AttNS: Attention-Inspired Numerical Solving For Limited Data Scenarios

Zhongzhan Huang [1 2] **Mingfu Liang** [3] **Shanshan Zhong** [1] **Liang Lin** [1 2]

Abstract

We propose the attention-inspired numerical solver (AttNS), a concise method that helps the generalization and robustness issues faced by the AI-Hybrid numerical solver in solving differential equations due to limited data. AttNS is inspired by the effectiveness of attention modules in Residual Neural Networks (ResNet) in enhancing model generalization and robustness for conventional deep learning tasks. Drawing from the dynamical system perspective of ResNet, We seamlessly incorporate attention mechanisms into the design of numerical methods tailored for the characteristics of solving differential equations. Our results on benchmarks, ranging from high-dimensional problems to chaotic systems, showcase AttNS consistently enhancing various numerical solvers without any intricate model crafting. Finally, we analyze AttNS experimentally and theoretically, demonstrating its ability to achieve strong generalization and robustness while ensuring the convergence of the solver. This includes requiring less data compared to other advanced methods to achieve comparable generalization errors and better prevention of numerical explosion issues when solving differential equations. https://github.com/dedekinds/NeurVec.

1. Introduction

The AI-Hybrid numerical solvers (AHS) are emerging methods (Mishra, 2018; San & Maulik, 2018; Bar-Sinai et al., 2019; Kochkov et al., 2021; Subel et al., 2021; Bruno et al., 2022; List et al., 2022; Dresdner et al., 2022; Frezat et al., 2022; Huang et al., 2023b; Xu et al., 2023) for solving differential equations, designed to integrate classical numerical techniques with deep learning technology. These approaches leverage high-resolution data to learn corrections (Um et al., 2020) for low-resolution numerical solutions. They aim to combine the strengths of both techniques – the reliability of classical numerical methods and the expressive power of neural networks – to mitigate the inherent speed-accuracy trade-off in solving differential equations. For example, for an ordinary differential equation (ODE)

$$\mathrm{d}\mathbf{u}(\mathbf{t})/\mathrm{d}t = \mathbf{f}[\mathbf{u}(t)], \mathbf{u}(0) = \mathbf{c}_0, \tag{1}$$

where $\mathbf{u}(\mathbf{t})$ is a time-dependent d-dimensional state and the $\mathbf{c}_0 \in \mathbb{R}^d$ is initial condition. The forward AHS of Eq. (1) can usually (Dresdner et al., 2022) be written as

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \underbrace{S(\mathbf{f}, \mathbf{u}_n, \Delta t_c)\Delta t_c}_{\text{Low-resolution term}} + \underbrace{\text{Net}(\mathbf{u}_n|\phi, \mathbb{D}_f)}_{\text{Correction term}}, \tag{2}$$

where $\text{Net}(\cdot)$ is a neural network with learnable parameters ϕ, $S(\cdot)$ is a numerical integration scheme $\mathbb{D}_f$ denotes the high-resolution data, i.e., the data with high-precision generated by fine step size Δt_f. A coarse step size, Δt_c, accelerates computation, and the associated decrease in accuracy is compensated by the correction term $\text{Net}(\mathbf{u}_n|\phi, \mathbb{D}_f)$ trained by $\mathbb{D}_f$, achieving a good balance between speed and accuracy in solving differential equations.

However, the performance of current AHS is heavily reliant on the quantity of high-quality training data due to the data-driven deep learning approach adopted by the correction term, and **the obtaining of sufficient training data is very expensive**. For instance, if we use data with $\Delta t_f = 0.001$ to train the correction term for $\Delta t_c = 0.1$, generating a single trajectory of data would require over $100\times$ additional computations, which are hard to be accelerated by parallelization due to the iterative nature of Eq.(2). Besides, the high demand for data imposed by complex equations (Huang et al., 2023b) further exacerbates the acquisition cost. Given these challenges, we ask a critical question:

Can we improve the AHS for effective computations even in ***limited data scenarios****?*

To answer this question, in this paper, we conduct a comprehensive analysis using the ODE shown in Eq. (1) as examples. First, we present the specific challenges for AHS in limited data scenarios in Section 2.3. In addition to the inherent generalization issues of neural networks due to limited data, robustness issues (Liang et al., 2022) related to chaos in solving differential equations need to be carefully

[1]Sun Yat-sen University, China [2]Peng Cheng Laboratory, China [3]Northwestern University, USA. Correspondence to: Liang Lin <linliang@ieee.org>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

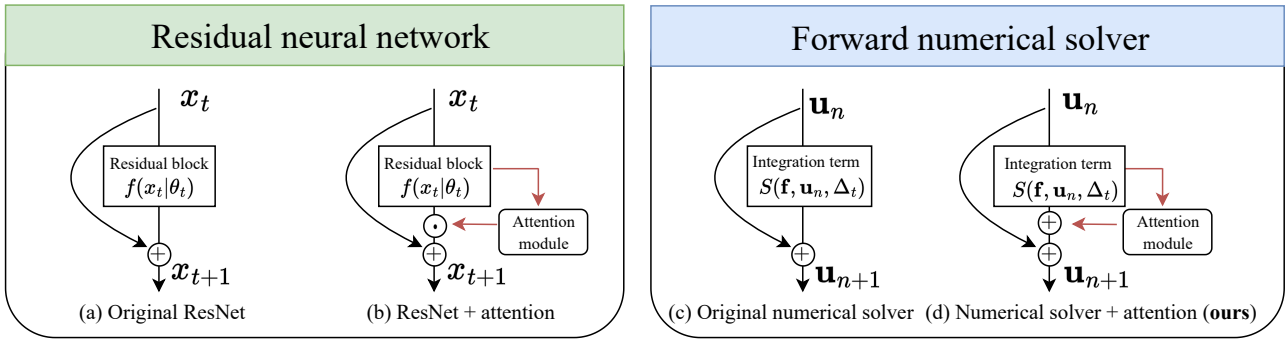

Figure 1: The correspondence of (a) ResNet and (c) forward numerical solver. (d) is a numerical solver with the attention mechanism (ours) inspired by the structure of (b) ResNet with attention. $\odot$ is element-wise multiplication and $\oplus$ is the addition operator. The use of $\oplus$ is **tailored for** solving differential equations and see Section 3 for details.

considered. In response to these challenges, we note that the attention mechanism module in Residual Neural Network (ResNet) has been widely validated to effectively enhance the model's generalization and robustness (Hu et al., 2018; Huang et al., 2020; Woo et al., 2018; Wang et al., 2020; Liang et al., 2020; Zhong et al., 2023) in conventional deep learning tasks (see Section 2.4). Also, considering the correspondence between ResNet and numerical solvers established in the dynamical system perspective of ResNet (see Section 2.2 and Fig.1), we follow the characteristics of differential equation solving and seamlessly integrate the attention mechanism module into the numerical solver, proposing a simple yet effective Attention-Inspired Numerical Solver (AttNS) in Section 3.

Through experiments with several standard benchmarks of high-dimensional or chaotic systems, including the spring-mass system, the elastic pendulum, and the K-link pendulum, the results in Section 4 show that AttNS consistently outperforms existing state-of-the-art AHS in limited data scenarios, without the need for intricate model crafting. Next, in Section 5, we conduct experimental and theoretical analyses of AttNS, proving its strong generalization and robustness abilities akin to the attention mechanism in conventional deep learning tasks, while ensuring solver convergence. This includes achieving equivalent generalization errors with less data compared to other advanced methods and better preventing numerical explosion issues in solving differential equations. Finally, we discuss the limitations of AttNS. We summarize our contributions as follows:

- For effective AHS in limited data scenarios, we incorporate the attention mechanism into AHS and proposed a simple-yet-effective method AttNS. The standard benchmarks show that AttNS consistently outperforms other advanced AHS methods.
- We conducted experimental and theoretical analyses for AttNS, demonstrating its strong generalization and robustness capabilities while ensuring solver convergence. Finally, we discuss the limitations of AttNS.

2. Preliminaries and Related Works

2.1. The Solving of Differential Equation

For a given equation as shown in Eq. (1), classical forward numerical methods, such as the Euler method (Shampine, 2018), Runge-Kutta method (Butcher, 2016), etc., solve it using the iterative formula

$$\mathbf{u}_{n+1} = \mathbf{u}_n + S(\mathbf{f}, \mathbf{u}_n, \Delta t)\Delta t, \quad \mathbf{u}_0 = \mathbf{c}_0. \tag{3}$$

Different methods have different S. For example, for the Euler method (Shampine, 2018), we have $S(\mathbf{f}, \mathbf{u}_n, \Delta t)\Delta t = \mathbf{f}(\mathbf{u}_n)\Delta t$, where Δt is a given step size, and $\mathbf{u}_n \in \mathbb{R}^d$ is an approximated solution at time $\sum_{i=0}^{n} \Delta t$. The AI-Hybrid numerical solver (AHS) considered in this paper, as shown in Eq. (2), introduces a correction term to help high-precision solving of the differential equation even with larger step sizes which makes solving faster. For other methods utilizing deep learning techniques for solving, such as direct fitting using neural networks (Geneva & Zabaras, 2022; Liang et al., 2021), operator-based methods focusing on replacing differential operators (Li et al., 2021b; Lu et al., 2021), neuralODE-based methods (Chen et al., 2018; Kidger et al., 2021; Dupont et al., 2019), Hamiltonian-based methods (Greydanus et al., 2019; Chen et al., 2019; Liang et al., 2022), and physics-informed neural networks (Choudhary et al., 2020; Raissi et al., 2019; Ji et al., 2021; Cai et al., 2021), they're **hard to conduct a fair comparison with AHS due to their fundamentally different paradigms**. Therefore, this paper primarily focuses on AHS instead of other deep learning-based solving methods.

2.2. Dynamical System Perspective of ResNet

As shown in Fig.1(a), the simplified residual block in ResNet (He et al., 2016) can be written as

$$x_{t+1} = x_t + f(x_t|\theta_t), \tag{4}$$

where $x_t \in \mathbb{R}^d$ is the input of neural network $f(\cdot|\theta_t)$ with the learnable parameters θ_t in t-th block . Several recent studies (Weinan, 2017; Queiruga et al., 2020; Zhu et al., 2022; Meunier et al., 2022) have uncovered valuable connections between residual blocks and dynamic systems. i.e., the residual blocks can be interpreted as one step of forward numerical methods in Eq. (3) and Fig.1(c). The initial condition $\mathbf{u}_0 = \mathbf{c}_0$ in Eq. (3) corresponds to the initial input x_0 of the network, and $\mathbf{u}_t$ corresponds to the input feature x_t in t-th block. The output of neural network $f(\cdot|\theta_t)$ in t-th block can be regarded as an integration $S(\mathbf{u}_t, \mathbf{f}, \Delta t)$ with step size Δt and numerical integration scheme S. Given the above connection, some dynamical system theories (Chang et al., 2017; Chen et al., 2018; Huang et al., 2022b; Lu et al., 2018) can be transferred to the analysis of ResNet.

2.3. The Challenges for AHS in Limited Data Scenarios

We first briefly define limited data scenarios (LDS). Generally, the performance of AHS is positively correlated with the training data size (Huang et al., 2023b), and reaches saturation at data quantity N_b, i.e., further increasing data size hardly boosts performance. In this paper, LDS are situation that the data size is $\lfloor pN_b \rfloor$, where $1 \leq p \leq 0.5$. In fact, the LDS easily occurs as N_b required for a given equation is not known before solving and generating data. A sound way is to make sure, as much as possible, that the solver maintains a good enough solving performance even in LDS, i.e., the goal of this paper. The main challenges of this goal are:

(1) **Generalization issue**. When the training data size is insufficient, the deep learning module in AHS suffers from the inherent generalization problem (Zhang et al., 2021; Neyshabur et al., 2017), i.e., poor prediction of the inputs outside the training data distribution. This results in a trained correlation term that cannot accurately compensate for the accuracy loss caused by the low-resolution term.

