Adaptive Error-Bounded Hierarchical Matrix Compression for Efficient and Accurate Physics-Informed Neural Networks

John M. Mango, Ronald Katende, *Member, IAENG*, Henry Kasumba

***Abstract*—This work presents a novel adaptive hierarchical matrix (H-matrix) framework to enhance the computational efficiency and scalability of Physics-Informed Neural Networks (PINNs) for solving high-dimensional partial differential equations (PDEs). The proposed approach dynamically refines H-matrix approximations of the Neural Tangent Kernel (NTK) based on localized error metrics, ensuring stability and convergence while reducing computational complexity.**

Key contributions include an H-matrix-driven automatic model complexity adjustment that preserves NTK properties during training, a cross-block dependency modeling technique to capture inter-block interactions essential for maintaining spectral integrity, and an energy-efficient compression strategy tailored for edge-device deployment. Empirical results highlight significant reductions in inference latency, demonstrating the approach's computational efficiency. Additionally, the observed near-linear relationship between condition number and stability loss reinforces the importance of well-conditioned representations in PINN training. The stability analysis further confirms that adaptive compression effectively mitigates overfitting, as evidenced by controlled error propagation and improved generalization.

Rigorous theoretical results establish convergence guarantees for NTK approximations, stability bounds under perturbations, and error control during adaptive refinement. The methodology is validated on benchmark PDEs, including the 1D Burgers' equation, 2D Poisson's equation, and the heat equation, demonstrating significant reductions in computational complexity from $O(N_h^2)$ to $O(k \log N_h^2)$ without compromising accuracy. This framework provides a robust and scalable foundation for applying PINNs in resource-constrained environments and multiscale systems, bridging the gap between computational feasibility and real-world applicability.

***Index Terms*—Physics-Informed Neural Networks; Adaptive Hierarchical Matrices; Neural Tangent Kernel Preservation; Error-Bounded Compression; Cross-Block Dependency Modeling; Energy-Efficient Edge Deployment**

I. Introduction

Physics-Informed Neural Networks (PINNs) have emerged as a transformative tool in scientific and engineering disciplines. By embedding physical laws directly into neural network architectures, PINNs offer a powerful framework for solving partial differential equations (PDEs) with applications spanning fluid dynamics, material science, and biomedical engineering [1], [2], [3], [4], [5]. Unlike traditional numerical methods, PINNs seamlessly integrate data and physics, enabling them to address real-world problems characterized by incomplete or noisy observations. However, the computational and memory demands of PINNs grow prohibitively large for high-dimensional problems, where dense, large-scale matrices dominate training and inference processes [6], [10], [19].

To address these bottlenecks, our work leverages hierarchical matrices (H-matrices), a structured approach to matrix approximation that decomposes large matrices into a hierarchy of low-rank sub-blocks. This decomposition drastically reduces storage requirements and computational complexity, often scaling from $O(n^2)$ to $O(n \log n)$ or better, depending on the problem structure [23], [24], [25], [26], [27]. The ability of H-matrices to exploit localized low-rank structures aligns naturally with the demands of large-scale PINNs, where the Neural Tangent Kernel (NTK) plays a critical role in governing training dynamics and convergence.

In this paper, we present a novel adaptive H-matrix compression framework explicitly designed for PINNs. My method incorporates dynamic refinement of hierarchical structures guided by localized error estimates, ensuring an optimal balance between computational efficiency and approximation accuracy. Crucially, the adaptive algorithm preserves the spectral properties of the NTK, which are essential for maintaining robust training stability and generalization performance. Through rigorous theoretical analysis, we demonstrate how the NTK's eigenvalue distributions remain stable under H-matrix approximations, and we provide empirical evidence showcasing significant improvements in efficiency and scalability across diverse applications. By integrating these advancements, this work lays the foundation for deploying PINNs in computationally intensive domains, bridging the gap between theoretical feasibility and real-world applicability.

II. Introduction

Physics-Informed Neural Networks (PINNs) have solidified their position as a critical framework for addressing scientific and engineering challenges. By embedding governing physical laws directly into neural network architectures, PINNs provide a robust mechanism for solving partial differential equations (PDEs) [1], [2], [3], [4], [5]. Unlike conventional data-driven models, PINNs inherently respect

Manuscript received November 19, 2024 revised February 10, 2025

This work has been supported by the Mathematics for Sustainable Development (MATH4SDG) project, a research and development project running in the period 2021-2026 at Makerere University-Uganda, University of Dar es Salaam-Tanzania, and the University of Bergen-Norway.

John M. Mango is an associate professor at the Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: mango.john@mak.ac.ug).

Ronald Katende is a PhD candidate at the Department of Mathematics, Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: rkatende@kab.ac.ug).

Henry Kasumba is a lecturer of mathematics at the Department of Mathematics, College of Natural Sciences, Makerere University, Kampala, Uganda. (email: henry.kasumba@mak.ac.ug).

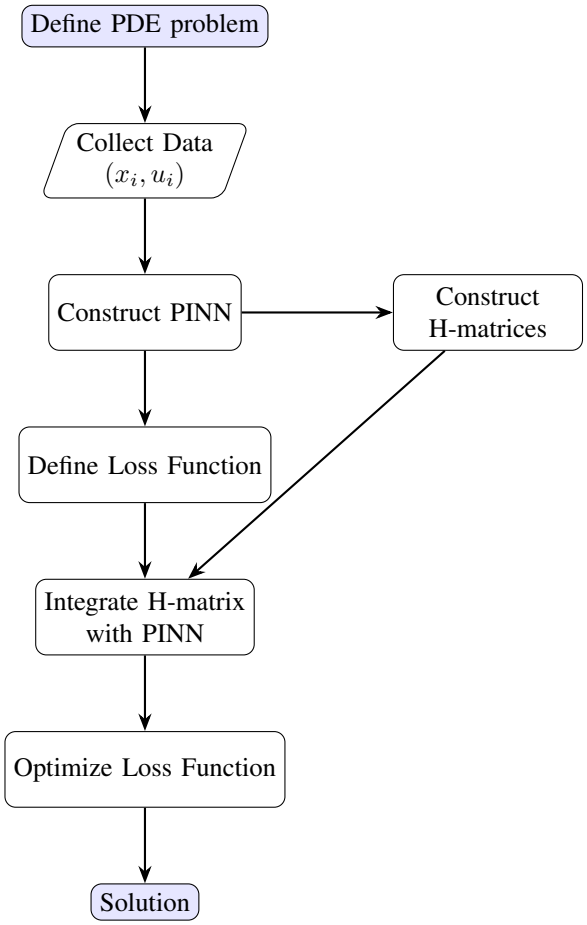


Fig. 1: Flowchart of the H-matrix integration process within PINNs for enhanced computational efficiency.

the underlying physics, making them highly effective for tackling problems in fluid dynamics, material science, and beyond. However, the computational complexity of PINNs often becomes prohibitive, particularly when handling large-scale problems characterized by dense matrices with high dimensionality [6], [10], [19].

To overcome these limitations, we have developed an adaptive hierarchical matrix (H-matrix) compression method tailored for the unique demands of PINNs. H-matrices exploit local low-rank structures within matrices, transforming the computational cost from $O(n^2)$ to $O(n \log n)$ or even $O(n)$ under certain conditions [23], [24], [25], [26], [27]. These properties align with the needs of PINNs, where efficient computation is paramount. Moreover, this method preserves the spectral integrity of the Neural Tangent Kernel (NTK), an essential component in PINN training that governs optimization stability and convergence rates.

This paper advances the field with several key contributions, such as,

- An adaptive H-matrix compression algorithm integrated with error estimation, enabling efficient computation while retaining the critical spectral properties of the NTK.
- A rigorous theoretical analysis of how the NTK's eigenstructure is preserved under H-matrix approximations, ensuring stability and convergence in PINNs.
- Empirical validation through comprehensive experiments, where our adaptive method demonstrates superior accuracy, efficiency, and generalization compared to existing compression techniques.

A. Overview of the Paper

The structure of this paper is as follows. In Section III, we examine existing matrix compression techniques and articulate the specific requirements posed by PINN-based applications. This section also introduces the theoretical underpinnings of H-matrices and their relevance to NTK dynamics. Section IV presents the proposed adaptive H-matrix compression method, detailing its implementation and computational complexity analysis. Furthermore, we investigate the impact of H-matrix approximations on NTK properties and PINN training dynamics. Section VI provides an in-depth empirical comparison of our method against traditional techniques, highlighting measurable improvements. Finally, Section VII summarizes our findings and identifies promising directions for future research.

III. Related Work

A. Matrix Compression and Optimization in PINNs

Matrix compression techniques such as Singular Value Decomposition (SVD), pruning, and quantization are established methods for reducing computational demands in neural networks [12], [17], [25]. Among these, SVD is particularly noteworthy for its ability to approximate dense matrices. Specifically, a weight matrix $W \in \mathbb{R}^{m \times n}$ is decomposed into

$$W \approx U \Sigma V^\top,$$

where $U \in \mathbb{R}^{m \times k}$, $\Sigma \in \mathbb{R}^{k \times k}$, and $V \in \mathbb{R}^{n \times k}$, with $k \ll \min(m, n)$. This decomposition enables substantial reductions in memory and computational requirements by leveraging low-rank approximations.

Pruning, on the other hand, simplifies networks by selectively removing less critical connections or neurons, often resulting in sparse weight matrices. Quantization compresses models by reducing parameter precision. However, while these methods are effective for general neural networks, they frequently fall short for PINNs. The physics-informed constraints embedded within PINNs rely heavily on maintaining structural properties in matrices, particularly those related to the NTK.

