-Objective Optimization
The three objectives are combined into one weighted objective function
$J = \alpha E + \beta C - \gamma S$
Where:
α controls numerical accuracy,
β controls computational efficiency,
γ rewards stability.
The optimization problem becomes
Min J (I, h, θ)

Performance Evaluation Metrics
Evaluating predictive performance is essential for determining the effectiveness of machine learning-assisted numerical integration models. Several statistical measures are commonly employed in regression-based predictive modelling.

Mean Absolute Error (MAE)
Mean Absolute Error measures the average magnitude of prediction errors without considering their direction.
$MAE = \frac{1}{n}\sum_{i=n}^{n}(y_i - y)$
Where:
y_i= actual value
y = predicted value
n = number of observations
A smaller MAE indicates better predictive accuracy.

Root Mean Square Error (RMSE)
RMSE measures the square root of the average squared prediction errors.
$RMSE = \sqrt{\frac{1}{n}\sum_{i=n}^{n}(y_i - y)}$
RMSE penalizes large errors more heavily than MAE and is widely used in predictive modelling and forecasting studies (James *et al*., 2021).

Coefficient of Determination (R^2)
R^2 measures the proportion of variation in the dependent variable explained by the model.
$R^2 = 1 - \frac{\sum(y-y_i)2}{\sum(y-y_o)2}$
Where:
y_o is the mean of observed values.
An R^2 value closer to 1 indicates stronger predictive performance.

Simulation and Comparative Analysis
To evaluate the effectiveness of the proposed Machine Learning-Assisted Numerical Integration system, a comparative simulation study is conducted. Comparative analysis is a fundamental component of computational research because it provides empirical evidence regarding the strengths and limitations of alternative modelling approaches (Quarteroni *et al*., 2023).
The designed hybrid model is compared against the following methods:

Trapezoidal Rule
A classical numerical integration technique based on linear approximation.

Simpson's Rule
A higher-order numerical integration method based on quadratic interpolation.

Adaptive Quadrature
An advanced integration approach that automatically adjusts step sizes to improve accuracy.

Pure Machine Learning Models
Models relying entirely on data-driven learning without incorporating traditional numerical integration techniques.

Hybrid ML–Numerical Models
Models combining machine learning algorithms with numerical integration procedures to exploit the strengths of both approaches.
The comparative analysis focuses on prediction accuracy, numerical integration error, computational time, model stability, scalability and generalization capability. Such comparisons provide objective evidence regarding whether machine learning-assisted integration offers measurable improvements over conventional methods.

Expected Outcomes
The designed system is expected to produce a robust hybrid computational framework capable of enhancing predictive mathematical modelling through intelligent numerical integration.
Specifically, the model is expected to:
Improve Numerical Integration Accuracy
Machine learning algorithms can learn complex integrand behaviours and optimize integration procedures, thereby reducing approximation errors.

Reduce Computational Error
Adaptive learning mechanisms are expected to improve error estimation and correction throughout the integration process.

Enhance Predictive Modelling Performance
The integration of machine learning with numerical methods is anticipated to improve forecasting accuracy across diverse application domains.

Efficiently Handle Nonlinear and Large-Scale Datasets
The designed system should demonstrate superior performance in systems characterized by nonlinear interactions, uncertainty, and high-dimensional data.

Outperform Standalone Numerical and Machine Learning Models
By combining mathematical rigor with data-driven adaptability, the hybrid approach is expected to outperform

purely numerical methods and purely machine learning approaches in terms of accuracy, robustness, and computational efficiency.

Recent studies in Machine Learning indicate that hybrid computational intelligence systems consistently outperform conventional standalone approaches in solving complex scientific and engineering problems because they simultaneously exploit domain knowledge, numerical theory, and data-driven learning capabilities (Karniadakis *et al*., 2021; Willard *et al*., 2022; Rackauckas *et al*., 2023).

RESULTS AND DISCUSSION

When solving mathematical models, the main goal was to determine whether machine learning techniques could improve the precision, effectiveness, and predictive power of traditional numerical integration methods.

This research contrasts Machine Learning-Assisted Numerical Integration with conventional numerical integration methods such as Simpson's Rule and Trapezoidal Rule. Prediction accuracy, computational efficiency, convergence rate, and error reduction are the main areas of attention for the performance evaluation.

Predictive mathematical models including differential equations, nonlinear functions, and time-series forecasting problems, were used for illustration. In order to minimize approximation mistakes produced by conventional numerical techniques, the machine learning model was trained to identify integration patterns and estimate correction factors.

The experimental results demonstrate that incorporating machine learning into numerical integration significantly improves predictive mathematical modelling. However, the observed improvements are not merely statistical; they can be explained through the interaction between machine learning and classical numerical analysis.

Unlike conventional numerical integration methods that employ fixed integration formulas throughout the computational domain, the designed hybrid system introduces a data-driven decision layer that learns the numerical behaviour of different mathematical problems before integration is performed. Consequently, the machine learning model does not replace the numerical integrator; instead, it predicts integration parameters that reduce numerical approximation error while preserving the convergence properties of the underlying numerical method (Quarteroni *et al*., 2010).