(2) **Robustness issue**. Many differential equations are chaotic (Greydanus et al., 2019), i.e. sensitive to the small perturbations in inputs, which makes it hard for AHS to fundamentally model the ubiquitous and elusive randomness of chaos with only limited data, leading to poor generalization (Abu-Mostafa et al., 2012). It is easy to accumulate and propagate error for a non-robust AHS during the iterative solution process in Eq. (2), leading to a numerical explosion.

2.4. Attention Mechanism for ResNet

For the residual block mentioned in Eq. (4), the attention mechanism[1] can be formulated as

$$x_{t+1} = x_t + f(x_t|\theta_t) \odot \alpha_t, \tag{5}$$

where $\alpha_t = Q[f(x_t|\theta_t)]$ is t-th attention module (Hu et al., 2018; Woo et al., 2018) and $\odot$ is element-wise multiplication, which also shown in Fig.1(b). many recent works (Hu et al., 2018; Huang et al., 2020; Woo et al., 2018; Wang et al., 2020; Liang et al., 2020; Zhong et al., 2023) validate the effectiveness of these attention modules to improve model's generalization and robustness in **conventional deep learning tasks**, such as image classification, detection, segmentation, style migration, etc. Specifically,

(1) **Improving generalization**. The attention mechanism has the capability to adaptively learn the weights of features across various tasks and input data (Qin et al., 2021; Huang et al., 2020; Wang et al., 2020). This flexibility enables the model to focus on task-relevant information, adapt to new data, and effectively capture intricate relationships, ultimately enhancing the generalization performance of various conventional deep learning tasks.

(2) **Enhancing robustness**. The attention mechanism has been theoretically proven to mitigate small perturbations in the input, thereby improving robustness (Zhong et al., 2023). Specifically, let ϵ represent the perturbation from noise, satisfying $\|x_0^\epsilon - x_0\| = \epsilon$. The upper bound of the error $\|x_t^\epsilon - x_t\|$ at the L-th layer, caused by x_0^ϵ as input, is given by $\epsilon \prod_{t=1}^{L-1} (1 + \alpha_t \|\theta_t\|_2)$ according to Eq (5), where the adaptive control is exerted by the attention weights α_t. This perspective is supported by experimental evidence in tasks such as style transfer and noise attacks (Liang et al., 2020; Zhong et al., 2023).

3. Method

As mentioned in Section 2.4, attention mechanisms can help model generalization and robustness issues in conventional deep learning tasks. These issues, as revealed in Section 2.3, happen to be the primary challenges for AHS in limited data scenarios. Therefore, it is natural for us to consider seamlessly transferring the attention mechanism to the numerical solver by exploiting the correspondence between the forward numerical solver and ResNet mentioned in Section 2.2. Some previous works also considered using attention to help solution (Geneva & Zabaras, 2022; Takamoto et al., 2023; Rodriguez-Torrado et al., 2022; Hemmasian & Barati Farimani, 2023), **but they are not for AHS and not the attention mechanisms in Eq. (5)**.

(1) A straightforward baseline (AttNS-m). According to Eq. (5), the straightforward numerical solving with attention can be set as Eq. (6) with multiplication operator.

$$\text{AttNS-m:} \quad \hat{\mathbf{u}}_{n+1} = \hat{\mathbf{u}}_n + \hat{S}\Delta t_c \odot Q[\hat{S}|\phi], \tag{6}$$

where $\hat{S} = S(\mathbf{f}, \hat{\mathbf{u}}_n, \Delta t_c)$ and ϕ is the learnable parameters of attention module $Q[\cdot|\phi]$. The training loss is defined as the ℓ_2-Squared distance between the estimated trajectory $\hat{\mathbf{u}} = [\hat{\mathbf{u}}_1, ..., \hat{\mathbf{u}}_N]$ and the ground truth trajectory from the dataset $\mathbb{D}(\text{traj}(\mathbf{u}))$ following previous works (Dresdner et al., 2022):

$$R_e = \lambda \cdot \|\hat{\mathbf{u}} - \mathbf{u}\|_2^2 / N, \tag{7}$$

[1] Unlike Transformer based attention (Vaswani et al., 2017), the attention mechanisms considered in this paper are for ResNet, such as SENet (Hu et al., 2018).

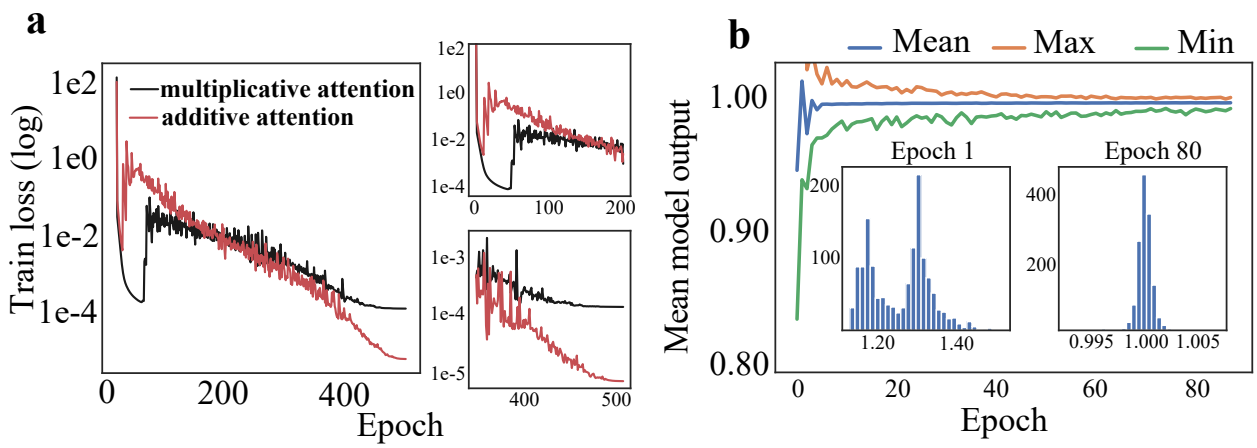

Figure 2: **a**. The loss curves for two kinds of AttNS-m and AttNS. The loss minimization of multiplicative attention is fast at first and then slow during the last epochs, which have a local minimum with a large loss; **b**. The mean of attention value $Q[\hat{S}|\phi]$ (blue) while using Eq.(6), which quickly converges to the vicinity of constant 1 during training.

where λ is a penalty coefficient. λ can alleviate the problem that the value of R_e is too small due to the magnitude of $\mathbf{u}$ in some specific differential equations being too small. This is because, in such a scenario, the gradient will be small enough and affect the optimization of the learnable parameter ϕ. For $\hat{S} \in \mathbb{R}^d$, the architecture of the attention module (Xu et al., 2023) in Eq. (6) is

$$Q[\hat{S}|\phi] = \mathbf{W_h} \circ \mathbf{a} \circ \cdots \circ \mathbf{W_2} \circ \mathbf{a} \circ \mathbf{W_1}[\hat{S}], \qquad (8)$$

where $\mathbf{a}$ is rational activation function (Boullé et al., 2020); $\mathbf{W_i}, i = 2, \cdots, h-1$ are $d_1 \times d_1$ matrices, $\mathbf{W_h} \in \mathbb{R}^{d \times d_1}$ and $\mathbf{W_1} \in \mathbb{R}^{d_1 \times d}$. We set $d_1 = 1024$ and $h = 2$ by default. In Section 5, we will further discuss the concise design in Eq. (8) is tailored for efficient solving of AHS, on the one hand, the small number of parameters enables faster inference speed, and on the other hand, it can be proven that this architecture has the desirable Lipschitz continuity, which provides a strong guarantee for the convergence analysis of our proposed solver. Furthermore, we compare this attention module architecture with other variants.

(2) A baseline tailored for AHS (AttNS). Actually, in mathematics, the attention of AttNS-m in Eq. (6) with multiplicative operator to calibrate the error from low-resolution term may impact the accuracy. The reasons are that (i) the step size Δt_c in $\hat{S}$ used for discretization is usually not equal to the new step size $\Delta t_c' := \Delta t_c \odot Q[\hat{S}|\phi]$ in AttNS-m, which isn't satisfied the mathematical correctness in Eq. (3); (ii) During the learning process of $Q[\hat{S}|\phi]$, this mathematical correctness further forces it to expend some optimization efforts primarily fitting a constant vector $\mathbf{I}$ with all elements being 1, which will bring negative impacts on its accuracy (Wang et al., 2022). Specifically, in Eq. (2), the correction term aims to compensate for the errors of the low-resolution term, and hence the magnitude of the compensation term ϵ_c will be small. Then if multiplicative attention is used, we have $\hat{S}\Delta t_c + \epsilon_c = \hat{S}\Delta t_c \odot Q[\hat{S}|\phi]$, i.e., $(Q[\hat{S}|\phi] - \mathbf{I})\hat{S}\Delta t_c = \epsilon_c$. Since $\|\hat{S}\Delta t_c\| \gg \|\epsilon_c\|$, we have $Q[\hat{S}|\phi] \approx \mathbf{I}$. This analysis is also supported by Fig. 2b.

Algorithm 1 The processing of AttNS and AttNS-m.

Input: A coarse step size Δt_c; The number of step N which satisfied the evaluation time $T = N\Delta t_c$; A given equation $\mathrm{d}\mathbf{u}/\mathrm{d}t = \mathbf{f}(\mathbf{u}), \mathbf{u}(0) = \mathbf{c}_0$; A given numerical integration scheme S; The high-quality dataset $\mathbb{D}(\mathrm{traj}(\mathbf{u}))$. The attention module $Q[\cdot|\phi]$; The learning rate η.
Output: The learned attention module $Q[\cdot|\phi]$.

```
for epoch from 0 to Total epoch do
                                        ▷ Estimate the trajectories
    Sample u = [u_0, u_1, ..., u_N] ~ D(traj(u));
    û_0 ← u_0;
    for t from 0 to N − 1 do
        Calculate integration term Ŝ ← S(f, u_t, Δt_c);
        Calculate attention term Q[Ŝ|φ];
        Get û_{t+1} by Eq.(6) for AttNS-m or Eq.(10) for AttNS;
    end for
                                        ▷ Update the attention module
    û ← [û_1, ..., û_N];
    Calculate the loss R_e by Eq.(7);
    Update the parameters φ by φ ← φ − η∇_φ R_e;
end for
return the attention module Q[·|φ]
```

To mitigate this issue, a simple-yet-effective strategy is to normalize the attention as $\mathbf{I} + \widetilde{Q}[\hat{S}|\phi]$ and just learn the residual part $\widetilde{Q}[\hat{S}|\phi]$, and we can rewrite Eq. (6) as

$$\begin{aligned} \hat{\mathbf{u}}_{n+1} &= \hat{\mathbf{u}}_n + \hat{S}\Delta t_c \odot \left(\mathbf{I} + \widetilde{Q}[\hat{S}|\phi]\right) \\ &= \hat{\mathbf{u}}_n + \hat{S}\Delta t_c \underbrace{+\hat{S}\Delta t_c \odot \widetilde{Q}[\hat{S}|\phi]}_{\text{Additive attention}}, \end{aligned} \qquad (9)$$

where the term $\hat{S}\Delta t_c \odot \widetilde{Q}[\hat{S}|\phi]$ can be regarded as the addi-