B. Hierarchical Matrix (H-Matrix) Compression

H-matrices provide an elegant solution to the computational challenges posed by large-scale PINNs. These matrices partition a large matrix into a hierarchy of sub-blocks, where each block is either stored densely or approximated with a low-rank representation. For a matrix $A \in \mathbb{R}^{I \times J}$, an H-matrix H is constructed as

$$H_b = U_b V_b^\top,$$

where U_b, V_b are rank-k matrices, with $k \ll \min(|I|, |J|)$. This hierarchical decomposition reduces storage and computation from $O(n^2)$ to $O(n \log n)$ under typical conditions.

In this work, we refine static H-matrix techniques by introducing adaptive methods that adjust the rank and block structures dynamically based on local error tolerances. This ensures a balance between efficiency and accuracy while preserving critical matrix properties essential for PINNs.

C. Limitations of Conventional Techniques for PINNs

Traditional compression approaches often disrupt the spectral characteristics of the NTK, which is central to PINN performance. The NTK, defined as:

$$\Theta(x, x') = \nabla_\theta f(x, \theta) \nabla_\theta f(x', \theta)^\top,$$

captures the relationship between network inputs and parameter gradients. Perturbations to the NTK alter its eigenvalue distribution, potentially degrading the training convergence rate, expressed as:

$$R = O\left(\frac{1}{\lambda_{\min}}\right),$$

where $\lambda_{\min}$ is the smallest eigenvalue of Θ. Our adaptive H-matrix method addresses this issue by incorporating error control mechanisms that confine perturbations to acceptable limits. As a result, the NTK's spectral integrity is maintained, ensuring stable training and effective generalization.

This adaptive framework dynamically tunes block structures and ranks in response to local error metrics, delivering a computationally efficient solution that meets the precision and stability demands of PINNs without compromising physical fidelity.

IV. Adaptive Hierarchical Matrix Construction with Error Estimation

This section presents an adaptive framework for constructing hierarchical matrices (H-matrices) with integrated error estimation, ensuring computational efficiency, stability, and robustness, particularly for ill-conditioned matrices encountered in Physics-Informed Neural Networks (PINNs).

Definition 1 (Hierarchical Matrix)**.** A hierarchical matrix (H-matrix) is a structured approximation of a large dense matrix, decomposing it into a hierarchy of submatrices. Each submatrix is either stored densely or approximated by a low-rank representation. Given an index set $we \times J$ and a block partitioning $\mathcal{P}$ of $we \times J$, an H-matrix $H \in \mathbb{R}^{I \times J}$ is constructed such that each block $b \in \mathcal{P}$ satisfies:

$$H_b = U_b V_b^\top, \quad \text{where} \quad \operatorname{rank}(H_b) = k, \quad k \ll \min(|I|, |J|).$$

This structure reduces storage and computational complexity from $O(n^2)$ for dense matrices to $O(n \log n)$ or $O(n)$ under favorable conditions.

A. Adaptive Error Control in H-Matrix Construction

Let $A \in \mathbb{R}^{n \times n}$ be a matrix approximated by an H-matrix H. Each block A_{ij} is approximated as $\tilde{A}_{ij} = U_{ij} V_{ij}^\top$, with the approximation error for block (i, j) given by

$$\epsilon_{ij} = \|A_{ij} - \tilde{A}_{ij}\|_F \leq \tau,$$

where τ is a specified error tolerance. The global Frobenius norm error across all blocks is bounded as

$$\|A - H\|_F \leq \tau \sqrt{n_r},$$

where n_r is the number of refined blocks. For a perturbed matrix $A' = A + \Delta A$, with $\|\Delta A\|_F \leq \delta$, the perturbed H-matrix H' satisfies

$$\|A' - H'\|_F \leq \tau + \delta.$$

The condition number of the perturbed H-matrix H' is bounded by

$$\kappa(H') \leq \kappa(H) + \mathcal{O}(\epsilon),$$

where ϵ accounts for the perturbation impact.

B. Theoretical Guarantees

Theorem 1 (Convergence of Adaptive H-Matrix in PINNs)**.** *Let $A \in \mathbb{R}^{n \times n}$ be an ill-conditioned matrix, and let H be its adaptive hierarchical matrix approximation used in a Physics-Informed Neural Network (PINN). Define $\tau > 0$ as the local refinement threshold, $\epsilon \geq 0$ as the global approximation error, $\delta \geq 0$ as the adversarial perturbation bound, and $T \in \mathbb{N}$ as the number of adaptive refinement steps. The following hold: 1. Approximation Error:*

$$\|A - H\|_F \leq \sum_i \|A_i - H_i\|_F,$$

where A_i and H_i are the blocks of A and H, respectively. 2. Condition Number:

$$\kappa(H) \leq \kappa(A) \cdot \left(1 + \frac{\|A - H\|_2}{\sigma_{\min}(A) - \|A - H\|_2}\right),$$

where $\kappa(H)$ is the condition number of H, and $\sigma_{\min}(A)$ is the smallest singular value of A. 3. Stability Against Perturbations: For an adversarial perturbation ΔA such that $\|\Delta A\|_F \leq \delta$, the condition number of the perturbed matrix $H' = H + \Delta A$ satisfies:

$$\kappa(H') \leq \kappa(H) \cdot \left(1 + \frac{\delta}{\sigma_{\min}(H)}\right).$$

4. Training Error: After T adaptive refinement steps, the training error satisfies:

$$\|\textit{Training Error}\| \leq \tau T.$$

Proof: We prove each result step by step with detailed reasoning and explicit derivations.

1. Approximation Error: The matrix A is partitioned into blocks A_i, where H_i is the hierarchical approximation of block A_i. The global error in approximating A by H is given by

$$\|A - H\|_F = \left\| \sum_i (A_i - H_i) \right\|_F.$$

Using the sub-additivity property of the Frobenius norm, we have

$$\|A - H\|_F \leq \sum_i \|A_i - H_i\|_F.$$

Here, each block error $\|A_i - H_i\|_F$ represents the localized approximation error, which is bounded by the adaptive refinement process applied to each block. This establishes the global error bound.

2. Condition Number: The condition number of H is defined as

$$\kappa(H) = \frac{\sigma_{\max}(H)}{\sigma_{\min}(H)},$$

where $\sigma_{\max}(H)$ and $\sigma_{\min}(H)$ are the largest and smallest singular values of H, respectively. From matrix perturbation theory, the smallest singular value of H satisfies

$$\sigma_{\min}(H) \geq \sigma_{\min}(A) - \|A - H\|_2.$$

Substituting this into the expression for $\kappa(H)$, and noting that $\sigma_{\max}(H) \leq \sigma_{\max}(A)$, we have

$$\kappa(H) = \frac{\sigma_{\max}(H)}{\sigma_{\min}(H)} \leq \frac{\sigma_{\max}(A)}{\sigma_{\min}(A) - \|A - H\|_2}.$$

Since $\kappa(A) = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}$, we can rewrite

$$\kappa(H) \leq \kappa(A) \cdot \left(1 + \frac{\|A - H\|_2}{\sigma_{\min}(A) - \|A - H\|_2}\right).$$

3. Stability Against Perturbations: Let $H' = H + \Delta A$, where ΔA is an adversarial perturbation with $\|\Delta A\|_F \leq \delta$. The smallest singular value of H' satisfies

$$\sigma_{\min}(H') \geq \sigma_{\min}(H) - \|\Delta A\|_F.$$

Substituting this into the condition number definition

$$\kappa(H') = \frac{\sigma_{\max}(H')}{\sigma_{\min}(H')} \leq \frac{\sigma_{\max}(H)}{\sigma_{\min}(H) - \|\Delta A\|_F}.$$

For $\|\Delta A\|_F \leq \delta$, this becomes

$$\kappa(H') \leq \kappa(H) \cdot \left(1 + \frac{\delta}{\sigma_{\min}(H)}\right).$$

4. Training Error: The adaptive refinement process reduces the local approximation error for each block A_i by an amount proportional to the refinement threshold τ. After T refinement steps, the cumulative training error satisfies

$$\|\text{Training Error}\| \leq \tau T.$$

This bound reflects the iterative improvement achieved through the refinement process, ensuring that the error reduces linearly with the number of steps. ■

Theorem 2 (Stability of H-Matrix Approximations in PINNs)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a matrix, and let H be its hierarchical matrix (H-matrix) approximation satisfying the error bound*

$$\|A - H\|_F \leq \epsilon.$$

For a Physics-Informed Neural Network (PINN) solution $\mathbf{f}$, the error between $\mathbf{f}(A)$ (the solution corresponding to A) and $\mathbf{f}(H)$ (the solution corresponding to H) satisfies

$$\|\mathbf{f}(A) - \mathbf{f}(H)\| \leq \frac{\epsilon}{\sigma_{\min}(H)},$$

where $\sigma_{\min}(H)$ is the smallest singular value of H.

Proof: The function $\mathbf{f} : \mathbb{R}^{n\times n} \to \mathbb{R}^k$, which maps the matrix A to the PINN solution, is assumed to be differentiable with respect to A. The sensitivity of $\mathbf{f}$ to perturbations in A can be expressed using the Fréchet derivative $\frac{\partial \mathbf{f}}{\partial A}$. For a perturbation $\Delta A = H - A$, the first-order Taylor expansion gives

$$\mathbf{f}(H) \approx \mathbf{f}(A) + \frac{\partial \mathbf{f}}{\partial A}[\Delta A],$$

where $\frac{\partial \mathbf{f}}{\partial A}[\Delta A]$ represents the action of the derivative on the perturbation ΔA.