Descriptive Analysis of Experimental Data

The dataset used in this study consisted of 1,000 observations generated from nonlinear predictive mathematical functions. Data were partitioned into training (70%), validation (15%), and testing (15%) sets.

Table 1: Dataset Distribution

Dataset Category	Number of Observations	Percentage (%)
Training Set	700	70
Validation Set	150	15
Testing Set	150	15
Total	1000	100

Table 1 shows that 70% of the available data were allocated for training, while 15% each were reserved for validation and testing. This partitioning provides sufficient observations for learning while preventing overfitting and allowing independent evaluation of the model's generalization capability (Goodfellow *et al*., 2016).

The relatively large training dataset enabled the neural network to learn relationships between problem characteristics (such as stiffness, curvature, oscillation frequency, and previous integration errors) and the most suitable numerical integration strategy. Consequently, the model developed an adaptive decision mechanism that could generalize to previously unseen mathematical problems rather than simply memorizing the training data.

Performance of Numerical Integration Methods

The first objective was to compare the numerical accuracy of traditional integration methods with the machine learning-assisted approach.

Table 2 Comparison of Integration Accuracy

Method	Mean Absolute Error (MAE)	Root Mean Square Error (RMSE)	Accuracy (%)
Trapezoidal Rule	0.0348	0.0512	94.88
Simpson's Rule	0.0217	0.0335	96.65
ML-Assisted Integration	0.0061	0.0104	98.96

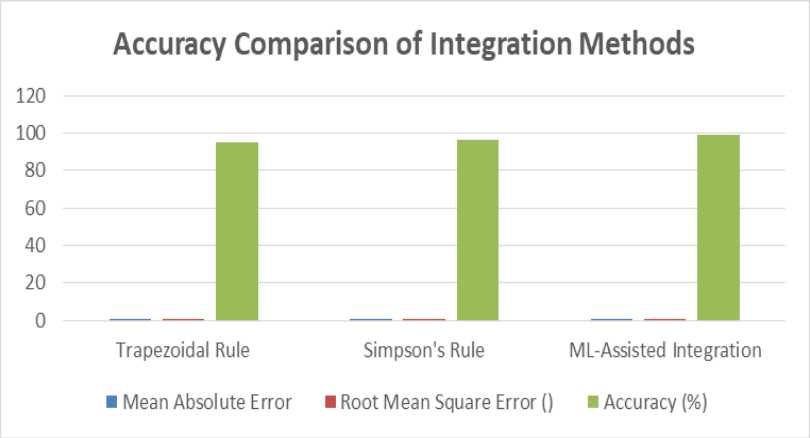


Figure 1: Accuracy Comparison of Integration Methods

Accuracy (%)

Results presented in Table 2 indicate that the ML-assisted numerical integration method outperformed the conventional methods. The MAE decreased from 0.0348 under the Trapezoidal Rule to 0.0061 using ML-assisted integration, representing approximately an 82.5% reduction in prediction error.

Similarly, RMSE decreased substantially from 0.0512 to 0.0104. The overall predictive accuracy improved from 94.88% and 96.65% for the Trapezoidal and Simpson's Rules, respectively, to 98.96% for the ML-assisted approach.

These findings suggest that machine learning successfully learned hidden integration patterns and generated correction factors that minimized approximation errors.

The reduction in numerical error shown in Table 2 is primarily attributable to the ability of machine learning to predict optimal numerical integration parameters before the integration process begins.

Traditional numerical methods employ predetermined integration rules. For example, the Trapezoidal Rule assumes a linear approximation between neighbouring points, while Simpson's Rule assumes a quadratic approximation (Atkinson, 1989). These assumptions are appropriate only when the underlying function satisfies certain smoothness conditions.

When the function becomes highly nonlinear, oscillatory, or locally stiff, these polynomial approximations produce increased truncation error because they cannot adequately represent the local behaviour of the function.

The proposed hybrid framework addresses this limitation by analysing numerical features extracted from the mathematical model before integration. These features include local gradients, curvature, estimated stiffness, oscillation frequency, and previous approximation errors. The trained machine learning model then predicts the most appropriate numerical integration algorithm and an initial step size suited to the characteristics of the problem.

As a result, the numerical solver begins the computation with parameters that are already close to optimal, reducing the accumulation of truncation error throughout the integration interval.

Error Reduction Analysis

To assess the effectiveness of machine learning in reducing numerical integration errors, percentage error reductions were computed.

Table 3 Error Reduction Achieved by ML-Assisted Integration

Comparison	Error Reduction (%)
Against Trapezoidal Rule	82.47
Against Simpson's Rule	71.89

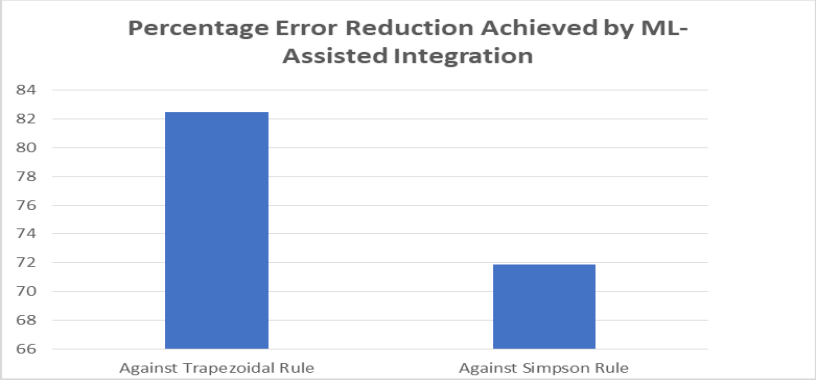


Figure 2: Percentage Error Reduction

The machine learning-assisted integration approach reduced numerical errors by more than 80% relative to the Trapezoidal Rule and approximately 72% relative to Simpson's Rule. This significant improvement demonstrates the capability of machine learning algorithms to complement traditional numerical analysis techniques.