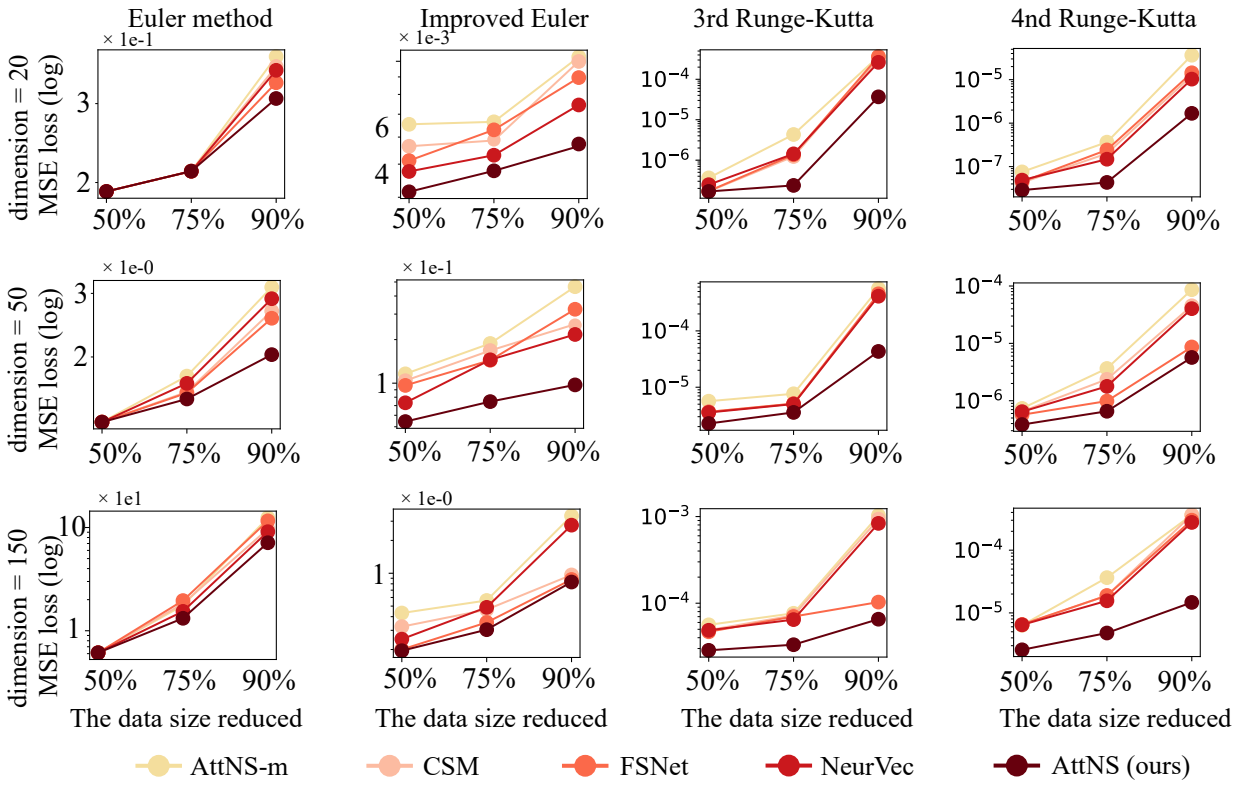

Figure 3: The results of various AHS for four forward numerical solvers on the spring-mass system with different dimensions. "50%" denotes that we reduce the amount of training data by 50%. **The smaller the loss, the better the performance**.

tive attention (Wu et al., 2021; Cheuk et al., 2021; Huang et al., 2022a; Li et al., 2021a; Gao et al., 2021) correction term. Briefly, we set the iterative formula of AttNS to be

$$\text{AttNS:} \quad \hat{\mathbf{u}}_{n+1} = \hat{\mathbf{u}}_n + \hat{S}\Delta t_c + Q[\hat{S}|\phi]. \tag{10}$$

From Eq. (9), AttNS ensures the benefits of attention while satisfying mathematical correctness over AttNS-m, as illustrated in Fig.2a. In terms of the formula, the primary distinction between AttNS and general AHS lies in their input, where the former is $S(\mathbf{f}, \hat{\mathbf{u}}_n, \Delta t_c)$ and another is $\hat{\mathbf{u}}_n$. In fact, this difference is sufficient for AttNS to achieve better generalization error and ensure robustness, which will be discussed in detail in Section 5. Furthermore, the performance gap introduced by the input of attention has also been observed in conventional deep learning tasks (Guo et al., 2020; Hu et al., 2018; Huang et al., 2023a), further validating the rationale behind AttNS. We summarise AttNS and AttNS-m in Alg. 1.

4. Experiments

In this section, we consider two perspectives to verify the effectiveness of the proposed AttNS: (1) on different numerical solvers and (2) different differential equation benchmarks. Specifically, since AttNS is one of AHS, we use many commonly used forward numerical solvers as backbones to evaluate the enhancement achieved by AttNS, including the Euler method, Improved Euler method, 3rd and 4th order Runge-Kutta methods (see Appendix for details). On the other hand, to demonstrate that AttNS is competent for complex differential equations, we further experiment on two chaotic dynamical systems on the 4th order Runge-Kutta method, i.e., k-link pendulum and elastic pendulum. For all experiments in this section, for fair comparisons, we follow the settings in (Huang et al., 2022c; Chen et al., 2020) for all generations of initial conditions and metrics. In this section, we consider AttNS, AttNS-m, NeurVec (Huang et al., 2023b), CSM (Dresdner et al., 2022) and FSNet (Xu et al., 2023) as the baselines of AHS.

(1) **On different numerical solvers**. In this section, we consider a high-dimensional linear system, namely the spring-mass system, and four forward numerical solvers (see Appendix for details). In a spring-mass system, there are d masses and $d+1$ springs connecting in sequence, and they are placed horizontally with two ends connected to two fixed blocks. The corresponding ODE of this system is

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} q_i \\ p_i \end{pmatrix} = \begin{pmatrix} p_i/m \\ k_i\left(q_{i-1}-q_i\right)+k_{i+1}\left(q_{i+1}-q_i\right) \end{pmatrix}, \tag{11}$$

$i = 1, 2, \cdots, d, q_0 = q_{d+1} = 0$, where m_i and k_i are the

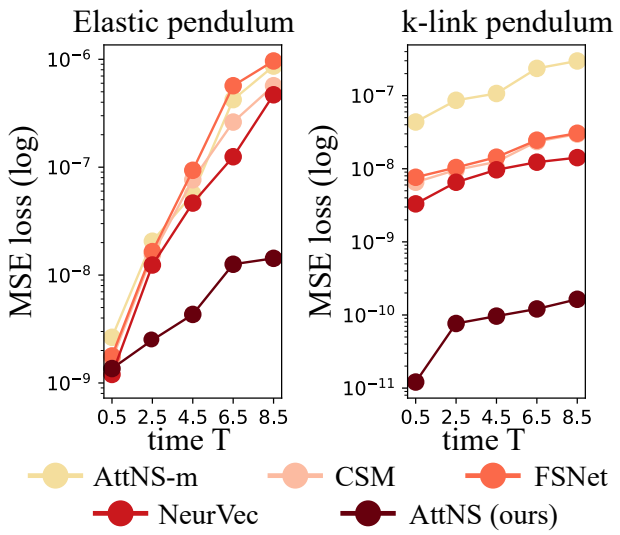

Figure 4: The simulation on different chaotic systems with step size $1e-1$. The Mean Squared Error (MSE) loss on the test set for (a) pendulum and (b) elastic pendulum.

mass of the t-th mass and force coefficient of the t-th spring, respectively. The momentum and the position of t-th mass are denoted as p_i and q_i. We adopt the coarse step size $\Delta t = 2e-1$ for the numerical solver and the fine step size $1e-3$ for training the AHS.

The experiment results at evaluation time T=20 are shown in Fig. 3. For the Euler method with low simulation accuracy, our AttNS achieves consistent performance with the state-of-the-art (SOTA) AHS for the spring-mass system in different dimensions. For other numerical solvers with higher accuracy, AttNS can better enhance the solver than others. Even for the spring-mass system with increasing dimensions, although the difficulty of the simulation increases, AttNS can still maintain the performance. In addition, we can observe that our AttNS can achieve similar performance as the SOTA AHS with less data size, showing that we are capable of training an efficient and robust AHS with a small amount of data. AttNS-m performs poorly in most different settings, which is consistent with the analysis of Section 3.

(2) **On different chaotic systems**. We reduce the amount of training data by 50% and use two chaotic dynamical systems to verify the effectiveness of our AttNS, i.e., the elastic and k-link pendulum. The elastic pendulum considers a ball without volume connected to an elastic rod. Under the effect of gravity and force of spring (Breitenberger & Mueller, 1981), the motion of the ball will be chaotic, and its ODE is

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\theta\\ r\\ \dot{\theta}\\ \dot{r}\end{pmatrix}=\begin{pmatrix}\dot{\theta}\\ \dot{r}\\ \frac{1}{r}(-g\sin\theta-\dot{\theta}\dot{r})\\ r\dot{\theta}^2-\frac{k}{m}(r-l_0)+g\cos\theta\end{pmatrix}, \tag{12}$$

where k, m, l_0, and g are related constants. There are two variables θ and r in Eq. (12). Specifically, r is the length of the spring, and θ is the angle between the spring and the vertical axis. For k-link pendulum, it considers K balls connected end to end with K rods under the effect of gravity (Lopes & Tenreiro Machado, 2017), and its ODE is

$$\mathrm{d}(\theta,\dot{\theta})/\mathrm{d}t=(\dot{\theta},\mathbf{A}^{-1}\mathbf{b}) \tag{13}$$

where $\theta=(\theta_1,\theta_2,\cdots,\theta_K)$ and θ_i is the angle between the ith rod and the vertical axis. Let $\mathbf{b}=(b_1,b_2,\cdots,b_K)$ and $b_i=-\sum_{j=1}^{K}\left[c(i,j)\dot{\theta}_j^2\sin(\theta_i-\theta_j)\right]-(K-i+1)g\sin\theta_i$. $\mathbf{A}$ is a $K\times K$ matrix and the element in $\mathbf{A}_{i,j}$ is $[K-\max(i,j)+1]\cos(\theta_i-\theta_j)$. The simulation results are shown in Fig. 4. From Fig. 4, we can observe that even for chaotic systems that are more difficult to solve, the proposed AttNS still maintains a better solution accuracy under limited data scenarios and outperforms other AHS.

5. Discussion

(1) **The convergence analysis of AttNS**. Our AttNS is a numerical solver, so we need to estimate its numerical convergence. Now we take our AttNS with step size $k\Delta t$ and the Euler method as an example to provide the convergence analysis for the proposed AttNS in Theorem 5.1.

Theorem 5.1. *We consider ODE $d\mathbf{u}/dt=\mathbf{f}(\mathbf{u}), \mathbf{u}(0)=\mathbf{c}_0$ and Euler method $\mathbf{u}_{n+1}=\mathbf{u}_n+\Delta t\mathbf{f}(\mathbf{u}_n)$. We assume that: (1) $\mathbf{f}$ is Lipschitz continuous with Lipschitz constant L, and (2) the second derivative of the true solution $\mathbf{u}$ is uniformly bounded by $M>0$, i.e., $\|\mathbf{u}''\|_\infty\leq M$ on $[0,T]$. Moreover, we assume that the attention module $Q[\cdot]$ in AttNS is Lipschitz continuous with Lipschitz constant $k\Delta tL_{att}$. For the solution of AttNS $\hat{\mathbf{u}}$ with step size $k\Delta t$, we have*

$$|\hat{\mathbf{u}}_N-\mathbf{u}(T)|\leq\alpha\Delta t+\beta\sqrt{\delta}, \tag{14}$$

where $\alpha=\frac{1}{2L}M\exp(2TL)$, $\beta=\frac{\sqrt{T}\exp(TL(1+L_{att}))}{\sqrt{L(1+L_{att})}}$ and δ is a error term about the training loss R_e. If the AttSlover can fit the training data well, i.e., $R_e\to 0$, the error $\delta\to 0$.

Proof. (See Appendix C). □

In fact, the conditions of Theorem 5.1 are mild. First, the Lipschitz condition of the attention module Q is provided by the analysis of Eq. (8) in Section 5(2), where we prove that it can be readily satisfied. Second, the boundedness of the true solution $\mathbf{u}$ can be directly verified for multiple physical systems (see Appendix). Theorem 5.1 can also be used to analyze the convergence of AttNS. For instance, some previous works (Du et al., 2019; Jacot et al., 2018) reveal that under certain mild conditions, gradient descent can allow a neural network to converge to a globally optimal

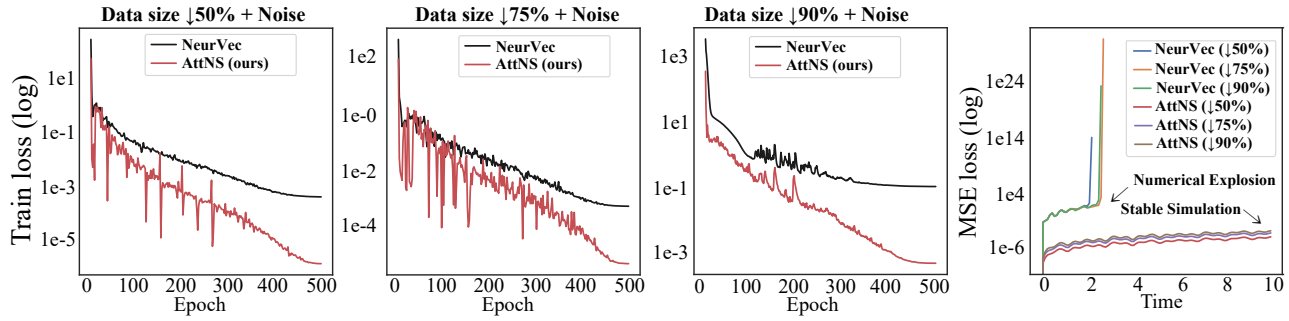

Figure 5: The noise attack experiments for the elastic pendulum under different data sizes. AttNS, with the attention mechanism, can better mitigate the adverse effects of noise than other AHS.