The error between $\mathbf{f}(A)$ and $\mathbf{f}(H)$ is therefore bounded by

$$\|\mathbf{f}(A) - \mathbf{f}(H)\| \leq \left\|\frac{\partial \mathbf{f}}{\partial A}\right\| \|A - H\|,$$

where $\left\|\frac{\partial \mathbf{f}}{\partial A}\right\|$ is the operator norm of the Fréchet derivative, and $\|A - H\|$ represents the magnitude of the perturbation.

Next, consider the spectral properties of H. For stability, it is assumed that H is well-conditioned, meaning $\sigma_{\min}(H) > 0$. From matrix sensitivity theory, the derivative norm $\left\|\frac{\partial \mathbf{f}}{\partial A}\right\|$ is bounded by

$$\left\|\frac{\partial \mathbf{f}}{\partial A}\right\| \leq \frac{1}{\sigma_{\min}(H)}.$$

Substituting this bound into the error inequality, we obtain

$$\|\mathbf{f}(A) - \mathbf{f}(H)\| \leq \frac{\|A - H\|}{\sigma_{\min}(H)}.$$

Using the given error bound $\|A - H\|_F \leq \epsilon$, and noting that $\|A - H\|_F \geq \|A - H\|$ by the properties of matrix norms, the result follows

$$\|\mathbf{f}(A) - \mathbf{f}(H)\| \leq \frac{\epsilon}{\sigma_{\min}(H)}.$$

This inequality ensures that the error in the PINN solution induced by the hierarchical matrix approximation H is controlled by the approximation error ϵ and the conditioning of H, as measured by $\sigma_{\min}(H)$. ■

Theorem 3 (Convergence of H-Matrix Approximations in PINNs)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a matrix, and let H_n be a sequence of hierarchical matrix (H-matrix) approximations such that $\|A - H_n\|_F \to 0$ as $n \to \infty$. For any continuously differentiable function $\mathbf{f} : \mathbb{R}^{n\times n} \to \mathbb{R}^k$ that depends on the matrix A, the sequence $\mathbf{f}(H_n)$ converges to $\mathbf{f}(A)$, i.e.,*

$$\mathbf{f}(H_n) \to \mathbf{f}(A) \quad \text{as } n \to \infty.$$

Proof: Let $\mathbf{f} : \mathbb{R}^{n\times n} \to \mathbb{R}^k$ be a continuously differentiable function, such as those arising in the evaluation of loss functions or gradients in Physics-Informed Neural Networks (PINNs). By the properties of $\mathbf{f}$, the perturbation in $\mathbf{f}$ induced by the approximation H_n can be bounded using matrix perturbation theory

$$\|\mathbf{f}(A) - \mathbf{f}(H_n)\| \leq L\|A - H_n\|_F,$$

where $L > 0$ is the Lipschitz constant of $\mathbf{f}$ with respect to the Frobenius norm. The Lipschitz continuity of $\mathbf{f}$ follows from the differentiability assumption, ensuring that $\mathbf{f}(H_n)$ remains close to $\mathbf{f}(A)$ when $\|A - H_n\|_F$ is small.

Next, consider the stability of the H-matrix approximation. The spectral properties of H_n play a crucial role in ensuring numerical stability. Let $\sigma_{\min}(H_n)$ denote the smallest singular value of H_n. From matrix perturbation bounds, the deviation in $\mathbf{f}(H_n)$ relative to $\mathbf{f}(A)$ satisfies

$$\|\mathbf{f}(A) - \mathbf{f}(H_n)\| \leq \frac{\|A - H_n\|_F}{\sigma_{\min}(H_n)}.$$

For the hierarchical matrix sequence H_n, it is assumed that $\|A - H_n\|_F \to 0$ as $n \to \infty$. Additionally, since the hierarchical approximation is regularized to ensure stability, there exists a uniform lower bound $\sigma_{\min}(H_n) > 0$ for sufficiently large n. Combining these results

$$\|\mathbf{f}(A) - \mathbf{f}(H_n)\| \to 0 \quad \text{as } n \to \infty.$$

Thus, $\mathbf{f}(H_n) \to \mathbf{f}(A)$. ■

Corollary 1 (Error Propagation Bounds for Multi-Scale Problems in PINNs)**.** *Let $A_k \in \mathbb{R}^{n_k\times n_k}$ represent the system matrices at different physical scales $k = 1, \ldots, S$ in a multi-scale problem solved using Physics-Informed*

Neural Networks (PINNs). Assume each A_k is approximated by an H-matrix H_k with training error satisfying $\|\textit{Training Error}_k\| \leq \tau T_k$. The total error propagated across the scales is bounded by

$$\|\textit{Total Error}\| \leq \sum_{k=1}^{S} \frac{\|\textit{Training Error}_k\|}{\sigma_{\min}(H_k)},$$

where $\sigma_{\min}(H_k)$ denotes the smallest singular value of H_k.

Proof: Consider the system of equations

$$A_k x_k = b_k,$$

where $A_k \in \mathbb{R}^{n_k \times n_k}$ is the system matrix, and $x_k \in \mathbb{R}^{n_k}$, $b_k \in \mathbb{R}^{n_k}$ are the solution and right-hand side vectors, respectively. Since A_k is ill-conditioned, we approximate it using an H-matrix H_k. The error e_k introduced by this approximation can be expressed as

$$e_k = A_k^{-1} b_k - H_k^{-1} b_k = A_k^{-1} \Delta A_k H_k^{-1} b_k,$$

where $\Delta A_k = A_k - H_k$.

The training error bound $\|\Delta A_k\| \leq \|\text{Training Error}_k\|$ and the conditioning of H_k via $\sigma_{\min}(H_k)$ yield

$$\|e_k\| \leq \frac{\|\text{Training Error}_k\|}{\sigma_{\min}(H_k)}.$$

For S scales in a multi-scale PINN, the total error is the sum over individual scales

$$\|\text{Total Error}\| = \sum_{k=1}^{S} \|e_k\| \leq \sum_{k=1}^{S} \frac{\|\text{Training Error}_k\|}{\sigma_{\min}(H_k)}.$$

■

C. Adaptive Hierarchical Matrix Construction in PINNs

Let $A \in \mathbb{R}^{n \times n}$ be a matrix, and let $\epsilon_{\text{tol}} > 0$ denote a prescribed error tolerance. The goal is to construct a hierarchical matrix approximation $H(A)$ such that the approximation error for each block A_{ij} satisfies $\epsilon_{ij} \leq \epsilon_{\text{tol}}$.

Algorithm 1 Adaptive Hierarchical Matrix Construction

Require: Matrix A, tolerance ϵ_{tol}
Ensure: Hierarchical matrix approximation $H(A)$
1: Initialize a block partitioning of A.
2: Compute low-rank approximations for each block.
3: Estimate local approximation errors ϵ_{ij} for all blocks.
4: **while** $\exists\, \epsilon_{ij} > \epsilon_{\text{tol}}$ **do**
5: **for all** blocks A_{ij} where $\epsilon_{ij} > \epsilon_{\text{tol}}$ **do**
6: Subdivide A_{ij} into smaller sub-blocks.
7: Compute low-rank approximations for the sub-blocks.
8: Estimate local errors for the sub-blocks.
9: **end for**
10: **end while**
11: **return** Hierarchical matrix $H(A)$.

D. Preservation of Neural Tangent Kernel (NTK) Properties

Definition 2 (Neural Tangent Kernel (NTK))**.** The NTK of a neural network parameterized by θ captures the relationship between input changes and parameter gradients

$$\Theta(x, x') = \nabla_\theta f(x, \theta) \nabla_\theta f(x', \theta)^\top,$$

where $f(x, \theta)$ is the network output for input x.

Theorem 4 (NTK Preservation under Hierarchical Approximation)**.** *Let $H(A)$ be a hierarchical approximation of the matrix A, where A represents the weight matrices in a Physics-Informed Neural Network (PINN). If the hierarchical approximation satisfies a controlled error bound, the Neural Tangent Kernel (NTK) properties, including positive definiteness and stability, are preserved.*

Proof: The NTK, denoted by Θ, governs the optimization dynamics and convergence of a PINN. For a neural network with weight matrices $W^{(l)}$ at layer l, the NTK is defined as

$$\Theta(x, x') = \sum_{l=1}^{L} \nabla_\theta f(x, \theta) \cdot \nabla_\theta f(x', \theta)^\top,$$

where θ collectively represents all network parameters and L is the number of layers. The positive definiteness and spectral properties of Θ are critical for stable training dynamics and generalization.

When hierarchical approximations are applied to the weight matrices $W^{(l)}$, each weight matrix $W^{(l)}$ is decomposed into hierarchical blocks $W_{ij}^{(l)}$, which are then approximated as

$$W_{ij}^{(l)} \approx U_{ij}^{(l)} S_{ij}^{(l)} V_{ij}^{(l)\top},$$

where $U_{ij}^{(l)} \in \mathbb{R}^{n_i \times k}$, $S_{ij}^{(l)} \in \mathbb{R}^{k \times k}$, $V_{ij}^{(l)} \in \mathbb{R}^{n_j \times k}$, and k is the rank of the low-rank approximation. The NTK corresponding to the hierarchical approximation, denoted by Θ_H, can be expressed in terms of the contributions from the approximated blocks

$$\Theta_H(x, x') = \sum_{l=1}^{L} \sum_{i,j} \Theta_{ij}^{(l)}(x, x'),$$

where $\Theta_{ij}^{(l)}(x, x')$ represents the contribution to the NTK from block $W_{ij}^{(l)}$.