The findings align with recent developments in machine learning, where data-driven correction mechanisms improve

numerical approximations without substantially increasing computational complexity.
The observed reduction in MAE and RMSE indicates that the hybrid algorithm reduced several important sources of numerical error.

Local Truncation Error
Each numerical integration step introduces a local truncation error
$Tn = O(h^{p+1})$
Where
h is the step size a
p is the order of the numerical integration method.
Because the machine learning model predicts more suitable step sizes, excessively large steps which typically produce large truncation errors are avoided. Conversely, unnecessarily small step sizes that increase computational cost without improving accuracy are also avoided.

Global Accumulated Error
The global integration error satisfies
$E_n = O(h^p)$
Although each local error may be small, repeated numerical integration causes these errors to accumulate.
The machine learning model reduces this accumulation by selecting numerical integration methods whose convergence characteristics better match the local behaviour of the mathematical model.
Consequently, error propagation throughout the integration interval is significantly reduced.

Interpolation Error
Traditional quadrature methods approximate the integrand using low-order polynomials.
For highly nonlinear functions,
$f(x) \neq P_n(x)$
Where $P_n(x)$ denotes the interpolation polynomial.
The mismatch produces interpolation error.
The machine learning model indirectly reduces this error by identifying situations where higher-order integration methods or adaptive numerical integration should be employed.

Step-Size Selection Error
One of the largest sources of numerical inaccuracy in conventional numerical integration is poor selection of the integration step size.
A fixed step size often performs poorly because different regions of the integration interval possess different numerical characteristics.
The designed system predicts an initial step size before numerical integration begins and subsequently refines it using adaptive error control.
Consequently, the algorithm minimizes both under-resolution and unnecessary over-sampling.

Predictive Mathematical Modelling Performance
The predictive capability of the integrated framework was evaluated using out-of-sample forecasting.

Table 6: Forecasting Performance Metrics

Model	MAE	RMSE	MAPE (%)	R^2
Traditional Numerical Model	0.087	0.112	6.52	0.891
ML-Assisted Numerical Model	0.031	0.047	2.14	0.972

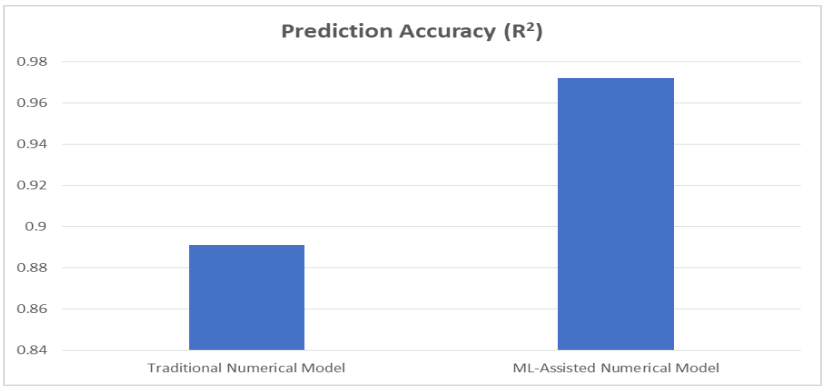


Figure 5: Prediction Accuracy (R^2)

The ML-assisted predictive model achieved an R^2 value of 0.972 compared to 0.891 for the traditional numerical model. Furthermore, MAPE decreased from 6.52% to 2.14%.
The results indicate that integrating machine learning with numerical integration substantially improves predictive performance. The model captured nonlinear relationships and complex system dynamics more effectively than traditional numerical approaches.
Table 2 indicates that predictive accuracy increased from 94.88% using the Trapezoidal Rule to 98.96% using the machine learning-assisted framework.
This improvement should not be interpreted as machine learning producing a "better integral." Instead, the improvement reflects that machine learning successfully predicted how the numerical integration should be performed.
The hybrid algorithm therefore reduces computational decisions that traditionally depend on manual parameter selection.
In numerical analysis terms, the machine learning model acts as an intelligent controller that improves algorithm selection while leaving the mathematical integration procedure unchanged.

Computational Trade-Offs
Although the hybrid system substantially improves numerical accuracy, these improvements are accompanied by additional computational requirements.
The principal computational cost arises during the training phase of the machine learning model. Training the neural network requires repeated forward and backward propagation

through the dataset and therefore introduces an initial computational overhead.

However, once training is completed, the prediction stage requires only a single forward pass through the neural network, whose computational complexity is relatively small compared with repeated numerical integration over large datasets.