Table 1: The impact of the depth h and width d_1 of the attention module on the simulation performance. Rel. Infe. Speed is the relative inference speed of the network based on our proposed AttNS setting, i.e., $h = 2$ and $d_1 = 1024$.

	Spring-mass		k-link pendulum		elastic pendulum	
	MSE loss	**Rel. Infe. Speed**	**MSE loss**	**Rel. Infe. Speed**	**MSE loss**	**Rel. Infe. Speed**
$h = 2$ (ours)	2.58e-6	-	4.26e-9	-	5.41e-7	-
$h = 3$	2.49e-6 (↑ 3.61%)	↓ 88.49%	4.39e-9 (↓ 2.52%)	↓ 91.60%	5.28e-7 (↑ 2.46%)	↓ 90.89%
$h = 4$	2.55e-6 (↑ 1.18%)	↓ 93.95%	4.42e-9 (↓ 3.62%)	↓ 95.64%	5.58e-7 (↓ 3.05%)	↓ 95.33%
$d_1 = 512$	3.76e-5 (↓ 83.36%)	↑ 27.59%	2.56e-8 (↓ 93.14%)	↑ 15.36%	2.56e-6 (↓ 78.87%)	↑ 17.25%
$d_1 = 1024$ (ours)	2.58e-6	-	4.26e-9	-	5.41e-7	-
$d_1 = 2048$	2.41e-6 (↑ 7.05%)	↓ 26.48%	3.98e-9 (↑ 7.03%)	↓ 23.12%	4.83e-7 (↑ 12.01%)	↓ 24.63%

solution, where the loss R_e in Eq. (7) tends to 0. Then according to Theorem 5.1, we have $\delta \to 0$,

$$|\hat{\mathbf{u}}_N - \mathbf{u}(T)| \leq \underbrace{|\mathbf{u}_{Nk} - \mathbf{u}(T)|}_{\leq \Delta t \cdot M \cdot \exp(2TL)/2L} + \underbrace{|\hat{\mathbf{u}}_N - \mathbf{u}_{Nk}|}_{\leq \beta\sqrt{\delta} \to 0}. \quad (15)$$

and thus $|\hat{\mathbf{u}}_N - \mathbf{u}(T)| = \mathcal{O}(\Delta t)$. In this case, the AttNS with step size $k\Delta t$ can achieve the same accuracy as the Euler method with the step size Δt, whose global truncation error is also $\mathcal{O}(\Delta t)$ (Butcher, 2016). However, the evaluation speed of AttNS is approximately $\mathcal{O}(k)$ times greater than that of the Euler method in this situation.

(2) **The network architecture of $Q[\hat{S}|\phi]$.** We design the attention module structure $Q[\hat{S}|\phi]$ in Eq. (8) primarily because, on the one hand, it has a smaller number of parameters and enables faster inference speed, which ensures the computational efficiency. For example, in Eq. (2), if data $\mathbb{D}_f$ is generated by a fine step size Δt_f, then theoretically its evaluation speed is $\mathcal{O}(\Delta t_c/[(1+\epsilon)\Delta t_f]$ times (Huang et al., 2022c) that of the classical method Eq. (3), where $\epsilon > 0$ is related to the inference speed of the neural network. Therefore, the inference speed of the neural network in AHS is important.

On the other hand, it can be proven in Theorem 5.2 that this architecture possesses the desirable Lipschitz continuity property, which provides a strong guarantee for the convergence analysis of AttNS stated in Theorem 5.1. We will compare this attention module structure with other variants in Section 6.

Theorem 5.2. *For $x \in$ definition domain A, the attention module $Q(x) = \mathbf{W_2} \circ \mathbf{a} \circ \mathbf{W_1} x$, is Lipschitz continuous. $\mathbf{a}$ is rational activation function, i.e. $\mathbf{a}(x) = \frac{\sum_{i=1}^{3} a_i x^i}{\sum_{i=1}^{2} b_i x^i}$, where $a_i, b_i \in \mathbb{R}$.*

Proof. (See Appendix D). □

(3) **The the effectiveness of AttNS for generalization.** Now we extend our discussion of the data size, where some analyses have been given by Theorem 5.3 under the Euler method and some mild assumptions. Theorem 5.3 is based on the view of Vapnik-Chervonenkis theory (Vapnik, 1999). Compared to the general AHS, AttNS requires a smaller data size to achieve the same generalization error, which is consistent with the experimental results in Fig. 3.

Theorem 5.3. *Let Net$(\hat{S}|\phi, \mathbb{D}_f)$ and Net$(\mathbf{u}_n|\phi, \mathbb{D}_f)$, where $\hat{S} = S(\mathbf{f}, \mathbf{u}_n, \Delta t_c)$, be the correction term of general AHS and AttNS, respectively. For $\epsilon > 0$, when the data size is more than N', the empirical error of two methods satisfy $R_e(\phi|AHS) \leq \epsilon$ and $R_e(\phi|AttNS) \leq \epsilon$. For small enough $\epsilon_0 \ll \epsilon$ and Euler method, we have*

$$N(AttNS) \lesssim N(AHS), \quad (16)$$

where $N(*)$ is the lower bound of the data size that the generalization error of method $*$ can reach $\epsilon(1-\epsilon_0)^{-1}$.

Table 2: (Left) The impact of skip connection and considering Δt_c as a part of the input. (Right) The performance of other type network architectures for $Q[\hat{S}|\phi]$.

Benchmark	**w/** skip	**w/o** Δt_c	Ours	**Module**	**MSE (elastic)**	**MSE (k-link)**
Spring-mass	4.57e-4 (↓ 99.44%)	3.67e-6 (↓ 29.78%)	**2.58e-6**	LSTM (Hochreiter & Schmidhuber, 1997)	7.21e-06	1.35e-08
K-link	6.36e-6 (↓ 99.93%)	**4.18e-9** (↑ 1.91%)	4.26e-9	Transformer (Liu et al., 2021)	3.64e-04	2.65e-06
Elastic	1.39e-4 (↓ 99.61%)	1.65e-6 (↓ 67.21%)	**5.41e-7**	Ours	**5.41e-07**	**4.26e-09**

Proof. (See Appendix E). □

Although our proposed method does reduce the data size required by the SOTA method, the accuracy may not be sufficient for all the problems in the natural sciences and engineering. Therefore, if we need to solve a problem that requires high accuracy, it would be better to increase the training data size as much as possible to further improve the performance of the AI-enhance solver than only relying on the algorithmic design. This is because we can observe when the complete training data is used, the improvement brought by AttNS may be less than 10%-15%, which is reasonable since the AttNS is designed tailored to the data-insufficient scenario and we have not explicitly maximized the performance of AttNS when the data is sufficient. In the future, we will explore improving the AttNS in the data-sufficient scenario.

(4) **The the effectiveness of AttNS for robustness** . Our proposed method AttNS is inspired by the attention mechanism in ResNet. In Fig. 5, we empirically explore the robustness of AttNS by noise attack experiments under different data sizes. Injecting noise into the input is an explicit way of adding small perturbations that help us observe the robustness of the model. Specifically, in the training process, we can interfere with the training phase of AttNS by adding constant noise $\sigma = 1e-5$ to $\hat{\mathbf{u}}_n$, i.e., $\hat{\mathbf{u}}_n \leftarrow \hat{\mathbf{u}}_n + \sigma$ for all n in Eq. (10). Experimental results show that, compared with the baseline AHS, i.e., NeurVec, our proposed method can significantly regulate the noise, leading to smaller training loss and stable simulations. In contrast, NeurVec explodes numerically at the beginning of the simulation due to the effect of noise in all settings.

The above experiments illustrate that the proposed AttNS is robust enough to maintain a stable solution even in limited data scenarios.

(5) **Limitation** . From the experiment results in our paper, the proposed AttNS can achieve good enough simulation performance with less training data. However, AttNS does not completely prevent the solution from being disturbed under the inherent noise in the dataset. So we aim to alleviate the noise issue rather than solve it completely. If we want to further mitigate this issue, we may need more elaborate network structures and training settings to enhance the effectiveness of our AttNS.

The experiments in this paper only consider the ordinary differential equation (ODE) and not the benchmark for considering partial differential equations (PDE), this is because the ODE is more conducive to some of the theoretical analyses in this paper, whereas the complex PDE is not easy to analyze. Fortunately, the AHS can also be used for PDE solv (Dresdner et al., 2022), so this could be used as future work to use AttNS for PDE solving.

6. Ablation study

In this section, we perform several ablation studies on AttNS. In Table 1, we analyze the impact of the depth h and width d_1 for the attention module in Eq. (8). We observe that the depth h has little effect on the MSE loss in all benchmarks, but the model's performance is positively correlated with the width d_1. Therefore, to increase the inference speed, we choose a sufficiently shallow depth $h = 2$ and appropriate width $d_1 = 1024$ for AttNS.

Then we study popular residual structures for AttNS. From Table 2 (Left), the skip connection will bring significant negative impacts on simulation for both high-dimension linear systems and chaotic dynamical systems. This motivates us to adopt a simple stacking structure like Eq. (8) for our AttNS. Moreover, we analyze the form of the input of the attention module, where we take $\hat{S}$ as input instead of the complete integration term $\hat{S}\Delta t_c$ in Eq. (8).

In Table 2, we show that the input $\hat{S}$ has better performance than input $\hat{S}\Delta t_c$, except for k-link pendulum (similar performance). Furthermore, Table 2 (Right) demonstrates the performance of other network architectures for $Q[\hat{S}|\phi]$, highlighting the effectiveness of the attention module structure as defined in Eq. (8).

7. Conclusion

This paper discusses how to improve AHS for effective computation even in limited data scenarios. Using the dynamical system view of ResNet, we introduce the attention mechanism into the numerical solver and propose AttNS to help the generalization and robustness issue when the data is limited. Experimental results show the effectiveness of AttNS in improving various numerical solvers, where we also analyze the convergence, generalization, and robustness.

Acknowledgement

This work was supported by National Science and Technology Major Project (No.2021ZD0111601), National Natural Science Foundation of China (No.62325605, No.62272494), and Guangdong Basic and Applied Basic Research Foundation (No.2023A1515011374), and Guangzhou Science and Technology Program (No.2024A04J6365).

Impact Statement

The work presented in this paper aims to advance the field of scientific machine learning. Our work has many potential impacts in fields including physics, chemistry.

References

Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T. *Learning from data*, volume 4. AMLBook New York, 2012.

Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019.

Boullé, N., Nakatsukasa, Y., and Townsend, A. Rational neural networks. *Advances in Neural Information Processing Systems*, 33:14243–14253, 2020.

Breitenberger, E. and Mueller, R. D. The elastic pendulum: a nonlinear paradigm. *Journal of Mathematical Physics*, 22(6):1196–1210, 1981.

Bruno, O. P., Hesthaven, J. S., and Leibovici, D. V. Fc-based shock-dynamics solver with neural-network localized artificial-viscosity assignment. *Journal of Computational Physics: X*, 15:100110, 2022.

Butcher, J. C. *Numerical methods for ordinary differential equations*. John Wiley and Sons, 2016.