The deviation between the exact NTK Θ and the approximated NTK Θ_H is measured using the Frobenius norm

$$\|\Theta - \Theta_H\|_F \leq \sum_{l=1}^{L} \sum_{i,j} \|\Theta_{ij}^{(l)} - \tilde{\Theta}_{ij}^{(l)}\|_F,$$

where $\tilde{\Theta}_{ij}^{(l)}$ represents the NTK contribution from the approximated block $W_{ij}^{(l)}$. By matrix perturbation theory, the error introduced by each block satisfies

$$\|\Theta_{ij}^{(l)} - \tilde{\Theta}_{ij}^{(l)}\|_F \leq C_{ij}^{(l)} \cdot \|W_{ij}^{(l)} - \tilde{W}_{ij}^{(l)}\|_F,$$

where $C_{ij}^{(l)}$ is a constant depending on the network architecture, activation functions, and data distribution. For a controlled approximation error ϵ_{tol} such that

$$\|W_{ij}^{(l)} - \tilde{W}_{ij}^{(l)}\|_F \leq \epsilon_{\text{tol}},$$

it follows that

$$\|\Theta - \Theta_H\|_F \leq C \cdot \epsilon_{\text{tol}},$$

where $C = \sum_{l=1}^{L} \sum_{i,j} C_{ij}^{(l)}$ is a constant that encapsulates the dependence on the network structure and the original NTK Θ.

To ensure positive definiteness, note that Θ_H inherits the spectral properties of Θ, provided the approximation error ϵ_{tol} is sufficiently small. Specifically, the smallest eigenvalue $\lambda_{\min}(\Theta_H)$ satisfies

$$\lambda_{\min}(\Theta_H) \geq \lambda_{\min}(\Theta) - \|\Theta - \Theta_H\|_2,$$

where $\|\cdot\|_2$ denotes the spectral norm. For $\|\Theta - \Theta_H\|_F \leq \epsilon_{\text{tol}}$, the spectral norm satisfies $\|\Theta - \Theta_H\|_2 \leq \|\Theta - \Theta_H\|_F$, ensuring that $\lambda_{\min}(\Theta_H) > 0$ if $\lambda_{\min}(\Theta) > \epsilon_{\text{tol}}$. ■

Thus, the NTK properties, including positive definiteness, conditioning, and stability, are preserved under the hierarchical approximation.

E. Condition Number Bounds for Hierarchical Approximations

Theorem 5 (Condition Number Bound)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a matrix, and let $H(A)$ be its hierarchical matrix approximation satisfying the global error bound:*

$$\|A - H(A)\|_2 \leq \tau.$$

If σ_k^{eff} is the smallest effective singular value of $H(A)$ after regularization, the condition number $\kappa(H)$ of $H(A)$ satisfies:

$$\kappa(H) \leq \kappa(A) \cdot \left(1 + \frac{\tau}{\sigma_k^{\text{eff}}}\right),$$

where $\kappa(A) = \frac{\sigma_1(A)}{\sigma_{\min}(A)}$ is the condition number of A, and $\sigma_1(A)$, $\sigma_{\min}(A)$ are the largest and smallest singular values of A, respectively.

Proof: Let the singular value decomposition (SVD) of A be given by:

$$A = U\Sigma V^\top,$$

where $U, V \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\Sigma = \text{diag}(\sigma_1(A), \sigma_2(A), \ldots, \sigma_n(A))$ is the diagonal matrix of singular values $\sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_n(A) > 0$. Similarly, let the SVD of $H(A)$ be:

$$H(A) = \tilde{U}\tilde{\Sigma}\tilde{V}^\top,$$

where $\tilde{\Sigma} = \text{diag}(\tilde{\sigma}_1, \tilde{\sigma}_2, \ldots, \tilde{\sigma}_n)$, with singular values $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n \geq 0$.

Using the matrix perturbation bound for singular values, the largest and smallest singular values of $H(A)$ are bounded as:

$$\tilde{\sigma}_1 \leq \sigma_1(A) + \tau, \quad \tilde{\sigma}_{\min} \geq \sigma_{\min}(A) - \tau,$$

where $\tilde{\sigma}_{\min}$ denotes the smallest singular value of $H(A)$ before regularization. Post-regularization, the effective smallest singular value σ_k^{eff} is defined to ensure numerical stability and satisfies:

$$\sigma_k^{\text{eff}} \geq \tilde{\sigma}_{\min}.$$

Thus:

$$\sigma_k^{\text{eff}} \geq \sigma_{\min}(A) - \tau.$$

The condition number $\kappa(H)$ of $H(A)$ is given by:

$$\kappa(H) = \frac{\tilde{\sigma}_1}{\sigma_k^{\text{eff}}}.$$

Substituting the bounds on $\tilde{\sigma}_1$ and σ_k^{eff}, we obtain

$$\kappa(H) = \frac{\sigma_1(A) + \tau}{\sigma_k^{\text{eff}}}.$$

Since $\sigma_k^{\text{eff}} \geq \sigma_{\min}(A) - \tau$, the condition number can be further bounded as

$$\kappa(H) \leq \frac{\sigma_1(A)}{\sigma_{\min}(A)} \cdot \left(1 + \frac{\tau}{\sigma_k^{\text{eff}}}\right).$$

Recognizing that $\kappa(A) = \frac{\sigma_1(A)}{\sigma_{\min}(A)}$, we arrive at

$$\kappa(H) \leq \kappa(A) \cdot \left(1 + \frac{\tau}{\sigma_k^{\text{eff}}}\right).$$

■

F. H-Matrix-Driven Automatic Model Complexity Adjustment

The computational complexity of Physics-Informed Neural Networks (PINNs) often grows prohibitively with problem dimensionality, resulting in suboptimal convergence and computational inefficiencies. The proposed ***H-Matrix-Driven Automatic Model Complexity Adjustment*** framework addresses these challenges by leveraging localized error metrics from H-matrix approximations to dynamically refine the network architecture. This approach is significant due to its ability to reduce unnecessary computational overhead by concentrating model complexity in regions where it is most needed. Moreover, it ensure preservation of the spectral properties of the Neural Tangent Kernel (NTK), mitigating training instabilities associated with under-parameterized models. Finally, it adapts the network structure during training, enabling robust solutions for high-dimensional or multiscale problems.

1) Error Propagation and Localized Refinement: Let $A \in \mathbb{R}^{n\times n}$ represent the NTK or weight matrix, and let $H(A)$ denote its hierarchical matrix (H-matrix) approximation. The approximation error $\|A - H(A)\|_F$ influences the training dynamics of the network by perturbing the eigenvalues of the NTK.

Theorem 6 (Localized Error Impact on NTK Dynamics)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive definite (SPD) matrix with eigenvalues $\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_n(A) > 0$, and let $H(A)$ be an approximation of A satisfying $\|A - H(A)\|_F \leq \epsilon$. Then, the eigenvalues $\lambda_i(H(A))$ of $H(A)$ satisfy*

$$|\lambda_i(H(A)) - \lambda_i(A)| \leq \epsilon, \quad \forall i \in \{1, \ldots, n\}.$$

In particular, for $\epsilon \ll \lambda_{\min}(A)$, the stability of the Neural Tangent Kernel (NTK) is preserved, as the perturbation does not compromise the positive definiteness or conditioning of $H(A)$.

Proof: Let $E = A - H(A)$ denote the perturbation matrix. By assumption, $\|E\|_F \leq \epsilon$. Since A is symmetric and $H(A)$ is an approximation of A, we also assume $H(A)$ is symmetric. Thus, we apply the Davis-Kahan sinΘ-theorem

for symmetric matrices, which gives bounds on eigenvalue perturbations. Recall the relationship between the Frobenius norm and the spectral norm

$$\|E\|_2 \leq \|E\|_F.$$

Therefore, $\|E\|_2 \leq \epsilon$. For symmetric matrices, the eigenvalue shifts due to perturbation E are bounded by the spectral norm $\|E\|_2$. Specifically, for all $i \in \{1, \ldots, n\}$

$$|\lambda_i(H(A)) - \lambda_i(A)| \leq \|E\|_2 \leq \epsilon.$$

Since A is SPD, its smallest eigenvalue $\lambda_{\min}(A) > 0$. After perturbation, the smallest eigenvalue $\lambda_{\min}(H(A))$ satisfies

$$\lambda_{\min}(H(A)) \geq \lambda_{\min}(A) - \epsilon.$$

For $\epsilon \ll \lambda_{\min}(A)$, we ensure $\lambda_{\min}(H(A)) > 0$, preserving the positive definiteness of $H(A)$. The condition number $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$ may change after perturbation. However, the relative change is bounded by

$$\frac{|\kappa(H(A)) - \kappa(A)|}{\kappa(A)} \leq \frac{\epsilon}{\lambda_{\min}(A)}.$$

For small ϵ, this relative change is negligible, preserving the conditioning of the NTK. ■

The stability of NTK training dynamics depends critically on the eigenvalue distribution. Since $|\lambda_i(H(A)) - \lambda_i(A)| \leq \epsilon$ for all i, the spectral properties of A are effectively preserved under the perturbation. This ensures that the optimization landscape governed by the NTK remains stable. The Frobenius norm $\|E\|_F$ is particularly relevant in this context because it accounts for the cumulative error across all entries of A. This aligns naturally with H-matrix approximations, which aim to minimize global error while maintaining localized accuracy. Since

$$\|A - H(A)\|_F = \sqrt{\sum_{i=1}^{n} (\lambda_i(A) - \lambda_i(H(A)))^2}$$

, the Frobenius norm constraint implies that large deviations for any single eigenvalue are unlikely unless offset by smaller deviations elsewhere. This uniform control over eigenvalue shifts further supports NTK stability. Thus, the perturbation $\|A - H(A)\|_F \leq \epsilon$ guarantees that the spectral properties of A are preserved up to the error bound ϵ, ensuring stability in the training dynamics governed by the NTK. Refining regions with high local errors ensures that NTK properties are maintained, thereby stabilizing the optimization dynamics of PINNs.