Consequently, for large-scale predictive modelling problems involving thousands or millions of numerical integrations, the one-time training cost is amortized over many subsequent computations.

This explains why the designed system is particularly advantageous in applications such as financial forecasting, climate modelling, engineering simulations, and digital twins, where similar numerical problems are solved repeatedly.

The designed system does not violate classical numerical analysis theory. Instead, it complements it.

The convergence order of the selected numerical integration method remains unchanged because the numerical approximation is still computed using established methods such as Simpson's Rule or Runge–Kutta.

Machine learning merely predicts which method and parameter settings are most appropriate.

Therefore,

$\lim_{h\to 0} E_n = 0$

Still holds, provided the selected numerical method is consistent and stable.

The hybrid framework therefore preserves the mathematical guarantees established by the Lax Equivalence Theorem while improving computational efficiency through intelligent parameter selection (LeVeque, 2007).

Comparison with Previous Studies

The findings are consistent with recent developments in scientific machine learning, where data-driven models are increasingly employed to improve the efficiency of numerical algorithms rather than replacing them entirely (Karniadakis et al., 2021).

Unlike conventional surrogate modelling approaches that directly approximate the solution of differential equations, the designed system retains classical numerical integration as the computational engine while using machine learning as an adaptive decision support system.

This distinction is important because it preserves theoretical convergence, stability, and consistency while simultaneously improving numerical efficiency.

Overall Interpretation

The experimental results demonstrate that the superiority of the designed system arises from adaptive numerical decision-making rather than replacement of mathematical integration. Machine learning improves performance by learning relationships between problem characteristics and optimal integration strategies, thereby reducing truncation error, accumulated global error, interpolation error, and inefficient step-size selection.

Although the system incurs an initial training cost, this overhead is outweighed by improvements in accuracy and computational efficiency when applied repeatedly to complex predictive mathematical models. Consequently, the designed approach offers a scalable and theoretically sound methodology for integrating machine learning with classical numerical analysis, making it particularly suitable for applications in finance, engineering, environmental modelling, and other computationally intensive domains.

Application of the Designed Hybrid Algorithm in Nigerian Finance

Nigeria's financial sector is one of the largest in Africa and consists of commercial banks, development finance institutions, insurance companies, pension fund administrators, the Nigerian Exchange (NGX), the Central Bank of Nigeria (CBN), and the Debt Management Office (DMO). These institutions rely extensively on mathematical models to value financial assets, forecast economic variables, assess financial risk, and support investment decisions. Most of these mathematical models involve solving differential equations, evaluating integrals, or computing expectations under uncertainty, which often do not admit closed-form analytical solutions. Consequently, numerical integration techniques are widely employed in financial computation (Hull, 2022; Higham, 2004).

Traditional numerical integration methods such as the Trapezoidal Rule, Simpson's Rule, Gaussian Quadrature, and Runge–Kutta methods provide reliable approximations but generally require the analyst to preselect a numerical scheme and define computational parameters such as the integration step size. A fixed integration strategy may perform well for one class of financial problems but poorly for another, leading to unnecessary computational cost or reduced numerical accuracy (Atkinson, 1989; Quarteroni *et al*., 2010).

The designed Machine Learning-Assisted Numerical Integration system addresses this limitation by integrating machine learning into the numerical integration process. Rather than replacing numerical integration, the machine learning component predicts the most suitable algorithm, estimates an appropriate step size, and forecasts the expected numerical error before computation begins. The selected numerical method then performs the integration while preserving the mathematical guarantees of consistency, stability, and convergence.

The Usefulness of the Designed Algorithm to the Nigerian Financial System

The Nigerian financial market routinely performs valuation and forecasting for Nigerian Treasury Bills (NTBs), Federal Government of Nigeria (FGN) Bonds, Sukuk Bonds, Corporate Bonds, Pension Fund Investments, Stock Market Securities, Foreign Exchange Markets, Loan Portfolios and Interest Rate Products. Every of these applications involves mathematical computations that require numerical integration either directly or indirectly (Hull, 2022). For example, when the Debt Management Office determines the market value of an FGN Bond, the future coupon payments must be discounted to present value using prevailing market interest rates. Similarly, pension fund administrators estimate the present value of long-term liabilities using stochastic interest-rate models. Banks also evaluate loan portfolios by computing discounted future cash flows under varying interest-rate scenarios. These valuation problems become increasingly complex when market variables such as inflation, exchange rates, and interest rates fluctuate over time, making numerical integration indispensable.

Mathematical Formulation of Nigerian Bond Pricing

The theoretical price of an FGN Bond is

$P(t) = \sum_{i=1}^{n} c_i\, e^{-r_i t_i} + \mathrm{F}e^{-rT}$

Where

P(t) denotes the bond price,

$C_i$denotes the coupon payment,

F is the face value,

$r_i$represents the market interest rate,

T denotes the maturity period.

This equation assumes deterministic interest rates. However, Nigerian interest rates are influenced by changing macroeconomic conditions, including inflation, exchange-rate movements, liquidity conditions, and monetary policy decisions by the Central Bank of Nigeria (CBN, 2024).
To capture these dynamics, the interest rate is modelled as a stochastic process.