Cai, S., Mao, Z., Wang, Z., Yin, M., and Karniadakis, G. E. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021.

Chang, B., Meng, L., Haber, E., Tung, F., and Begert, D. Multi-level residual networks from dynamical systems view. *arXiv preprint arXiv:1710.10348*, 2017.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31, 2018.

Chen, Z., Zhang, J., Arjovsky, M., and Bottou, L. Symplectic recurrent neural networks. In *International Conference on Learning Representations*, 2019.

Chen, Z., Zhang, J., Arjovsky, M., and Bottou, L. Symplectic recurrent neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkgYPREtPr.

Cheuk, K. W., Luo, Y.-J., Benetos, E., and Herremans, D. Revisiting the onsets and frames model with additive attention. *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2021.

Choudhary, A., Lindner, J. F., Holliday, E. G., Miller, S. T., Sinha, S., and Ditto, W. L. Physics-enhanced neural networks learn order and chaos. *Physical Review E*, 101 (6):062207, 2020.

Dahlquist, G. Convergence and stability in the numerical integration of ordinary differential equations. *Mathematica Scandinavica*, pp. 33–53, 1956.

Dresdner, G., Kochkov, D., Norgaard, P. C., Zepeda-Nunez, L., Smith, J., Brenner, M., and Hoyer, S. Learning to correct spectral methods for simulating turbulent flows. *Transactions on Machine Learning Research*, 2022.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1eK3i09YQ.

Dupont, E., Doucet, A., and Teh, Y. W. Augmented neural odes. *Advances in neural information processing systems*, 32, 2019.

Ferziger, J. H., Perić, M., and Street, R. L. *Computational methods for fluid dynamics*, volume 3. Springer, 2002.

Frezat, H., Le Sommer, J., Fablet, R., Balarac, G., and Lguensat, R. A posteriori learning for quasi-geostrophic turbulence parametrization. *Journal of Advances in Modeling Earth Systems*, 14(11):e2022MS003124, 2022.

Gao, C., Ye, H., Cao, F., Wen, C., Zhang, Q., and Zhang, F. Multiscale fused network with additive channel-spatial attention for image segmentation. *Knowl. Based Syst.*, 214:106754, 2021.

Geneva, N. and Zabaras, N. Transformers for modeling physical systems. *Neural Networks*, 146:272–289, 2022.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Guo, J., Ma, X., Sansom, A., McGuire, M., Kalaani, A., Chen, Q., Tang, S., Yang, Q., and Fu, S. Spanet: Spatial pyramid attention network for enhanced image recognition. In *2020 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6. IEEE, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

Hemmasian, A. and Barati Farimani, A. Reduced-order modeling of fluid flows with transformers. *Physics of Fluids*, 35(5), 2023.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

Huang, W., Xu, H., and Yu, Y. Mrp-net: Seizure detection method based on modified recurrence plot and additive attention convolution neural network. *SSRN Electronic Journal*, 2022a.

Huang, Z., Liang, S., Liang, M., and Yang, H. Dianet: Dense-and-implicit attention network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4206–4214, 2020.

Huang, Z., Liang, S., Liang, M., He, W., and Lin, L. Layer-wise shared attention network on dynamical system perspective. *arXiv preprint arXiv:2210.16101*, 2022b.

Huang, Z., Liang, S., Zhang, H., Yang, H., and Lin, L. Accelerating numerical solvers for large-scale simulation of dynamical system via neurvec. *arXiv preprint arXiv:2208.03680*, 2022c.

Huang, Z., Liang, M., Qin, J., Zhong, S., and Lin, L. Understanding self-attention mechanism via dynamical system perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (¡strong¿ICCV¡/strong¿)*, 2023a.

Huang, Z., Liang, S., Zhang, H., Yang, H., and Lin, L. On fast simulation of dynamical system with neural vector enhanced numerical solver. *Scientific Reports*, 13(1): 15254, 2023b.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

Ji, W., Qiu, W., Shi, Z., Pan, S., and Deng, S. Stiff-pinn: Physics-informed neural network for stiff chemical kinetics. *The Journal of Physical Chemistry A*, 125(36): 8098–8106, 2021.

Kidger, P., Chen, R. T., and Lyons, T. J. " hey, that's not an ode": Faster ode adjoints via seminorms. In *ICML*, pp. 5443–5452, 2021.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., and Hoyer, S. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lehtonen, J. The lambert w function in ecological and evolutionary models. *Methods in Ecology and Evolution*, 7(9):1110–1118, 2016.

Li, X., Xu, Q., Chen, X., and Li, C. Additive attention for cnn-based classification. *2021 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 55–59, 2021a.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Liang, S., Huang, Z., Liang, M., and Yang, H. Instance enhancement batch normalization: An adaptive regulator of batch noise. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4819–4827, 2020.

Liang, S., Jiang, S. W., Harlim, J., and Yang, H. Solving PDEs on unknown manifolds with machine learning. *arXiv preprint arXiv:2106.06682*, 2021.

Liang, S., Huang, Z., and Zhang, H. Stiffness-aware neural network for learning Hamiltonian systems. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uVXEKeqJbNa.

List, B., Chen, L.-W., and Thuerey, N. Learned turbulence modelling with differentiable fluid solvers: physics-based loss functions and optimisation horizons. *Journal of Fluid Mechanics*, 949:A25, 2022.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Lopes, A. M. and Tenreiro Machado, J. Dynamics of the N-link pendulum: a fractional perspective. *International Journal of Control*, 90(6):1192–1200, 2017.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via DeepONet based on the

universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

Lu, Y., Zhong, A., Li, Q., and Dong, B. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3282–3291. PMLR, 2018.

Meunier, L., Delattre, B. J., Araujo, A., and Allauzen, A. A dynamical system perspective for lipschitz neural networks. In *International Conference on Machine Learning*, pp. 15484–15500. PMLR, 2022.

Mishra, S. A machine learning framework for data driven acceleration of computations of differential equations. *arXiv preprint arXiv:1807.09519*, 2018.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017.

Qin, Z., Zhang, P., Wu, F., and Li, X. Fcanet: Frequency channel attention networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 783–792, 2021.

Queiruga, A. F., Erichson, N. B., Taylor, D., and Mahoney, M. W. Continuous-in-depth neural networks. *arXiv preprint arXiv:2008.02389*, 2020.

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

Rodriguez-Torrado, R., Ruiz, P., Cueto-Felgueroso, L., Green, M. C., Friesen, T., Matringe, S., and Togelius, J. Physics-informed attention-based neural network for hyperbolic partial differential equations: application to the buckley–leverett problem. *Scientific reports*, 12(1): 7557, 2022.

San, O. and Maulik, R. Extreme learning machine for reduced order modeling of turbulent geophysical flows. *Physical Review E*, 97(4):042322, 2018.

Shampine, L. F. *Numerical solution of ordinary differential equations*. Routledge, 2018.

Subel, A., Chattopadhyay, A., Guan, Y., and Hassanzadeh, P. Data-driven subgrid-scale modeling of forced burgers turbulence using deep learning with generalization to higher reynolds numbers via transfer learning. *Physics of Fluids*, 33(3), 2021.

Takamoto, M., Alesiani, F., and Niepert, M. Learning neural pde solvers with parameter-guided channel attention. *arXiv preprint arXiv:2304.14118*, 2023.

Um, K., Brand, R., Fei, Y. R., Holl, P., and Thuerey, N. Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. *Advances in Neural Information Processing Systems*, 33:6111–6122, 2020.

Vapnik, V. *The nature of statistical learning theory*. Springer science & business media, 1999.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, Q., Wu, B., Zhu, P., Li, P., and Hu, Q. Eca-net: Efficient channel attention for deep convolutional neural networks. In *CVPR*, 2020.

Wang, X., Xue, W., Han, Y., and Yang, G. Efficient climate simulation via machine learning method. *arXiv preprint arXiv:2209.08151*, 2022.

Weinan, E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 1(5):1–11, 2017.

Woo, S., Park, J., Lee, J.-Y., and Kweon, I. S. Cbam: Convolutional block attention module. In *ECCV*, pp. 3–19, 2018.

Wu, C., Wu, F., Qi, T., and Huang, Y. Fastformer: Additive attention can be all you need. *ArXiv*, abs/2108.09084, 2021.

Xu, Y., Liu, S., and Lin, L. Fsnet: A fast simulation algorithm for chaotic dynamical systems. 2023. URL https://github.com/small-whirlwind/FSNet.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

Zhong, S., Huang, Z., Wen, W., Qin, J., and Lin, L. Asr: Attention-alike structural re-parameterization. *ArXiv*, abs/2304.06345, 2023.

Zhu, M., Chang, B., and Fu, C. Convolutional neural networks combined with runge–kutta methods. *Neural Computing and Applications*, pp. 1–15, 2022.

A. The details of the proposed algorithm.

Table 3: Summary of the datasets mentioned in this paper.

Benchmark	Type	Dimension	Data size	Step size	Generative Method	Evaluation time T
Spring-mass	Train	*	5k	1e-3	*	20
Spring-mass	Validation	*	0.1k	1e-5	RK4	20
Spring-mass	Test	*	5k	1e-5	RK4	20
2-link pendulum	Train	4	20k	1e-3	RK4	10
2-link pendulum	Validation	4	1k	1e-5	RK4	10
2-link pendulum	Test	4	10k	1e-5	RK4	10
Elastic pendulum	Train	4	20k	1e-3	RK4	50
Elastic pendulum	Validation	4	1k	1e-5	RK4	50
Elastic pendulum	Test	4	10k	1e-5	RK4	50

The details of dataset. We summarize the information on training, validation, and test data for all benchmarks in this paper in Table 3. We obtain the discrete solutions every step size up to the model time T and the evaluation time for all experimental results of MSE loss in this paper is T. The Generative method of Spring-mass during training depends on the numerical solver used in Table 1 in main text. Moreover, the dimension of the Spring-mass system also depends on d in Table 1 in main text, and if $d = 20$, the dimension is $2d = 40$. "RK4" denotes 4th order Runge-Kutta method.

Table 4: The initialization of different benchmarks. "Uniform random" means that the variables are sampled with uniform distribution of a given range. "Constant" means the variable is initialized as a constant. The dimension of $\mathbf{p}$ and $\mathbf{q}$ in Spring-mass system depends on d in Table 1 in main text, and if $d = 20$, their dimension are $d = 20$.

Task	**Variable**	**Dimension**	**Type**	**Range of initialization**	**Model input?**
Spring-mass	$\mathbf{p}$	*	Uniform random	$[-2.5, 2.5]^{20}$	✓
Spring-mass	$\mathbf{q}$	*	Uniform random	$[-2.5, 2.5]^{20}$	✓
Elastic pendulum	θ	1	Uniform random	$[0, \pi/8]$	✓
Elastic pendulum	r	1	Constant	10	✓
Elastic pendulum	$\dot{\theta}$	1	Constant	0	✓
Elastic pendulum	$\dot{r}$	1	Constant	0	✓
Elastic pendulum	l_0	1	Constant	10	
Elastic pendulum	g	1	Constant	9.8	
Elastic pendulum	k	1	Constant	40	
Elastic pendulum	m	1	Constant	1	
2-link pendulum	θ	2	Uniform random	$[0, \pi/8]^2$	✓
2-link pendulum	$\dot{\theta}$	2	Constant	0	✓
2-link pendulum	m	1	Constant	1	
2-link pendulum	g	1	Constant	9.8	

Numerical solvers. In this paper, we consider four forward numerical solvers to validate the effectiveness of our proposed AttNS, including the Euler method, Improved Euler method, 3rd and 4th order Runge-Kutta methods. In this section, we introduce these solvers. As mentioned in Eq. (2) in main text, these solvers have different integration terms $S(f, u_n, \Delta t)$.