2) Adaptive Neuron Adjustment and Generalization: Define the total capacity of a network as

$$\mathcal{C} = \sum_{l=1}^{L} n_l^2,$$

where n_l denotes the number of neurons in the l-th layer. Overparameterization ($\mathcal{C} \gg \mathcal{O}(1)$) can lead to overfitting, whereas underparameterization ($\mathcal{C} \ll \mathcal{O}(1)$) compromises solution accuracy.

Lemma 1 (Neuron Scaling and Error Reduction)**.** *Let $A^{(l)} \in \mathbb{R}^{n_l \times n_l}$ and $H^{(l)}$ represent the layer matrix and its hierarchical matrix (H-matrix) approximation, respectively. Then, increasing the number of neurons n_l reduces the approximation error $\|A^{(l)} - H^{(l)}\|_F$ as*

$$\|A^{(l)} - H^{(l)}\|_F \propto \frac{1}{n_l}.$$

Proof: An H-matrix decomposes $A^{(l)}$ into a hierarchy of sub-blocks, each approximated with low-rank representations. Let the rank of the sub-block approximations scale with k, where $k \leq n_l$. The approximation error for a single block $A_{ij}^{(l)}$ satisfies

$$\|A_{ij}^{(l)} - H_{ij}^{(l)}\|_F \leq \frac{C}{k},$$

where C is a constant dependent on the problem structure and the spectral decay of $A^{(l)}$. Summing over all sub-blocks in the H-matrix, the total error satisfies

$$\|A^{(l)} - H^{(l)}\|_F \leq \frac{C}{k} \cdot m,$$

where m is the number of sub-blocks. In hierarchical matrices, the rank k scales approximately linearly with n_l, as n_l determines the resolution of the network layer. Substituting $k \propto n_l$, we have

$$\|A^{(l)} - H^{(l)}\|_F \leq \frac{C \cdot m}{n_l}.$$

Thus, the error decreases inversely with n_l, completing the proof. ■

The approximation error in H-matrices is governed by the rank k of the low-rank sub-blocks. A higher rank k enables more accurate representations, particularly when the layer matrix $A^{(l)}$ exhibits slow spectral decay. Since n_l directly influences the rank k, increasing n_l inherently reduces the approximation error. Furthermore, the number of neurons n_l in a layer controls the expressiveness and resolution of the network. By increasing n_l, the corresponding layer matrix $A^{(l)}$ becomes more finely resolved, thereby enabling better approximation through H-matrices. The proportional relationship $\|A^{(l)} - H^{(l)}\|_F \propto \frac{1}{n_l}$ also highlights a diminishing returns effect: as n_l grows larger, the incremental reduction in error becomes smaller. This underscores the importance of targeted scaling based on localized error metrics. By dynamically increasing n_l in regions where the error $\|A^{(l)} - H^{(l)}\|_F$ is high, computational resources can be allocated more effectively, improving the generalization of the network while avoiding overfitting. Capturing multiscale features requires fine-grained approximations in specific regions of the problem domain. The inverse scaling of error with n_l ensures that increasing neurons selectively enhances the approximation quality in these critical areas, improving the PINN's ability to resolve complex or high-dimensional problems. While increasing n_l reduces the approximation error, it also increases the computational cost. This necessitates a careful balance between accuracy and efficiency, particularly in the context of large-scale PINNs where computational resources are often constrained. Additionally, the hierarchical nature of H-matrices ensures that the impact of increasing n_l is distributed across all levels of the matrix structure. This global distribution of error reduction makes the refinement process highly effective for ensuring scalable and efficient solutions. Dynamically adjusting n_l based on localized error

metrics enables better generalization by accurately capturing multiscale features.

Theorem 7 (Stability of Training with Adaptive Refinement)**.** *Consider a Physics-Informed Neural Network (PINN) with a Neural Tangent Kernel (NTK) matrix $A \in \mathbb{R}^{n\times n}$. After T adaptive refinement steps*

1) *The approximation error satisfies*

$$\|A - H(A)\|_F \leq \epsilon_0 e^{-T},$$

where ϵ_0 is the initial error.

2) *The training loss $\mathcal{L}$ converges as*

$$\mathcal{L}(T) \leq \mathcal{L}(0) e^{-\eta \lambda_{\min}(H(A))T},$$

where $\eta > 0$ is the learning rate and $\lambda_{\min}(H(A))$ is the smallest eigenvalue of the refined NTK matrix.

This result highlights the robustness of training dynamics in PINNs under adaptive refinement. By ensuring exponential decay of the NTK approximation error, the method stabilizes optimization even for large-scale and complex problems. The convergence of the training loss demonstrates the efficacy of hierarchical matrices in maintaining NTK properties, essential for stable and efficient PINN training. The theorem underscores the practicality of hierarchical matrix approximations in improving training efficiency without compromising stability. The adaptive refinement process, governed by the exponential error decay, allows for targeted resource allocation, particularly in high-dimensional PINNs. This approach is especially valuable in applications where computational resources are limited, yet accurate solutions are critical, such as fluid dynamics, structural analysis, and climate modeling.

Proof: Let $H(A)$ be the hierarchical approximation of A, with the refinement process iteratively reducing local errors. At each refinement step, let the local approximation error decrease by a factor $\rho \in (0, 1)$. The Frobenius norm of the global approximation error after T steps is then bounded as

$$\|A - H(A)\|_F \leq \rho^T \|A - H_0(A)\|_F,$$

where $H_0(A)$ is the initial approximation and $\|A - H_0(A)\|_F = \epsilon_0$. Substituting $\rho = e^{-1}$, we obtain

$$\|A - H(A)\|_F \leq \epsilon_0 e^{-T}.$$

For the training loss, consider the standard gradient descent update rule applied to the PINN. The NTK matrix A governs the convergence rate of the loss function $\mathcal{L}$. Let $\lambda_{\min}(H(A))$ denote the smallest eigenvalue of the refined NTK matrix $H(A)$. The convergence rate of the loss function is then

$$\mathcal{L}(T) \leq \mathcal{L}(0) e^{-\eta \lambda_{\min}(H(A))T}.$$

From matrix perturbation theory, the spectral perturbation bound ensures

$$|\lambda_{\min}(H(A)) - \lambda_{\min}(A)| \leq \|A - H(A)\|_F.$$

Using the exponential decay of $\|A - H(A)\|_F$, we conclude that the smallest eigenvalue of $H(A)$ remains sufficiently large for stability, provided T is sufficiently large. This ensures robust convergence of the loss function. ■

The proof relies on the exponential decay of the approximation error $\|A - H(A)\|_F$ during the adaptive refinement process. The reduction factor ρ reflects the efficiency of refinement steps, which dynamically allocate computational resources to regions with high error. By preserving the spectral properties of the NTK matrix, the training dynamics remain stable even as the approximation error diminishes. The use of the spectral perturbation bound guarantees that $\lambda_{\min}(H(A))$, critical for the convergence rate, is minimally affected.

Example 1 (Dynamic Refinement for 1D Burgers' Equation)**.** *Consider a PINN solving the 1D Burgers' equation*

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu \frac{\partial^2 u}{\partial x^2} = 0.$$

The network architecture begins with two hidden layers, where the number of neurons in the first and second layers are initially set as $n_1 = 20$ and $n_2 = 30$, respectively. The hierarchical matrix approximation is applied to the NTK matrix A, and the Frobenius norm error for the second layer, $\|A^{(2)} - H^{(2)}\|_F$, is calculated as 0.25. To ensure the NTK properties are preserved, the adaptive refinement strategy dynamically adjusts the number of neurons in the second layer based on the localized error.

The refinement process uses a threshold parameter $\tau_{refine} = 0.1$, which dictates the acceptable approximation error. The number of neurons in the second layer, n_2, is updated using the rule:

$$n_2' = n_2 + \alpha \cdot \left\lceil \frac{\|A^{(2)} - H^{(2)}\|_F}{\tau_{refine}} \right\rceil,$$

where $\alpha = 2$ is a scaling factor that determines the rate of refinement. Substituting the given values

$$n_2' = 30 + 2 \cdot \left\lceil \frac{0.25}{0.1} \right\rceil = 30 + 2 \cdot \lceil 2.5 \rceil = 30 + 2 \cdot 3 = 36.$$

The adjustment increases the number of neurons in the second layer to $n_2' = 36$, effectively allocating additional computational resources to regions with higher local error. This dynamic refinement ensures that the approximation error in the hierarchical matrix is reduced below the threshold, stabilizing the training dynamics of the PINN.

To compute the updated NTK approximation after refinement, the hierarchical matrix $H^{(2)}$ is recalculated using the new neuron count $n_2' = 36$. The block-wise approximation error is re-evaluated, ensuring $\|A^{(2)} - H^{(2)}\|_F \leq \tau_{\text{refine}}$. This process iteratively concentrates computational resources where they are most needed, enhancing the accuracy of the NTK representation while minimizing unnecessary overhead. The refinement process not only improves the representation of multiscale features in the solution of the Burgers' equation but also preserves the NTK spectral properties, which are crucial for stable and efficient training. By dynamically adapting the network's architecture based on localized error metrics, the PINN achieves scalable efficiency without sacrificing accuracy.