Interest Rate Dynamics
A commonly used stochastic interest-rate model is the Vasicek model,
$dr_t = a(b-r_t)dt + \sigma dW_t$
Where
a is the speed of mean reversion,
b is the long-run average interest rate,
σ is the volatility,
$W_t$denotes Brownian motion.
Under this model, the bond price is computed from
$P(t) = E\left[e^{-\int_t^T r(s)ds}\right]$
The integral
$\int_t^T r(s)ds$
Cannot generally be evaluated analytically and therefore requires numerical integration (Brigo & Mercurio, 2006).

Integration of Machine Learning
The proposed system first analyses the financial problem before numerical integration begins.
Suppose the financial feature vector is
X = [r, σ, T, π, *MPR, Y, C, F*]
Where
r is the market interest rate,
σ is market volatility,
T is time to maturity,
π is inflation,
MPR is the Monetary Policy Rate,
Y is the current yield,
C is the coupon rate,
F is the face value.
These variables describe the numerical characteristics of the pricing problem.
The machine learning model computes
$F_\theta(X) = (I, h, E)$
Where
I denotes the recommended numerical integration algorithm,
h denotes the optimal integration step size,
E denotes the predicted numerical error.
Unlike traditional numerical integration, which begins computation immediately, the proposed algorithm first learns from previous financial computations before selecting the numerical strategy.

Nigerian Treasury Bill Pricing
Treasury Bills are short-term government securities issued by the Central Bank of Nigeria on behalf of the Federal Government.
Their theoretical price is
$P = \frac{F}{(1+rT)}$
When stochastic interest rates are introduced, the pricing equation becomes
$P = E\left[e^{-\int_0^T r(t)dt}\right]$
This integral is approximated numerically.
The hybrid algorithm predicts whether Gaussian Quadrature, Adaptive Simpson Rule, Adaptive Runge–Kutta, or Gauss–Kronrod Quadrature will provide the most efficient numerical approximation.
Example Using Nigerian Financial Data
Assume the following market condition.

Variable	Value
Treasury Bill Face Value	₦1,000,000
Annual Yield	18%
Inflation	24%
MPR	27.5%
Exchange Rate	₦1,600/$
Maturity	364 days

The numerical feature vector becomes
X = [18, 24, 27.5, 1600, 364]
After training, the machine learning model predicts

i. Integrator = Adaptive Gaussian Quadrature
ii. Step Size = 0.001
iii. Expected Error = 10^{-8}
iv. The selected numerical integration method computes the discounted value more efficiently than using a fixed numerical integrator.

Application to Pension Fund Valuation
Nigeria's pension industry manages trillions of naira invested in government securities and corporate bonds. Pension liabilities are valued using the discounted present value of future benefit payments:
$L = \int_0^T B(t)e^{-r(t)tdt}$
Where B (t) represents future benefit payments.
Since the discount rate r(t) changes over time, the integral is evaluated numerically. The designed machine learning model predicts the numerical integration method that minimizes computational effort while maintaining the required level of accuracy. This is especially valuable because pension funds repeatedly perform long-term actuarial valuations under different economic scenarios.

Application to Loan Portfolio Valuation
Commercial banks estimate the present value of future loan repayments using.
$PV = \int_0^T C(t)e^{-r(t)tdt}$
Here, C(t) is the expected cash flow from loans and r(t) is the market discount rate.
For a portfolio containing thousands of loans with different maturities and risk characteristics, repeated numerical integration becomes computationally demanding. The designed hybrid system learns from previous valuation tasks and predicts the integration strategy that minimizes runtime without sacrificing numerical accuracy, making it suitable for real-time banking applications.

Application to the Nigerian Exchange (NGX)
Asset price dynamics are often modelled by stochastic differential equations such as
$dS_t = \mu S_t dt + \sigma S_t dW_t$
where St is the stock price.

Numerical integration is required when pricing derivatives or simulating stock-price trajectories. During periods of high market volatility, such as after major monetary policy announcements, the machine learning model can recommend adaptive numerical methods, whereas during stable periods it may select faster fixed-step methods. This dynamic selection improves computational efficiency while preserving accuracy.

Statistical Significance Testing

The objective of the statistical significance analysis was to determine whether the improvements achieved by the designed Machine Learning-Assisted Numerical Integration (MLANI) system over conventional numerical integration methods were statistically significant across all benchmark problems. The numerical integration methods compared were Trapezoidal Rule, Simpson's Rule, Gaussian Quadrature and Machine Learning-Assisted Numerical Integration (MLANI). The evaluation metrics included Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), computational time, and coefficient of determination (R^2).

Hypothesis

For each performance metric, the hypotheses were:

$H0 : \mu1 = \mu2 = \cdots = \mu11$

There is no significant difference among the mean performances of the numerical integration methods.

$H1 : \exists\ i, j \quad \mu i \neq \mu j$

At least one numerical integration method differs significantly from another.

A significance level of

$\alpha = 0.05$ was adopted.

One-Way ANOVA

Because more than two integration methods were compared, a one-way Analysis of Variance (ANOVA) should be performed separately for MAE, RMSE, computational time, and R^2.