For 3rd order Runge-Kutta method ,the coefficients $\lambda_1 = \lambda_3 = \frac{1}{6}$ and $\lambda_2 = \frac{2}{3}$. Besides, $K_1 = \mathbf{f}(\mathbf{u}_n)$, $K_2 = \mathbf{f}(\mathbf{u}_n + \frac{\Delta t}{2} K_1)$ and $K_3 = \mathbf{f}(\mathbf{u}_n - \Delta t K_1 + 2\Delta t K_2)$. For the 4th order Runge-Kutta method, $\beta_1 = \beta_4 = \frac{1}{6}$ and $\beta_2 = \beta_3 = \frac{1}{3}$. Moreover, $J_1 = \mathbf{f}(\mathbf{u}_n)$, $J_2 = \mathbf{f}(\mathbf{u}_n + \frac{\Delta t}{2} J_1)$, $J_3 = \mathbf{f}(\mathbf{u}_n + \frac{\Delta t}{2} J_2)$ and $J_4 = \mathbf{f}(\mathbf{u}_n + \Delta t J_3)$. Generally speaking, 4th order Runge-Kutta method has the smallest Global truncation error $\mathcal{O}(\Delta t^4)$, i.e., 4th order Runge-Kutta method has the highest accuracy.

Table 5: The integration terms for different kinds of numerical solvers with step size Δt.

Numerical solver	Integration term $S(f, u_n, \Delta t)$	Global truncation error
Euler	$\Delta t \mathbf{f}(\mathbf{u}_n)$	$\mathcal{O}(\Delta t)$
Improved Euler	$\frac{\Delta t}{2}[\mathbf{f}(\mathbf{u}_n) + \mathbf{f}(\mathbf{u}_n + \Delta t \mathbf{f}(\mathbf{u}_n))]$	$\mathcal{O}(\Delta t^2)$
3rd order Runge-Kutta	$\Delta t(\lambda_1 K_1 + \lambda_2 K_2 + \lambda_3 K_3)$	$\mathcal{O}(\Delta t^3)$
4th order Runge-Kutta	$\Delta t(\beta_1 J_1 + \beta_2 J_2 + \beta_3 J_3 + \beta_4 J_4)$	$\mathcal{O}(\Delta t^4)$

However, due to its most complex integration term $\Delta t(\beta_1 J_1 + \beta_2 J_2 + \beta_3 J_3 + \beta_4 J_4)$, its simulation speed is the slowest among the four numerical solvers in Table 5.

B. The motivation and challenge behind the proof of our Theorems

B.1. Preliminaries

This work focuses on the numerical solvers for differential equations. The stability (or robustness) and convergence are two essential metrics for numerical solvers. Specifically, we introduce the definition of the stability and convergence for ODE solvers as follows (Ferziger et al., 2002; Dahlquist, 1956):

Stability refers to the sensitivity of the solver to initial conditions, parameters, and round-off errors during computation. A stable solver can produce reliable results even if the input conditions vary slightly with "noise". If a numerical method is unstable, it may produce unreasonable results or even lead to computation failure.

Convergence refers to the ability of numerical computation to approach the true solution as the step size (Δt) decreases. A convergent solver will produce increasingly accurate results, and the error will tend to zero as the step size approaches zero. If a solver does not converge, it will not produce accurate results even with a small step size.

The above preliminaries on stability and convergence highlight that these two metrics are distinct for numerical solvers of differential equations. That's why we separately discuss stability and convergence by Theorem 2.1 and Theorem 5.1, respectively. Specifically,

For stability, Theorem 2.1 investigates the ability of attention mechanisms to regulate noise and enhance model robustness, inspiring us to introduce attention mechanisms in numerical solvers.

For convergence, Theorem 5.1 examines the convergence of the proposed solver by analyzing approximation errors, and provides the convergence rate of the solver under well-trained neural networks.

Numerical solvers that have both good stability and convergence can obtain good solutions in different scenarios of differential equations. Moreover, since the AI-enhanced numerical solver is still exactly a numerical solver but only enhanced by a neural network, when we only have a small size of training data, the stability and convergence of such kind of solver may not be sufficiently guaranteed, and thus we must have a new kind of AI-enhanced numerical solver that can have better theoretical guarantee on the stability and convergence such that we may better offset the adverse effect on stability and convergence than other existing solvers when the size of the training data is small. Theorem 2.1 and Theorem 5.1 respectively reveal the potential for the proposed AttNS to become a sufficiently good AI-enhanced numerical solver from these two perspectives, and we also empirically show that our AttNS performs much better than other solvers when the size of training data is small. The further theoretical analysis can be the future work to enhance the existing AttNS.

B.2. The technical novelty and difficulty in proving theorem

We summarize the difficulty of proving Theorem 5.1 as follows:

1. Theorem 5.1 is used to analyze the convergence of a numerical solver integrated with attention mechanism. To the best of our knowledge, we are the first to introduce attention mechanisms into numerical solvers, and hence there are not many related proof techniques in the existing literature that can be used for analyzing the convergence of the numerical solver that incorporates attention modules.

2. In proving Theorem 5.1, our goal is to make the proof approach versatile and generalizable, i.e., we want to design a

unified approach that can be used to prove Theorem 5.1 for many attention modules Q that satisfy certain conditions. This requirement naturally adds difficulty on constructing our proof, but will provide us more general theoretical evidence for improving attention modules Q in the future.

3. To bridge the gap between theory and practice, it is challenging to set reasonable and mild conditions that can make Theorem 5.1 applicable for analyzing the convergence of solvers with specific empirical designs, such as our AttNS.

4. Since the solver we considered is AI-enhanced numerical, we must non-trivially connect the concepts of the numerical solver and the neural network to prove the convergence. However, the concept of the numerical solver and neural network both separately involve many different factors and perspectives in their own theoretical analyses, and the combination of these two concepts naturally introduces more factors, which greatly interferes with our target that we want to prove in Theorem 5.1. Therefore, we need to design a proof approach that can disentangle and eliminate interference factors that are irrelevant to our proof goal, enabling us to provide a clean proof that directly addresses our proof target for Theorem 5.1.

We also summarize the novelty of proving Theorem 5.1 as follows:

1. **Disentanglement.** As discussed in "Difficulty," in order to eliminate the complex interference factors in the theoretical analysis of numerical solvers and neural networks, we skillfully disentangle the convergence of AttNS $|\hat{\mathbf{u}}_{Nk} - \mathbf{u}(T)|$ into the convergence of the original numerical method $|\mathbf{u}_{Nk} - \mathbf{u}(T)|$ and the performance of the attention module $|\hat{\mathbf{u}}_N - \mathbf{u}_{Nk}|$ using the triangle inequality in the proof, i.e.,

$$|\hat{\mathbf{u}}_{Nk} - \mathbf{u}(T)| \leq \underbrace{|\mathbf{u}_{Nk} - \mathbf{u}(T)|}_{\leq \Delta t \cdot M \cdot \exp(2TL)/2L} + \underbrace{|\hat{\mathbf{u}}_N - \mathbf{u}_{Nk}|}_{\leq \beta\sqrt{\delta} \to 0}, \tag{17}$$

Since $|\mathbf{u}_{Nk} - \mathbf{u}(T)|$ and module training are independent, traditional mathematical methods can be used to analyze $|\mathbf{u}_{Nk} - \mathbf{u}(T)|$, while $|\hat{\mathbf{u}}_N - \mathbf{u}_{Nk}|$ can be proven using relevant methods for neural network convergence. This disentanglement successfully bridges the convergence of AttNS and the training performance of the neural network, and may also inspire future theoretical proofs of introducing neural networks into differential equation solvers.

2. **Versatility**. As mentioned in "Difficulty," our proof should be applicable to many different attention modules. Specifically, we make our proof be versatile by considering the Lipschitz continuity of the attention module, which is a very general property that can be used to formally evaluate different and new kinds of attention module in the future. We also show in Lemma D.3 that the MLP-alike attention module with a rational activation function in AttNS from Eq.(9) satisfies the Lipschitz continuity condition. The convergence of attention modules with Lipschitz conditions can be estimated uniformly using Theorem 5.1.

3. **Instructiveness.** The Lipschitz condition can guide the design of new attention modules in numerical solvers in the future and may be a necessary condition for the effectiveness of the solver. Specifically, attention modules that satisfy the Lipschitz condition, such as the "simple" MLP-alike structure with rational activation function shown in Eq.(9), can effectively solve differential equations under the proposed AttNS framework. However, some "complex" attention module structures, such as transformer and LSTM,may have poor prediction due to the fact that they generally lack the Lipschitz continuity condition, which is consistent with the experimental observation in the main text.

C. The proof of Theorem 5.1

Theorem 5.1. We consider ODE $\mathrm{d}\mathbf{u}/\mathrm{d}t = \mathbf{f}(\mathbf{u}), \mathbf{u}(0) = \mathbf{c}_0$ and Euler method $\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t \mathbf{f}(\mathbf{u}_n)$. We assume that (1) $\mathbf{f}$ is Lipschitz continuous with Lipschitz constant L and (2) the second derivative of the true solution $\mathbf{u}$ is uniformly bounded by $M > 0$, i.e., $\|\mathbf{u}''\|_\infty \leq M$ on $[0, T]$. Moreover, we assume that the attention module $Q[\cdot]$ in AttNS is Lipschitz continuous with Lipschitz constant $k\Delta t L_{\text{att}}$. For the solution of AttNS $\hat{\mathbf{u}}$ with step size $k\Delta t$, we have

$$|\hat{\mathbf{u}}_N - \mathbf{u}(T)| \leq \alpha \Delta t + \beta \sqrt{\delta}, \tag{18}$$

where $\alpha = \frac{1}{2L} M \exp(2TL)$, $\beta = \frac{\sqrt{T}\exp(TL(1+L_{\text{att}})}{\sqrt{L(1+L_{\text{att}})}}$ and δ is a error term about the training of AttSlover. If the AttSlover can fit the training data well, the error $\delta \to 0$.

Lemma C.1. *If the assumptions (1) and (2) in Theorem 5.1 hold, for Euler method $\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t \mathbf{f}(\mathbf{u}_n)$, we have*

$$|\mathbf{u}_{Nk} - \mathbf{u}(T)| \leq \frac{M \exp(2TL)}{2L} \Delta t, \tag{19}$$

where $0 = t_0 < t_1 < \cdots < t_{Nk} = T$ be uniform points on $[0, T]$ and $\Delta t = \frac{T}{Nk}$.

Lemma C.2. *For any $n \in \mathbb{N}_+$ and $x > 0$, we have*

$$(1 + x)^n \leq \exp(nx), \tag{20}$$

Proof. (For Lemma C.2). According to the Taylor expansion, we have

$$\exp(nx) = \sum_{i=0}^{\infty} \frac{(nx)^i}{i!} \geq \sum_{i=0}^{\infty} n^i \frac{x^i}{i!} \geq \sum_{i=0}^{\infty} \prod_{j=0}^{i-1} (n-j) \frac{x^i}{i!} = \sum_{i=0}^{\infty} C_n^i x^i = (1+x)^n \tag{21}$$

□

Proof. (For Theorem 5.1.) Let's consider the discretization for time interval $[0, T]$ as shown in Fig.6, and $E_n := \hat{\mathbf{u}}_n - \mathbf{u}_{nk}$

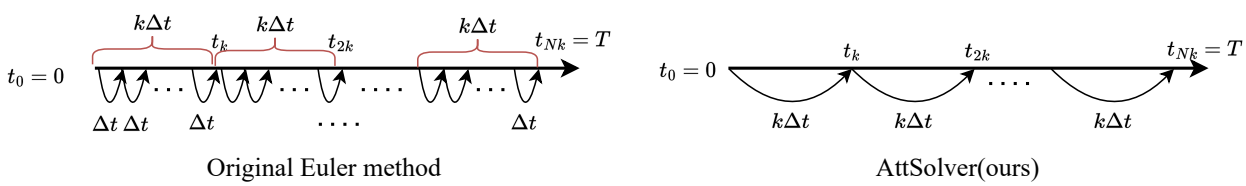

Figure 6: The discretization of the original Euler method (Left) and our AttNS (Right).

$$\begin{aligned} |\hat{\mathbf{u}}_{Nk} - \mathbf{u}(T)| &\leq |\mathbf{u}_{Nk} - \mathbf{u}(T)| + |\hat{\mathbf{u}}_N - \mathbf{u}_{Nk}| && \text{Since } |a-b| \leq |a| + |b| \\ &= \frac{M \exp(2TL)}{2L} \Delta t + E_N && \text{Since Lemma D.1} \end{aligned}$$