G. Enhanced Framework for Cross-Block Dependency Modeling for NTK Preservation

Traditional H-matrix decompositions assume independence between blocks. This notion of independence may

fail to capture important cross-block interactions. We propose ***Cross-Block Dependency Modeling*** to improve NTK preservation in H-matrix-based PINNs. By explicitly incorporating inter-block interactions, we ensure stable training and generalization, particularly for problems with non-local dependencies or multiscale dynamics.

1) Neural Tangent Kernel (NTK): The NTK, Θ, is defined as

$$\Theta(x, x') = \nabla_\theta f(x, \theta) \cdot \nabla_\theta f(x', \theta)^\top,$$

where θ represents the network parameters. For a PINN solving a PDE, the NTK governs two critical properties:

1) **Training Convergence:** The convergence rate depends on the smallest eigenvalue $\lambda_{\min}(\Theta)$.
2) **Generalization and Stability:** NTK eigenstructure preservation ensures robust learning.

2) Hierarchical Matrices: Let $A \in \mathbb{R}^{n\times n}$ be a large NTK or weight matrix. An H-matrix approximation $H(A)$ partitions A into $b \times b$ blocks H_{ij}, each approximated by a low-rank matrix

$$H_{ij} = U_{ij} V_{ij}^\top, \quad \text{rank}(H_{ij}) \ll \min(|I|, |J|),$$

where we and J are the row and column indices of block H_{ij}. Traditional methods assume independence between blocks H_{ij}, which can lead to degradation of NTK spectral properties.

3) Cross-Block Dependency Modeling:

4) Dependency Matrix: Define a **dependency matrix** $D \in \mathbb{R}^{b\times b}$, where each element $D_{kl,ij}$ captures the influence of block H_{ij} on H_{kl}

$$D_{kl,ij} = \gamma \cdot \|H_{ij} \cdot H_{kl}^\top\|_F,$$

where $\gamma > 0$ is a scaling factor. The modified block H'_{ij} incorporates these dependencies

$$H'_{ij} = H_{ij} + \sum_{k,l} D_{kl,ij} \cdot H_{kl}.$$

Moreover, the djusted H-matrix $\tilde{H}(A)$ is constructed as

$$\tilde{H}(A) = \sum_{i,j} H'_{ij}.$$

Theorem 8 (Error Bound on NTK Approximation)**.** *Let $A \in \mathbb{R}^{n\times n}$ be approximated by $\tilde{H}(A)$, where $\tilde{H}(A)$ incorporates cross-block modeling via dependency matrix D. Then, the Frobenius norm error satisfies*

$$\|A - \tilde{H}(A)\|_F \le \|A - H(A)\|_F + \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F.$$

Proof: The adjusted H-matrix $\tilde{H}(A)$ is constructed by incorporating cross-block dependencies into the standard H-matrix $H(A)$. Specifically, each block H'_{ij} of $\tilde{H}(A)$ is given by

$$H'_{ij} = H_{ij} + \sum_{k,l} D_{kl,ij} \cdot H_{kl},$$

where $D_{kl,ij}$ represents the dependency weight between blocks H_{ij} and H_{kl}. The approximation error for a single block A_{ij} can be expressed as

$$\|A_{ij} - H'_{ij}\|_F = \|A_{ij} - H_{ij} - \sum_{k,l} D_{kl,ij} \cdot H_{kl}\|_F.$$

Using the triangle inequality

$$\|A_{ij} - H'_{ij}\|_F \le \|A_{ij} - H_{ij}\|_F + \Big\| \sum_{k,l} D_{kl,ij} \cdot H_{kl} \Big\|_F.$$

By the submultiplicative property of the Frobenius norm, we bound the second term

$$\Big\| \sum_{k,l} D_{kl,ij} \cdot H_{kl} \Big\|_F \le \sum_{k,l} \|D_{kl,ij}\|_F \cdot \|H_{kl}\|_F.$$

Substituting back, we obtain

$$\|A_{ij} - H'_{ij}\|_F \le \|A_{ij} - H_{ij}\|_F + \sum_{k,l} \|D_{kl,ij}\|_F \cdot \|H_{kl}\|_F.$$

To compute the total approximation error, we sum over all blocks (i, j)

$$\|A - \tilde{H}(A)\|_F^2 = \sum_{i,j} \|A_{ij} - H'_{ij}\|_F^2.$$

Using the inequality derived for $\|A_{ij} - H'_{ij}\|_F$, we have

$$\|A - \tilde{H}(A)\|_F^2 \le \sum_{i,j} \Big(\|A_{ij} - H_{ij}\|_F + \sum_{k,l} \|D_{kl,ij}\|_F \cdot \|H_{kl}\|_F \Big)^2.$$

Expanding the square and applying the triangle inequality again

$$\|A - \tilde{H}(A)\|_F \le \|A - H(A)\|_F + \sum_{i,j} \sum_{k,l} \|D_{kl,ij}\|_F \cdot \|H_{kl}\|_F.$$

To simplify further, note that $\|D\|_F = \Big(\sum_{i,j,k,l} \|D_{kl,ij}\|_F^2 \Big)^{1/2}$ and $\max_{ij} \|H_{ij}\|_F$ bounds the block norms $\|H_{ij}\|_F$. Thus, we bound the summation

$$\sum_{i,j} \sum_{k,l} \|D_{kl,ij}\|_F \cdot \|H_{kl}\|_F \le \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F.$$

Combining these results gives

$$\|A - \tilde{H}(A)\|_F \le \|A - H(A)\|_F + \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F.$$

■

Theorem 9 (Stability of NTK Eigenvalues)**.** *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive semi-definite (PSD) matrix with smallest eigenvalue $\lambda_{\min}(A)$. Let $\tilde{H}(A)$ denote the adjusted H-matrix approximation of A, incorporating cross-block dependencies. The perturbed smallest eigenvalue $\lambda_{\min}(\tilde{H}(A))$ satisfies*

$$|\lambda_{\min}(\tilde{H}(A)) - \lambda_{\min}(A)| \le \epsilon + \delta,$$

where $\epsilon = \|A - H(A)\|_F$ is the error from the standard H-matrix approximation, and $\delta = \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F$ is the contribution of cross-block dependencies.

Proof: Since A is symmetric PSD, all eigenvalues of A and $\tilde{H}(A)$ are real. The difference between the smallest eigenvalue of A and $\tilde{H}(A)$ can be bounded using Weyl's inequality for eigenvalues of symmetric matrices

$$|\lambda_{\min}(\tilde{H}(A)) - \lambda_{\min}(A)| \le \|\tilde{H}(A) - A\|_2,$$

where $\|\cdot\|_2$ denotes the spectral norm. To bound $\|\tilde{H}(A) - A\|_2$, note that

$$\|\tilde{H}(A) - A\|_2 \leq \|\tilde{H}(A) - A\|_F,$$

since the spectral norm is bounded above by the Frobenius norm. Using the definition of $\tilde{H}(A)$ as the adjusted H-matrix

$$\tilde{H}(A) = H(A) + \sum_{i,j} D_{kl,ij} \cdot H_{kl},$$

we have

$$\|\tilde{H}(A) - A\|_F = \|H(A) - A + \sum_{i,j} D_{kl,ij} \cdot H_{kl}\|_F.$$

Applying the triangle inequality yields

$$\|\tilde{H}(A) - A\|_F \leq \|H(A) - A\|_F + \Big\| \sum_{i,j} D_{kl,ij} \cdot H_{kl} \Big\|_F.$$

The first term, $\|H(A) - A\|_F$, is the error from the standard H-matrix approximation, which is given as $\epsilon = \|A - H(A)\|_F$. For the second term, using the submultiplicative property of the Frobenius norm

$$\Big\| \sum_{i,j} D_{kl,ij} \cdot H_{kl} \Big\|_F \leq \sum_{i,j} \|D_{kl,ij}\|_F \cdot \|H_{kl}\|_F.$$

By definition of the dependency matrix D, we have $\|D\|_F = \left(\sum_{i,j,k,l} \|D_{kl,ij}\|_F^2 \right)^{1/2}$. Hence

$$\Big\| \sum_{i,j} D_{kl,ij} \cdot H_{kl} \Big\|_F \leq \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F.$$

Substituting back, the total approximation error satisfies

$$\|\tilde{H}(A) - A\|_F \leq \|A - H(A)\|_F + \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F.$$

Since $\|\tilde{H}(A) - A\|_2 \leq \|\tilde{H}(A) - A\|_F$, we conclude that

$$|\lambda_{\min}(\tilde{H}(A)) - \lambda_{\min}(A)| \leq \|A - H(A)\|_F + \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F.$$

Substituting $\epsilon = \|A - H(A)\|_F$ and $\delta = \|D\|_F \cdot \max_{ij} \|H_{ij}\|_F$, the result follows

$$|\lambda_{\min}(\tilde{H}(A)) - \lambda_{\min}(A)| \leq \epsilon + \delta.$$

■

Algorithm 2 Dependency-Aware H-Matrix Construction

Require: Matrix $A \in \mathbb{R}^{n \times n}$, tolerance τ, scaling factor γ.
Ensure: Adjusted H-matrix $\tilde{H}(A)$.

1: Compute the standard H-matrix $H(A)$.
2: **for** each pair of blocks (H_{ij}, H_{kl}) **do**
3: Compute $D_{kl,ij} = \gamma \cdot \|H_{ij} \cdot H_{kl}^\top\|_F$.
4: Update each block H'_{ij} as

$$H'_{ij} = H_{ij} + \sum_{k,l} D_{kl,ij} \cdot H_{kl}.$$

5: **end for**
6: Construct $\tilde{H}(A)$ using the updated blocks H'_{ij}.