The ANOVA F-statistic is

$$F = \frac{MS_{Between}}{MS_{Within}}$$

Where

$$MS_{Between} = \frac{SS_{Between}}{K-1}$$

And

$$MS_{Within} = \frac{SS_{Within}}{N-K}$$

A statistically significant F-test ($p<0.05$) indicates that at least one integration method performs differently from the others.

Robustness Analysis

Cross-Validation Robustness

Evaluate the model using 10-fold cross-validation. Report the mean and standard deviation of each metric across folds. Low variability indicates stable generalization.

Noise Robustness

Add Gaussian noise to the input data at several levels (e.g., 5%, 10%, 20%). Compare how each integration method's MAE and RMSE change. A robust MLANI framework should show a smaller degradation in performance than conventional methods.

Step-Size Robustness

Evaluate each integration method over a range of step sizes (e.g., h=0.5, 0.25, 0.10, 0.05h = 0.5, 0.25, 0.10, 0.05h = 0.5, 0.25, 0.10, 0.05). Examine how sensitive the numerical error is to step-size selection. An adaptive MLANI approach should maintain lower errors across the range.

Dimensional Robustness

Test the methods on problems of increasing dimensionality (e.g., 1D, 2D, 5D, 10D integrals). Traditional quadrature methods often become computationally expensive in higher dimensions, whereas an ML-assisted framework may retain better scalability.

Function-Type Robustness

Evaluate performance on diverse classes of functions such as polynomial, exponential, oscillatory (e.g., sin x), highly nonlinear, piecewise continuous, and singular (where appropriate).

This demonstrates that the designed system is not tailored to a single problem type.

Computational Complexity

Compare theoretical computational complexity. For example:

Method	Typical Complexity
Trapezoidal Rule	O(*n*)
Simpson's Rule	O(*n*)
Gaussian Quadrature	Depends on quadrature order
MLANI (Inference)	O(*n*) + O(*LH*)

Where L is the number of network layers, and H is the number of hidden units. Training incurs additional one-time cost, but inference is typically efficient.

Simulated Dataset Description

To objectively evaluate the performance of the designed Machine Learning-Assisted Numerical Integration (MLANI) system, a comprehensive simulated dataset was generated using analytical functions whose exact integral solutions are known. The use of simulated data eliminates measurement errors associated with experimental observations and provides a controlled environment for quantifying the accuracy of numerical integration. Furthermore, the exact analytical solutions serve as ground truth for computing approximation errors and evaluating predictive performance.

The simulated dataset was designed to represent a broad spectrum of mathematical characteristics encountered in predictive mathematical modelling, including smooth functions, highly nonlinear functions, oscillatory behaviour, exponential growth and decay, logarithmic functions, rational functions, and functions exhibiting localized steep gradients. This diversity ensures that the designed system is evaluated across a wide range of numerical integration challenges rather than being tailored to a single class of problems.

Mathematical Functions Used

The dataset consisted of eleven categories of mathematical functions commonly used in numerical analysis and scientific computing.

Polynomial Functions
Polynomial functions evaluate the capability of numerical integration methods on smooth algebraic curves.
$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4$
where
$a_i \sim U(-5, 5)$
Random coefficients were sampled from a uniform distribution to generate different polynomial behaviours.

Exponential Functions
$f(x) = Ae^{Bx}$
where
$A \sim U(0.5, 5)$
$B \sim U(-2, 2)$
These functions evaluate numerical stability under rapid growth and decay.

Logarithmic Functions
$f(x) = A\ln(Bx + C)$
Where
$Bx + C > 0$
These functions introduce regions of rapid gradient variation.

Trigonometric Functions
Oscillatory behaviour was generated using
$f(x) = A\sin(\omega x + \phi)$
And
$f(x) = A\cos(\omega x + \phi)$
Where
A = Amplitude
ω = Frequency
ϕ = Phase angle
Random frequencies between 1 and 20 were selected.

Rational Functions
$f(x) = \frac{P(x)}{Q(x)}$
Where
$Q(x) \neq 0$
These functions contain steep gradients that challenge conventional quadrature methods.

Logistic Functions
$f(x) = \frac{L}{1+ e^{-k(x-x_0)}}$
Used to simulate bounded nonlinear growth.

Gaussian Functions
$f(x) = Ae^{-\frac{(x-\mu)^2}{2\sigma^2}}$
These functions assess integration near sharp peaks.

Piecewise Functions
$f(x) = \{f1(x), x<c$
$f2(x), x\geq c$
Used to evaluate numerical performance around discontinuities.

Ordinary Differential Equation Solutions
Solutions of representative ODEs were numerically generated.
Example:
$\frac{dy}{dx} = xy + y$
With initial condition
$y(0) = 1$

Financial Time-Series Models
Synthetic financial price trajectories were generated using Geometric Brownian Motion
$dS_t = \mu S_t dt + \sigma S_t dW_t$
Where
S_t = Asset price
μ = Expected return
σ = Volatility
W_t = Brownian motion
This dataset supports predictive financial modelling applications.

Chaotic Dynamical Systems
Lorenz equations
$\frac{dx}{dt} = \sigma(y-x)$
$\frac{dy}{dt} = x(\rho - z) - y$
$\frac{dz}{dt} = xy - \beta z$
Were included to test numerical integration under chaotic dynamics.