Next we estimate the upper bound of E_N. For any $n \geq 0$ we have

$$\begin{aligned} \hat{\mathbf{u}}_{n+1} - \mathbf{u}_{k(n+1)} &= \hat{\mathbf{u}}_n + \mathbf{f}(\hat{\mathbf{u}}_n)(k\Delta t) + Q(\mathbf{f}(\hat{\mathbf{u}}_n)|\phi) - \mathbf{u}_{k(n+1)} \\ &= \hat{\mathbf{u}}_n - \mathbf{u}_{kn} + \big(\mathbf{f}(\hat{\mathbf{u}}_n) - \mathbf{f}(\mathbf{u}_{kn})\big)(k\Delta t) + Q(\mathbf{f}(\hat{\mathbf{u}}_n)|\phi) - Q(\mathbf{f}(\mathbf{u}_{kn})|\phi) - (k\Delta t)V_n. \end{aligned} \tag{22}$$

where V_n is an error term about training, i.e.,

$$V_n = \frac{1}{k\Delta t}(\mathbf{u}_{k(n+1)} - \mathbf{u}_{kn} - \mathbf{f}(\mathbf{u}_{kn})k\Delta t - Q(\mathbf{f}(\mathbf{u}_{nk})|\phi)). \tag{23}$$

Next, from Lipschitz conditions and the triangle inequality, we have

$$\begin{aligned}|\hat{\mathbf{u}}_{n+1}-\mathbf{u}_{k(n+1)}| &\leq |\hat{\mathbf{u}}_n-\mathbf{u}_{kn}|+L|\hat{\mathbf{u}}_n-\mathbf{u}_{kn}|(k\Delta t)+k\Delta t L_{\text{att}}|\mathbf{f}(\hat{\mathbf{u}}_n)-\mathbf{f}(\mathbf{u}_{kn})|+(k\Delta t)|V_n| \\ &\leq |\hat{\mathbf{u}}_n-\mathbf{u}_{kn}|+L|\hat{\mathbf{u}}_n-\mathbf{u}_{kn}|(k\Delta t)+k\Delta t L_{\text{att}}\cdot L|\hat{\mathbf{u}}_n-\mathbf{u}_{kn}|+(k\Delta t)|V_n| \\ &= (1+k\Delta tL+k\Delta tL_{\text{att}}L)|\hat{\mathbf{u}}_n-\mathbf{u}_{kn}|+(k\Delta t)|V_n|.\end{aligned} \tag{24}$$

Let $w=(1+k\Delta tL+k\Delta tL_{NV}L)$. Eq.(24) can be rewritten as

$$\begin{aligned}|E_{n+1}| &\leq w|E_n|+(k\Delta t)|V_n| \\ &\leq w(w|E_{n-1}|+(k\Delta t)|V_{n-1}|)+(k\Delta t)|V_n| \\ &= w^2|E_{n-1}|+w(k\Delta t)|V_{n-1}|+(k\Delta t)|V_n| \\ &\leq w^{n+1}|E_0|+(k\Delta t)\sum_{i=0}^{n}w^i|V_{n-i}| = (k\Delta t)\sum_{i=0}^{n}w^i|V_{n-i}|,\end{aligned} \tag{25}$$

where $E_0=0$ as $E_0=\hat{\mathbf{u}}_0-\mathbf{u}_0=\mathbf{c}_0-\mathbf{c}_0=0$. Let

$$\delta=\frac{1}{N}(\|V_0\|_2^2+\|V_1\|_2^2+...+\|V_{N-1}\|_2^2) \tag{26}$$

By the Cauchy inequality and Eq.(25),

$$\begin{aligned}|E_N| &= (k\Delta t)\sum_{i=0}^{N-1}w^i|V_{N-1-i}| \\ &\leq (k\Delta t)(\sum_{i=0}^{N-1}w^{2i})^{\frac{1}{2}}(\sum_{i=0}^{N-1}|V_{N-1-i}|^2)^{\frac{1}{2}} && \text{Since Cauchy inequality} \\ &= (k\Delta t)\sqrt{(w^{2N}-1)/(w^2-1)}\sqrt{N\delta}. && \text{Since Eq.(26)}\end{aligned}$$

Next, we simplify the term $\sqrt{(w^{2N}-1)/(w^2-1)}$. Note that $w=(1+k\Delta tL+k\Delta tL_{\text{att}}L)\geq 1$, hence

$$w^2-1\geq w-1. \tag{27}$$

Therefore,

$$\begin{aligned}\sqrt{(w^{2N}-1)/(w^2-1)} &\leq \sqrt{[(1+k\Delta tL+k\Delta tL_{\text{att}}L)^{2N}]/[k\Delta tL+k\Delta tL_{\text{att}}L]} && \text{Since Eq.(27)} \\ &\leq \sqrt{\frac{\exp(2Nk\Delta tL(1+L_{\text{att}}))}{k\Delta tL(1+L_{\text{att}})}} && \text{Since Lemma C.2}\end{aligned}$$

Therefore,

$$|E_N|\leq (k\Delta t)\frac{\exp(TL(1+L_{NV}))}{\sqrt{k\Delta tL(1+L_{\text{att}})}}\sqrt{N\delta}=\frac{\sqrt{T}\exp(TL(1+L_{\text{att}}))}{\sqrt{L(1+L_{\text{att}})}}\sqrt{\delta}.$$

Hence, we have

$$|\hat{\mathbf{u}}_N-\mathbf{u}(T)|\leq \alpha\Delta t+\beta\sqrt{\delta}, \tag{28}$$

where $\alpha=\frac{1}{2L}M\exp(2TL)$, $\beta=\frac{\sqrt{T}\exp(TL(1+L_{\text{att}})}{\sqrt{L(1+L_{\text{att}})}}$. Since Eq.(23), if the AttSlover can fit the training data well, $\|V_n\|_2\to 0$, and we have $\delta\to 0$. □

D. Lipschitz continuous of our $Q[\cdot]$

Lemma D.1. *For $x \in [a, b]$, the polynomial function $P(x) = \sum_{k=0}^{n} c_k x^k$ is Lipschitz continuous.*

Proof. For all $x, y \in [a, b]$,we have $|x^k - y^k| = (x - y) \sum_{i=0}^{k-1} x^i y^{k-1-i} \leq |x - y| \cdot k \cdot \max(a, b)^{k-1}$. Therefore, we have

$$|P(x) - P(y)| \leq \sum_{k=0}^{n} |c_k||x^k - y^k| \leq \underbrace{\{\sum_{k=0}^{n} k \cdot |c_k| \cdot \max(a, b)^{k-1}\}}_{\text{Lipschitz constant}} \cdot |x - y|. \tag{29}$$

□

Lemma D.2. *For $x \in$ definition domain A , the rational activation $\mathbf{a}(x) = \frac{\sum_{i=1}^{3} a_i x^i}{\sum_{i=1}^{2} b_i x^i}$ is Lipschitz continuous.*

Proof. For all $x_1, x_2 \in A$,Let $p(x) = \sum_{i=1}^{3} a_i x^i$ and $q(x) = \sum_{i=1}^{2} b_i x^i$. For all $x_1, x_2 \in A$,we have

$$\begin{aligned}
|\mathbf{a}(x_1) - \mathbf{a}(x_2)| &= |\frac{p(x_1)}{q(x_1)} - \frac{p(x_2)}{q(x_2)}| && (30)\\
&= |\frac{p(x_1)q(x_2) - q(x_1)p(x_2)}{q(x_1)q(x_2)}| && (31)\\
&= |\frac{p(x_1)q(x_2) - p(x_1)q(x_1) + p(x_1)q(x_1) - q(x_1)p(x_2)}{q(x_1)q(x_2)}| && (32)\\
&\leq \frac{|p(x_1)|}{|q(x_1)q(x_2)|}|q(x_2) - q(x_1)| + \frac{|q(x_1)|}{|q(x_1)q(x_2)|}|p(x_2) - p(x_1)|. && (33)
\end{aligned}$$

Note that, since the polynomial function $p(x)$ and $q(x)$ are continuous in A ,they are bounded in A . Let $\max(|p(x)|, |q(x)|) \leq M$ and $\min(|p(x)|, |q(x)|) \geq N$. Meoreover, from Lemma D.1, for all $x_1, x_2 \in A$, we assume $|q(x_2) - q(x_1)| \leq L_q|x_2 - x_1|$ and $|p(x_2) - p(x_1)| \leq L_p|x_2 - x_1|$.

Therefore, we have

$$\begin{aligned}
|\mathbf{a}(x_1) - \mathbf{a}(x_2)| &\leq \frac{|p(x_1)|}{|q(x_1)q(x_2)|}|q(x_2) - q(x_1)| + \frac{|q(x_1)|}{|q(x_1)q(x_2)|}|p(x_2) - p(x_1)| && (34)\\
&\leq \frac{M}{N^2}|q(x_2) - q(x_1)| + \frac{M}{N^2}|p(x_2) - p(x_1)| && (35)\\
&\leq \underbrace{\frac{M}{N^2} \cdot (L_p + L_q)}_{\text{Lipschitz constant}} \cdot |x_1 - x_2|. && (36)
\end{aligned}$$

□

Lemma D.3. *For $x \in$ definition domain A , the attention module shown in Eq.(9) in main text, i.e., $Q(x) = \mathbf{W_2} \circ \mathbf{a} \circ \mathbf{W_1} x$, is Lipschitz continuous. $\mathbf{a}$ is rational activation function, i.e. $\mathbf{a}(x) = \frac{\sum_{i=1}^{3} a_i x^i}{\sum_{i=1}^{2} b_i x^i}$, where $a_i, b_i \in \mathbb{R}$.*

Proof. For all $x, y \in A$, from Lemma D.2, there exist a constant L_1 s.t. $||\mathbf{a}(x) - \mathbf{a}(y)|| \leq L_1||x - y||$. Therefore,

$$\begin{aligned}
||\mathbf{a} \circ \mathbf{W_1}(x) - \mathbf{a} \circ \mathbf{W_1}(y)|| &\leq L_1||\mathbf{W_1}(x) - \mathbf{W_1}(y)|| && (37)\\
&\leq L_1||\mathbf{W_1}|| \cdot ||x - y||. && (38)
\end{aligned}$$

Moreover, we consider the attention module $Q[\cdot]$, and we have

$$\begin{aligned}
||Q(x) - Q(y)|| &= ||\mathbf{W_2} \circ \mathbf{a} \circ \mathbf{W_1}(x) - \mathbf{W_2} \circ \mathbf{a} \circ \mathbf{W_1}(y)|| && (39)\\
&\leq ||\mathbf{W_1}|| \cdot ||\mathbf{a} \circ \mathbf{W_1}(x) - \mathbf{a} \circ \mathbf{W_1}(y)|| && (40)\\
&\leq \underbrace{L_1 \cdot ||\mathbf{W_2}|| \cdot ||\mathbf{W_1}||}_{\text{Lipschitz constant}} \cdot ||x - y||. && (41)
\end{aligned}$$

Therefore, the attention module in Eq.(9) in main text we considered is Lipschitz continuous.

□

D.1. About the condition $\|\mathbf{u}''\|_\infty \leq M$ on $[0, T]$.

The condition that "the second derivative of the true solution is uniformly bounded" in Theorem 5.1 is usually easy to satisfy and our algorithm can collaborate with it. Specifically,

(1) The "second derivative" $||\mathbf{u}''||_\infty$ does not directly affect the training objective of AttNS, i.e., the AttNS can use a large step size $k\Delta t$ (fast estimation speed) to solve the differential equations such that we can achieve a similar accuracy to the original numerical method solved by a small step size Δt (slow estimation speed).