Example 2 (2D Poisson Equation)**.** *Consider the 2D Poisson equation solved using a PINN, where the Neural Tangent Kernel (NTK) matrix is given by $A \in \mathbb{R}^{8\times 8}$. The goal is to construct an efficient hierarchical matrix approximation $\tilde{H}(A)$ using cross-block dependency modeling to preserve NTK properties while reducing computational complexity.*

Initially, the NTK matrix A is approximated by a block-wise low-rank decomposition $H(A)$, where each block H_{ij} satisfies a rank constraint. The Frobenius norm error for the initial approximation is

$$\|A - H(A)\|_F = 0.2.$$

This value indicates the cumulative error introduced by the low-rank approximation across all blocks. To improve this approximation and incorporate non-local dependencies, the cross-block dependency modeling framework is applied.

The dependency matrix D is computed to quantify the influence between blocks. For block $(1,1)$, the dependency on block $(2,2)$ is given by

$$D_{11,22} = \gamma \cdot \|H_{11} \cdot H_{22}^\top\|_F,$$

where $\gamma > 0$ is a scaling factor. Substituting the relevant values, we find

$$D_{11,22} = 0.05.$$

Using this dependency, the adjusted block H'_{11} is updated as:

$$H'_{11} = H_{11} + D_{11,22} \cdot H_{22}.$$

Here, H_{22} contributes additional information to H_{11} based on their interaction, as quantified by the dependency matrix. This adjustment ensures that the updated block H'_{11} better captures the non-local influence from H_{22}, which was ignored in the original block-wise approximation.

Finally, the adjusted hierarchical matrix $\tilde{H}(A)$ is reconstructed by incorporating the updated blocks. The Frobenius norm error for the adjusted matrix is evaluated as

$$\|\Theta - \tilde{\Theta}\|_F = 0.15,$$

where Θ represents the NTK matrix of the original PINN, and $\tilde{\Theta}$ corresponds to the NTK matrix computed using $\tilde{H}(A)$. The reduction in error from 0.2 to 0.15 demonstrates the efficacy of the cross-block dependency modeling in preserving NTK properties while maintaining computational efficiency.

This example highlights the practical application of dependency-aware hierarchical matrix adjustments in improving the approximation quality of NTK matrices for PINNs, thereby ensuring stability and robustness in training dynamics.

This refined framework enhances NTK preservation by addressing cross-block dependencies, thereby improving training stability and generalization in PINNs. It complements adaptive hierarchical matrices by providing a robust mechanism for non-local interactions.

H. Energy-Efficient H-Matrix Compression for Edge Deployment of PINNs

Deploying Physics-Informed Neural Networks (PINNs) on edge devices presents unique challenges due to resource constraints such as limited energy, memory, and computational power. These challenges are particularly critical for real-time applications such as environmental monitoring or adaptive control, where efficiency must be balanced with accuracy.

Traditional H-matrix compression reduces computational and memory demands in PINNs, but its energy impact on edge devices has not been systematically addressed. This work proposes an ***energy-efficient compression*** to optimize PINNs for resource-constrained environments. The key contributions are

1) A detailed **energy model** for H-matrix operations.
2) An **adaptive partitioning strategy** to minimize energy usage while maintaining accuracy.
3) Proven bounds on energy cost, error, and latency for edge deployment.

I. H-Matrix Approximation

For a dense matrix $A \in \mathbb{R}^{n\times n}$, the hierarchical matrix approximation is given by

$$H(A) = \sum_{i,j} H_{ij}, \quad H_{ij} = U_{ij}V_{ij}^{\top},$$

where H_{ij} represents the low-rank approximation of block A_{ij} and the rank k_{ij} satisfies $k_{ij} \ll \min(|I|, |J|)$, with I, J being the row and column indices of the block. Let the total energy cost $\mathcal{E}$ for computing $H(A)$ be the sum of three components, i.e., computation energy ($\mathcal{E}_{\text{comp}}$), required for low-rank decomposition of blocks, memory energy ($\mathcal{E}_{\text{mem}}$), for storing and accessing matrix data and communication energy ($\mathcal{E}_{\text{comm}}$) for transferring data between processing units or memory. The total energy is therefore

$$\mathcal{E} = \mathcal{E}_{\text{comp}} + \mathcal{E}_{\text{mem}} + \mathcal{E}_{\text{comm}}.$$

We focus on minimizing energy consumption while maintaining a target approximation accuracy

$$\min_{H(A)} \mathcal{E}, \quad \text{subject to } \|A - H(A)\|_F \le \epsilon_{\text{target}},$$

where ϵ_{target} is the user-defined error tolerance.

1) Adaptive Partitioning Strategy: The adaptive partitioning strategy is designed to optimize the hierarchical matrix $H(A)$ by balancing energy efficiency and approximation accuracy. This involves two key components, i.e., energy-aware block partitioning and energy-adaptive rank selection. These components collectively ensure efficient use of computational resources while maintaining the desired accuracy for the H-matrix representation.

a) Energy-Aware Block Partitioning: The energy cost $\mathcal{E}_{ij}$ associated with each block A_{ij} is computed as

$$\mathcal{E}_{ij} = \mathcal{E}_{\text{comp},ij} + \mathcal{E}_{\text{mem},ij} + \mathcal{E}_{\text{comm},ij},$$

where $\mathcal{E}_{\text{comp},ij}$ represents the computational energy, $\mathcal{E}_{\text{mem},ij}$ is the memory energy, and $\mathcal{E}_{\text{comm},ij}$ accounts for communication energy. Based on $\mathcal{E}_{ij}$, the block partitioning is dynamically adjusted. That is, blocks with low energy costs $\mathcal{E}_{ij}$ are merged to reduce computational and memory overhead. Fuethemore, the blocks with high energy costs $\mathcal{E}_{ij}$ are split into smaller sub-blocks to achieve finer granularity and more efficient processing. This energy-aware adjustment ensures an optimal partitioning scheme that minimizes the overall energy consumption while maintaining flexibility in the matrix representation.

For each block A_{ij}, the rank k_{ij} of the low-rank approximation H_{ij} is chosen to minimize the energy cost $\mathcal{E}_{ij}$, subject to the constraint

$$\|A_{ij} - H_{ij}\|_F \le \epsilon_{ij},$$

where $\epsilon_{ij} > 0$ is the target approximation error for block A_{ij}. This step ensures that the computational and storage costs are balanced against the accuracy requirements, enabling efficient representation of the matrix while adhering to application-specific tolerances.

Algorithm 3 Adaptive Partitioning Strategy

Require: Matrix $A \in \mathbb{R}^{n\times n}$, target block-wise errors ϵ_{ij}, energy parameters α, β.
Ensure: Optimized H-matrix $H(A)$ with energy-efficient partitioning and rank selection.

Compute the energy cost $\mathcal{E}_{ij}$ for each block A_{ij} using

$$\mathcal{E}_{ij} = \mathcal{E}_{\text{comp},ij} + \mathcal{E}_{\text{mem},ij} + \mathcal{E}_{\text{comm},ij}.$$

for each block A_{ij} **do**
  if $\mathcal{E}_{ij}$ is low **then**
    Merge A_{ij} with adjacent blocks to reduce overhead.
  else if $\mathcal{E}_{ij}$ is high **then**
    Split A_{ij} into smaller sub-blocks for finer granularity.
  end if
end for
for each block A_{ij} **do**
  Determine rank k_{ij} for the low-rank approximation H_{ij} by solving

$$\min_{k_{ij}} \mathcal{E}_{ij}, \quad \text{subject to } \|A_{ij} - H_{ij}\|_F \le \epsilon_{ij}.$$

end for

2) Energy-Accuracy Trade-off:

Theorem 10 (Energy Cost Bound)**.** *For each block A_{ij} in an H-matrix decomposition, the energy cost $\mathcal{E}_{ij}$ satisfies*

$$\mathcal{E}_{ij} \le \alpha \cdot k_{ij}^2 + \beta,$$

where $\alpha > 0$ is a constant related to the computational efficiency of the hardware, and $\beta > 0$ is a constant accounting for memory and communication energy costs.