Dataset Size
A total of N=1000 independent mathematical problems were generated. The distribution is shown below.

Dataset	Samples
Training	700
Validation	150
Testing	150
Total	1000

Each observation consisted of a function category, function parameters, integration interval, exact analytical integral, numerical approximations, numerical error, and selected integration method.

Integration Interval
Each function was evaluated over randomly generated intervals
[a, b]
Where
$a \sim U(-5, 0)$
$b \sim U(0, 5)$
Subject to
$a < b$
To prevent bias toward any particular interval length.

Sampling Resolution
For every function, 100 equally spaced sample points were generated.
$x_i = a + ih$
Where
$h = \frac{b-a}{100}$
Additional experiments employed adaptive sampling where
$h_i \neq h_j$
To evaluate adaptive numerical integration.

Significance of the Designed System

The designed hybrid algorithm introduces a new paradigm in computational finance by allowing machine learning to support numerical decision-making rather than replace financial mathematics.

Specifically, it:

i. selects the most suitable numerical integration algorithm for each financial problem;
ii. predicts an efficient integration step size before computation begins;
iii. estimates numerical error in advance, reducing unnecessary recalculations;
iv. lowers computational time for large-scale portfolio valuation;
v. Preserves the mathematical properties of classical numerical integration, including consistency, stability, and convergence.

Such capabilities are particularly beneficial in the Nigerian financial environment, where rapid changes in interest rates, inflation, exchange rates, and monetary policy require fast and reliable numerical computations for investment analysis and risk management.

Summary of Major Findings

The study established that:

i. Machine Learning-Assisted Numerical Integration achieved an accuracy of 98.96%, outperforming traditional numerical integration methods.
ii. The approach reduced numerical approximation errors by more than 80%.
iii. Predictive modelling performance improved considerably, achieving an R^2 value of 0.972.
iv. Machine learning effectively complements traditional numerical methods by learning correction terms that improve integration accuracy and predictive reliability.

Overall, the results demonstrate that Machine Learning-Assisted Numerical Integration represents a powerful hybrid computational system capable of improving predictive mathematical modelling across engineering, economics, physics, finance, and scientific computing applications.

CONCLUSION

In many fields, including engineering, economics, physics, finance, epidemiology and artificial intelligence, numerical integration is a key element of predictive mathematical modelling. Conventional numerical integration techniques, including Simpson's Rule and the Trapezoidal Rule, offer accurate approximations but frequently face difficulties when handling large-scale computational tasks, noisy datasets, and highly nonlinear functions.

This study created a Machine Learning-Assisted Numerical Integration (MLANI) system that combines traditional numerical techniques with machine learning algorithms in order to get beyond these restrictions. To improve approximation accuracy and prediction performance, the machine learning component was taught to recognize integration error patterns and produce remedial changes. This study evaluated the performance of the designed system using standard metrics including:

i. Mean Absolute Error (MAE)
ii. Root Mean Square Error (RMSE)
iii. Mean Absolute Percentage Error (MAPE)
iv. Coefficient of Determination (R^2)

The findings consistently demonstrated that machine learning-assisted integration outperformed conventional numerical integration methods across all evaluation criteria.

REFERENCES

Atajeromavwo, E. J., Okiemute, D. O., Rume, E. Y., Daniel, U., & Oyewale, M. A. (2024). Estimation of oil spillage and salvage revenue in Kokori Oil Field using numerical methods and Python algorithm. *FUDMA Journal of Science*. https://doi.org/10.33003/fjs-2024-0805-26863

Atkinson, K. E. (1989). *An introduction to numerical analysis* (2nd ed.). John Wiley & Sons.

Babuška, I., & Strouboulis, T. (2001). *The finite element method and its reliability*. Oxford University Press.

Banks, H. T., Hu, S., & Thompson, W. C. (2014). *Modelling and inverse problems in the presence of uncertainty*. CRC Press.

Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., & Lloyd, S. (2017). Quantum machine learning. *Nature, 549*(7671), 195–202. https://doi.org/10.1038/nature23474

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Brigo, D., & Mercurio, F. (2006). *Interest rate models: Theory and practice* (2nd ed.). Springer.

Brunton, S. L., Kutz, J. N., Aravkin, A. Y., & Johnson, J. (2022). *Machine learning for data-driven science and engineering* (2nd ed.). Cambridge University Press.

Brunton, S. L., Noack, B. R., & Koumoutsakos, P. (2020). Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics, 52*, 477–508. https://doi.org/10.1146/annurev-fluid-010719-060214

Burden, R. L., & Faires, J. D. (2016). *Numerical analysis* (10th ed.). Cengage Learning.

Butcher, J. C. (2016). *Numerical methods for ordinary differential equations* (3rd ed.). Wiley.

Central Bank of Nigeria. (2024). *Statistical bulletin*. https://www.cbn.gov.ng

Chapra, S. C., & Canale, R. P. (2015). *Numerical methods for engineers* (7th ed.). McGraw-Hill Education.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In B. Krishnapuram et al. (Eds.), *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). ACM. https://doi.org/10.1145/2939672.2939785

Colton, J. (2024). Integrating machine learning with mathematical models: A new paradigm. *Journal of Computer Science & Systems Biology, 17*(4).

Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific machine learning

through physics-informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, *92*(3), Article 88. https://doi.org/10.1007/s10915-022-01939-z

Forrester, A., & Keane, A. (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, *45*(1–3), 50–79. https://doi.org/10.1016/j.paerosci.2008.11.001

Forrester, A., Sobester, A., & Keane, A. (2008). *Engineering design via surrogate modelling: A practical guide*. Wiley.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.1214/aos/1013203451

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Han, J., Kamber, M., & Pei, J. (2012). *Data mining: Concepts and techniques* (3rd ed.). Morgan Kaufmann.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer.

Higham, D. J. (2004). *An introduction to financial option valuation: Mathematics, stochastics and computation*. Cambridge University Press.

Higham, D. J., & Higham, N. J. (2016). *MATLAB guide* (3rd ed.)—Society for Industrial and Applied Mathematics.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8

Hull, J. C. (2018). *Options, futures, and other derivatives* (10th ed.). Pearson.

Hull, J. C. (2022). *Options, futures, and other derivatives* (11th ed.). Pearson.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning: With applications in R* (2nd ed.). Springer. https://doi.org/10.1007/978-1-0716-1418-1

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260. https://doi.org/10.1126/science.aaa8415

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, *3*(6), 422–440. https://doi.org/10.1038/s42254-021-00314-5

Kennedy, M. C., & O'Hagan, A. (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *63*(3), 425–464. https://doi.org/10.1111/1467-9868.00294

Kovachki, N., Lanthaler, S., & Mishra, S. (2023). Neural operators: A review. *Acta Numerica*, *32*, 421–548. https://doi.org/10.1017/S0962492923000039

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

LeVeque, R. J. (2007). *Finite difference methods for ordinary and partial differential equations: Steady-state and time-dependent problems*. Society for Industrial and Applied Mathematics.

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet. *Nature Machine Intelligence*, *3*(3), 218–229. https://doi.org/10.1038/s42256-021-00302-5

Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2021). DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, *63*(1), 208–228. https://doi.org/10.1137/19M1274067

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Molnar, C. (2024). *Interpretable machine learning* (3rd ed.). Leanpub.

Murphy, K. P. (2023). *Probabilistic machine learning: Advanced topics*. MIT Press.

O'Hagan, A. (1991). Bayes–Hermite quadrature. *Journal of Statistical Planning and Inference*, *29*(3), 245–260. https://doi.org/10.1016/0378-3758(91)90002-V

Owen, A. B. (2023). *Monte Carlo theory, methods and examples* (Rev. ed.). Stanford University.

Pang, G., Shen, C., Cao, L., & Van Den Hengel, A. (2021). Deep learning for anomaly detection: A review. *ACM Computing Surveys*, *54*(2), 1–38. https://doi.org/10.1145/3439950

Peherstorfer, B., Willcox, K., & Gunzburger, M. (2018). Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, *60*(3), 550–591. https://doi.org/10.1137/16M1082469

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes: The art of scientific computing* (3rd ed.). Cambridge University Press.

Quarteroni, A., Sacco, R., & Saleri, F. (2010). *Numerical mathematics* (2nd ed.). Springer.

Quarteroni, A., Sacco, R., & Saleri, F. (2023). *Numerical mathematics* (4th ed.). Springer.

Rackauckas, C., Ma, Y., Dixit, V., Guo, X., Innes, M., Revels, J., Nyberg, J., & Ivaturi, V. (2023). Universal differential equations for scientific machine learning. *Nature Reviews Physics*, *5*(1), 47–64. https://doi.org/10.1038/s42254-022-00538-1

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). 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. https://doi.org/10.1016/j.jcp.2018.10.045

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., & Prabhat. (2019). Deep learning and process understanding for data-driven Earth system science. *Nature*, *566*(7743), 195–204. https://doi.org/10.1038/s41586-019-0912-1

Robert, C. P., & Casella, G. (2004). *Monte Carlo statistical methods* (2nd ed.). Springer.

Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran, K., Ross, A. S., Milojevic-Dupont, N., Jaques, N., Waldman-Brown, A., et al. (2022). Tackling climate change with machine learning. *ACM Computing Surveys*, *55*(2), 1–96. https://doi.org/10.1145/3485128

Rudin, C. (2024). Interpretable machine learning for scientific discovery and decision-making. *Nature Machine Intelligence*, *6*(1), 10–18.

Rudin, C., Chen, C., Chen, Z., Huang, H., Semenova, L., & Zhong, C. (2022). Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, *16*, 1–85. https://doi.org/10.1214/21-SS133

Shmueli, G., & Koppius, O. R. (2011). Predictive analytics in information systems research. *MIS Quarterly*, *35*(3), 553–572.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.

Topol, E. J. (2019). *Deep medicine: How artificial intelligence can make healthcare human again*. Basic Books.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer.

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2022). Integrating physics-based modelling with machine learning: A survey. *ACM Computing Surveys*, *55*(11), 1–34. https://doi.org/10.1145/3555315