The role of Theorem 5.1 is to inform us that AttNS can achieve this desired goal. Specifically, given the evaluation time T and the Lipschitz constant L , if the neural network is well-trained, i.e., $\beta\sqrt{\delta} \to 0$, then the accuracy of AttNS, $|\hat{\mathbf{u}}_N - \mathbf{u}(T)|$, and the accuracy of the original numerical algorithm at small time steps Δt , $|\mathbf{u}_{Nk} - \mathbf{u}(T)|$, are comparable. This is because

$$|\hat{\mathbf{u}}_N - \mathbf{u}(T)| \leq \underbrace{|\mathbf{u}_{Nk} - \mathbf{u}(T)|}_{\leq \Delta t \cdot M \cdot \exp(2TL)/2L} + \underbrace{|\hat{\mathbf{u}}_N - \mathbf{u}_{Nk}|}_{\leq \beta\sqrt{\delta} \to 0}. \tag{42}$$

On the other hand, the above inequality reveals that the upper bound M of $|\mathbf{u}''|_\infty$ mainly affects the accuracy of both AttNS and the original numerical method. When M is large, the upper bound of their accuracy may increase, but it does not affect the goal that AttNS with large step size $k\Delta t$ can approximate to the original numerical method at small time steps Δt .

(2) Most "second derivative" $|\mathbf{u}''|_\infty$ are relatively small. We extracted 1000 trajectories from the testing sets of two systems, namely the spring-chain and the 2-link pendulum, and computed their $|\mathbf{u}''|_\infty$ by second-order difference scheme. The trajectories were generated using the RK4 method with a step size of 1e-4, which can be considered as a proxy for the true solution. The results are presented in the table below.

Table 6: The distribution of $|\mathbf{u}''|_\infty$.

$\|\mathbf{u}''\|_\infty$ **(spring-chain)**	**30**	**60**	**90**	**120**	**150**	**180**	**210**	**240**	**270**
The percentage of ¡ $\|\mathbf{u}''\|_\infty$	55.35%	80.51%	91.84%	96.68%	98.57%	99.41%	99.76%	99.91%	99.97%
$\|\mathbf{u}''\|_\infty$ **(2-link pendulum)**	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**
The percentage of ¡ $\|\mathbf{u}''\|_\infty$	41.10%	65.93%	80.55%	89.48%	94.66%	97.61%	99.03%	99.71%	99.95%

From these tables, we observe that the distribution of $|\mathbf{u}''|_\infty$ follows a long-tailed distribution, where the majority of $|\mathbf{u}''|_\infty$ are relatively small. Note that the magnitude of $|\mathbf{u}''|_\infty$ varies across different systems due to differences in dimensionality and coordinates. And only a minority of $|\mathbf{u}''|_\infty$ are large. This observation is consistent with the findings in (Liang et al., 2022) for other chaotic systems, i.e., three-body systems and billiard systems.

The large $|\mathbf{u}''|_\infty$ corresponds to stiffness steps in the dynamical system, where the solution changes abruptly and may have some impact on the model training. We next investigate whether our proposed Attsovler can collaborate with these large $|\mathbf{u}''|_\infty$.

(3) Our proposed AttNS can collaborate with large "second derivative". As the examples using the elastic pendulum system and 2-link pendulum, we selected the trajectories in the test set with the top 10% the largest "second derivative" and recombined them into a new test set. The experimental results with RK4 (step size = 1e-1) at evaluation time T=10 are presented below.

Method	**MSE loss (elastic)**	**MSE loss (2-link)**
RK4	4.33e-0	3.56e-2
NeurVec	6.32e-5	3.54e-6
AttNS (ours)	**1.34e-6**	**8.54e-8**

We can find that our method can still significantly outperform SOTA and traditional numerical methods in these trajectories with relatively large "second derivatives", although the performance does decrease.

E. The proof of Theorem 5.3.

Theorem 5.3. Let Net$(\hat{S}|\phi, \mathbb{D}_f)$ and Net$(\mathbf{u}_n|\phi, \mathbb{D}_f)$, where $\hat{S} = S(\mathbf{f}, \mathbf{u}_n, \Delta t_c)$, be the correction term of general AHS and AttNS, respectively. For $\epsilon > 0$, when the data size is more than N', the empirical error of two methods satisfy $R_e(\phi|\text{AHS}) \leq \epsilon$ and $R_e(\phi|\text{AttNS}) \leq \epsilon$. For small enough $\epsilon_0 \ll \epsilon$ and Euler method, we have

$$N(\text{AttNS}) \lesssim N(\text{AHS}), \tag{43}$$

where $N(*)$ is the lower bound of the data size that the generalization error of method $*$ can reach $\epsilon(1-\epsilon_0)^{-1}$.

Lemma E.1. *Consider the set of the models $S_k = \{f(\cdot|w), w \in \omega_k\}$. If $S_1 \subset S_2 \subset \cdots \subset S_k \subset \cdots$,*

$$h_1 \leq h_2 \leq \cdots \leq h_k \leq \cdots, \tag{44}$$

where h_i is the Vapnik-Chervonenkis dimension of the model in set $S_i, i = 1, 2, \cdots$.

Lemma E.2. *Let $f(x) = (\ln x + \alpha)/x$, $x \in (0, +\infty)$ and $\alpha \in \mathbb{R}$. $f(x)$ achieves its maximum value at $x = e^{1+\alpha}$, when $x \in (0, e^{1+\alpha})$, $f(x)$ rises monotonically, and when $x \in (e^{1+\alpha}, +\infty)$, $f(x)$ decreases monotonically.*

Lemma E.3. *Let $f(x) = x\exp(\frac{-a}{x})$ where $a > 0$ and $x \in \mathbb{R}$. Then $f(x)$ rises monotonically.*

Lemma E.4. *Let $W(x)$ be Lambert W function, if $|x| < 1/e$, the Taylor expansion of $W(x)$ is*

$$W(x) = \sum_{n=1}^{\infty} \frac{(-n)^{n-1}}{n!} x^n = x - x^2 + \frac{3}{2}x^3 - \frac{8}{3}x^4 + \cdots. \tag{45}$$

Assumption E.5. For $x \in \mathbb{R}$ and a general real function $f(x)$, we consider the maps $f_1 : x \to \nabla_x f(x)^T f(x)$ and $f_2 : f(x) \to \nabla_x f(x)^T f(x)$. For a small enough $\epsilon > 0$, if S_1 and S_2 are the sets that $\forall g_1(\phi) \in S_1$ and $g_2(\phi) \in S_2$,

$$\|g_1(\phi) - f_1\| \leq \epsilon, \quad \|g_2(\phi) - f_2\| \leq \epsilon, \tag{46}$$

we assume that $S_1 \subset S_2$. This assumption means that the complexity of the model for fitting f_1 is higher than that for f_2.

Proof. (For Theorem 6.1).

According to Vapnik-Chervonenkis theory (Vapnik, 1999) for regression, for any data distribution $P(x, y)$, model $\mathscr{A}(\phi)$, the generalization error $R(\phi)$ and empirical error $R_e(\phi)$, the inequality

$$R(\phi) \leq R_e(\phi) \left(1 - c\sqrt{\delta(h_{\mathscr{A}(\phi)}, N)}\right)_+^{-1}, \tag{47}$$

holds with probability $1 - \eta$. $h_{\mathscr{A}(\phi)}$ is Vapnik-Chervonenkis dimension of model $\mathscr{A}(\phi)$, N is the data size, c usually is set as 1, and

$$\delta(h, N) = (h\left[\ln\frac{N}{h} + 1\right] - \ln\eta)/N. \tag{48}$$

Therefore, from Eq.(47), for the generalization error $R(\phi|\text{Neur.})$ and $R(\phi|\text{Att.})$, we have

$$R(\phi|\text{Neur.}) \leq R_e(\phi|\text{Neur.}) \left(1 - c\sqrt{\delta(h_{\text{Neur.}(\phi)}, N)}\right)_+^{-1}, \tag{49}$$

and

$$R(\phi|\text{Att.}) \leq R_e(\phi|\text{Att.}) \left(1 - c\sqrt{\delta(h_{\text{Att.}(\phi)}, N)}\right)_+^{-1}, \tag{50}$$

Next, we consider how many data size can the term $\left(1-\sqrt{\delta(h_{\mathscr{A}(\phi)},N)}\right)^{-1}$ reach the accuracy $(1-\epsilon_0)^{-1}$. Let $\delta := (\ln\bar{N}+\alpha)/\bar{N}$, where $\bar{N}=N/h_{\mathscr{A}(\phi)}$ and $\alpha = 1-\ln\eta/h_{\mathscr{A}(\phi)}$.

$$\begin{aligned}(\ln\bar{N}+\alpha)/\bar{N}=\epsilon_0 &\iff \ln\bar{N}+\alpha=\bar{N}\epsilon_0\\ &\iff \bar{N}\exp(-\bar{N}\epsilon_0)=\exp(-\alpha)\\ &\iff -\epsilon_0\bar{N}\exp(-\bar{N}\epsilon_0)=-\epsilon_0\exp(-\alpha).\end{aligned} \tag{51}$$

Therefore $-\epsilon\bar{N}=W(-\epsilon_0\exp(-\alpha))$, where W is Lambert W function (Lehtonen, 2016), and

$$\bar{N}=-\frac{h_{\mathscr{A}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\mathscr{A}(\phi)}}-1)\right), \tag{52}$$

From Lemma E.2, we know that when the data size $N\geq\bar{N}$, $\left(1-\sqrt{\delta(h_{\mathscr{A}(\phi)},N)}\right)^{-1}$ can reach the accuracy $(1-\epsilon_0)^{-1}$. From Eq.(49) and Eq.(50), when $N\geq\max\left\{N',-\frac{h_{\text{Neur.}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\text{Neur.}(\phi)}}-1)\right)\right\}$ and $N\geq\max\left\{N',-\frac{h_{\text{Att.}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\text{Att.}(\phi)}}-1)\right)\right\}$, we have

$$R(\phi|\text{Att.})\leq\epsilon(1-\epsilon_0)^{-1},\quad R(\phi|\text{Neur.})\leq\epsilon(1-\epsilon_0)^{-1}, \tag{53}$$

holds with probability $1-\eta$. In fact, since $\epsilon_0\ll\epsilon$ and Lemma E.2, in Eq.(52), $\bar{N}\gg N'$, therefore, we can set

$$N(\text{AttNS})=-\frac{h_{\text{Att.}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\text{Att.}(\phi)}}-1)\right),\quad N(\text{AHS})=-\frac{h_{\text{Neur.}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\text{Neur.}(\phi)}}-1)\right). \tag{54}$$

Note that $\exp(\ln\eta/h_{\mathscr{A}(\phi)}-1)<\exp(-1)$, i.e.,

$$\left|-\epsilon_0\exp(\ln\eta/h_{\mathscr{A}(\phi)}-1)\right|<\exp(-1). \tag{55}$$

Therefore, since Lemma E.4, for Eq.(52), we have

$$\begin{aligned}-\frac{h_{\mathscr{A}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\mathscr{A}(\phi)}}-1)\right)&\approx-\frac{h_{\mathscr{A}(\phi)}}{\epsilon_0}\cdot\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\mathscr{A}(\phi)}}-1)\right)\\ &=h_{\mathscr{A}(\phi)}\exp(\frac{\ln\eta}{h_{\mathscr{A}(\phi)}}-1).\end{aligned} \tag{56}$$

Note that, for ODE $\mathrm{d}\mathbf{u}/\mathrm{d}t=\mathbf{f}(\mathbf{u}),\mathbf{u}(0)=\mathbf{c}_0$, the target label of the neural network in AI-enhanced numerical solver is actually the error term for different numerical solver, which is dominated by $\mathcal{O}\left[\nabla\mathbf{f}(\mathbf{u})\mathbf{f}(\mathbf{u})\right]$. From Assumption E.5 and Lemma E.1, we have

$$h_{\text{Att.}(\phi)}\leq h_{\text{Neur.}(\phi)}. \tag{57}$$

Therefore, we have

$$\begin{aligned}N(\text{AttNS})&=-\frac{h_{\text{Att.}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\text{Att.}(\phi)}}-1)\right) && \text{Since Eq.(54)}\\ &\lesssim-\frac{h_{\text{Neur.}(\phi)}}{\epsilon_0}\cdot W\left(-\epsilon_0\exp(\frac{\ln\eta}{h_{\text{Neur.}(\phi)}}-1)\right) && \text{Since Eq.(57) and Lemma E.3}\\ &=N(\text{AHS})\end{aligned}$$

□