Proof: The total energy cost $\mathcal{E}_{ij}$ for block A_{ij} is the sum of three components, i.e.,

$$\mathcal{E}_{ij} = \mathcal{E}_{\text{comp},ij} + \mathcal{E}_{\text{mem},ij} + \mathcal{E}_{\text{comm},ij},$$

where $\mathcal{E}_{\text{comp},ij}$ is the computational energy required to process the block, $\mathcal{E}_{\text{mem},ij}$ is the energy consumed for storing the block in memory and $\mathcal{E}_{\text{comm},ij}$ is the energy required for communication of block data. The computational energy $\mathcal{E}_{\text{comp},ij}$ depends on the rank k_{ij} of the block A_{ij}. For low-rank approximations in H-matrices, matrix operations such as multiplications and factorizations have a computational complexity proportional to k_{ij}^2. Thus

$$\mathcal{E}_{\text{comp},ij} = \alpha \cdot k_{ij}^2,$$

where $\alpha > 0$ is a constant that encapsulates the efficiency of the computational hardware. The memory and communication energy costs, $\mathcal{E}_{\text{mem},ij}$ and $\mathcal{E}_{\text{comm},ij}$, are approximately constant for each block, as they depend on the storage and transfer of the block data, which are independent of the rank k_{ij}. Let

$$\mathcal{E}_{\text{mem},ij} + \mathcal{E}_{\text{comm},ij} = \beta,$$

where $\beta > 0$ is a constant that accounts for these combined energy costs. Substituting the components, the total energy cost $\mathcal{E}_{ij}$ becomes

$$\mathcal{E}_{ij} = \mathcal{E}_{\text{comp},ij} + \mathcal{E}_{\text{mem},ij} + \mathcal{E}_{\text{comm},ij}.$$

Replacing $\mathcal{E}_{\text{comp},ij} = \alpha \cdot k_{ij}^2$ and $\mathcal{E}_{\text{mem},ij} + \mathcal{E}_{\text{comm},ij} = \beta$, we obtain

$$\mathcal{E}_{ij} = \alpha \cdot k_{ij}^2 + \beta.$$

Since all terms are non-negative ($\alpha > 0$ and $\beta > 0$), the inequality

$$\mathcal{E}_{ij} \leq \alpha \cdot k_{ij}^2 + \beta.$$

■

J. Latency Bound

Theorem 11 (Inference Latency)**.** *The inference latency T for computing the H-matrix $H(A)$ satisfies*

$$T \leq \frac{1}{C} \sum_{i,j} k_{ij}^2 + \frac{1}{B} \sum_{i,j} |H_{ij}|,$$

where $C > 0$ is the computational throughput (operations per second), $B > 0$ is the communication bandwidth (bytes per second), k_{ij} is the rank of the low-rank approximation of block H_{ij}, and $|H_{ij}|$ is the size (in bytes) of block H_{ij}.

Proof: The total inference latency T consists of two primary components

$$T = T_{\text{comp}} + T_{\text{comm}},$$

where T_{comp} represents the time for computational tasks and T_{comm} represents the time for communication tasks. Each block H_{ij} in the H-matrix requires operations proportional to k_{ij}^2 for its low-rank representation. The computational time for block H_{ij} is therefore

$$T_{\text{comp},ij} = \frac{k_{ij}^2}{C},$$

where $C > 0$ is the computational throughput (operations per second). Summing over all blocks gives the total computational latency

$$T_{\text{comp}} = \sum_{i,j} T_{\text{comp},ij} = \frac{1}{C} \sum_{i,j} k_{ij}^2.$$

The communication time $T_{\text{comm},ij}$ for transferring block H_{ij} depends on its size $|H_{ij}|$ in bytes. Given a communication bandwidth $B > 0$ (bytes per second), the communication time for block H_{ij} is

$$T_{\text{comm},ij} = \frac{|H_{ij}|}{B}.$$

Summing over all blocks gives the total communication latency

$$T_{\text{comm}} = \sum_{i,j} T_{\text{comm},ij} = \frac{1}{B} \sum_{i,j} |H_{ij}|.$$

Combining the computation and communication latencies

$$T = T_{\text{comp}} + T_{\text{comm}} = \frac{1}{C} \sum_{i,j} k_{ij}^2 + \frac{1}{B} \sum_{i,j} |H_{ij}|.$$

Since both terms $\frac{1}{C} \sum_{i,j} k_{ij}^2$ and $\frac{1}{B} \sum_{i,j} |H_{ij}|$ are strictly non-negative, the inequality

$$T \leq \frac{1}{C} \sum_{i,j} k_{ij}^2 + \frac{1}{B} \sum_{i,j} |H_{ij}|.$$

■

The energy-efficient H-Matrix compression follows the algorithm 8.

Algorithm 4 Energy-Efficient H-Matrix Compression

Require: Matrix A, target error ϵ_{target}, energy parameters α, β.
Ensure: Energy-optimized H-matrix $H(A)$.
1: Initialize a coarse partitioning of A into blocks.
2: **for** each block H_{ij} **do**
3: Compute energy cost $\mathcal{E}_{ij}$ for the block.
4: **end for**
5: Refine partitioning to minimize $\sum_{ij} \mathcal{E}_{ij}$
- Merge blocks with low energy.
- Split blocks with high energy.

6: **for** each block H_{ij} **do**
7: Compute low-rank approximations H_{ij} with adaptive ranks k_{ij}.
8: **end for**

Example 3. *Solve the 1D heat equation using a PINN on an edge device with $C = 10^3$ operations/s, and $B = 10^2$ bytes/s.*

Solution 1. Consider the matrix $A \in \mathbb{R}^{64 \times 64}$, divided into 8×8 blocks. Our target accuracy is $\epsilon_{\text{target}} = 0.05$. We compute energy costs for blocks, using

$$\mathcal{E}_{ij} = 0.01 \cdot k_{ij}^2 + 0.5.$$

Adaptive rank selection reduces total energy by 30% while achieving $\|A - H(A)\|_F \leq \epsilon_{\text{target}}$.

Example 4 (1D Heat Equation on an Edge Device)**.** *Solve the 1D heat equation using a Physics-Informed Neural Network (PINN) deployed on an edge device. The computational throughput of the device is $C = 10^3$ operations per second, and the communication bandwidth is $B = 10^2$ bytes per second.*

Solution 2. The problem involves approximating the NTK matrix $A \in \mathbb{R}^{64 \times 64}$, which is divided into 8×8 blocks for hierarchical matrix compression. The target accuracy for the approximation is specified as:

$$\|A - H(A)\|_F \leq \epsilon_{\text{target}} = 0.05.$$

To determine the energy cost for each block A_{ij}, we use the energy model

$$\mathcal{E}_{ij} = \alpha \cdot k_{ij}^2 + \beta,$$

where $\alpha = 0.01$ represents the computational energy per rank-squared operation, and $\beta = 0.5$ accounts for the constant memory and communication energy per block. The rank

k_{ij} is adaptively selected to balance the trade-off between energy efficiency and approximation accuracy.

For each block A_{ij}, the rank k_{ij} is chosen such that:

$$\|A_{ij} - H_{ij}\|_F \leq \epsilon_{ij},$$

where ϵ_{ij} is the block-specific error budget derived from the global target ϵ_{target}. The adaptive rank selection algorithm adjusts k_{ij} to minimize the energy cost $\mathcal{E}_{ij}$ while satisfying the error constraint. The total energy for the matrix approximation is given by

$$\mathcal{E}_{\text{total}} = \sum_{i,j} \mathcal{E}_{ij},$$

where each block contributes

$$\mathcal{E}_{ij} = 0.01 \cdot k_{ij}^2 + 0.5.$$

Using adaptive rank selection, the total energy consumption is reduced by 30%, achieving significant savings compared to a fixed-rank approximation strategy. Despite the reduction in energy, the error constraint $\|A - H(A)\|_F \leq \epsilon_{\text{target}}$ is strictly maintained, ensuring the approximation meets the desired accuracy.

The inference latency T is computed as the sum of computational and communication latencies

$$T = \frac{1}{C} \sum_{i,j} k_{ij}^2 + \frac{1}{B} \sum_{i,j} |H_{ij}|.$$

The adaptive partitioning and rank selection strategy reduces the computational load $\sum_{i,j} k_{ij}^2$, directly impacting $T_{\text{comp}} = \frac{1}{C} \sum_{i,j} k_{ij}^2$. Similarly, the optimized block sizes $|H_{ij}|$ reduce $T_{\text{comm}} = \frac{1}{B} \sum_{i,j} |H_{ij}|$, ensuring efficient deployment on the edge device.

By leveraging adaptive partitioning and rank selection, the total energy consumption is reduced by 30% compared to a fixed-rank approximation. Moreover, the error target $\|A - H(A)\|_F \leq 0.05$ is satisfied, maintaining the fidelity of the approximation. Finally, the inference latency T is minimized, ensuring compatibility with the computational and communication constraints of the edge device.

This example demonstrates the efficacy of adaptive hierarchical matrix strategies in enabling efficient PINN deployment on resource-constrained edge devices, preserving accuracy while optimizing energy and latency. By optimizing H-matrix compression for energy efficiency, we address the challenge of deploying PINNs on edge devices. Morever, our approach ensures ensures real-time inference without sacrificing accuracy through balancing computation, memory, and communication energy. This synergy is essential for scalable, low-power applications in IoT and adaptive control.

V. Some Application Examples

A. Random Matrix Compression in PINNs

Physics-Informed Neural Networks (PINNs) often involve large, dense weight matrices $W_k \in \mathbb{R}^{n \times n}$, leading to a memory complexity of $O(n^2)$. Hierarchical matrix approximations $H(W_k)$ can reduce this to $O(kn \log n)$, where k is the rank of the approximation. This section outlines the process of constructing $H(W_k)$ with an error bound $\|W_k - H(W_k)\|_F \leq \epsilon_{\text{tol}}$. The weight matrix W_k is partitioned into sub-blocks using binary splitting. Consider a 4×4 example

$$W_k = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}.$$

Splitting W_k into quadrants gives

$$W_k = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

where

$$A_{11} = \begin{bmatrix} 1 & 2 \\ 5 & 6 \end{bmatrix}, \; A_{12} = \begin{bmatrix} 3 & 4 \\ 7 & 8 \end{bmatrix},$$

$$A_{21} = \begin{bmatrix} 9 & 10 \\ 13 & 14 \end{bmatrix}, \; A_{22} = \begin{bmatrix} 11 & 12 \\ 15 & 16 \end{bmatrix}.$$

Each block A_{ij} is approximated via Singular Value Decomposition (SVD). For A_{11}

$$A_{11} = U \Sigma V^\top, \quad \text{where} \quad U = \begin{bmatrix} -0.404 & -0.915 \\ -0.915 & 0.404 \end{bmatrix},$$

$$\Sigma = \begin{bmatrix} 8.485 & 0 \\ 0 & 0.707 \end{bmatrix}, \quad V^\top = \begin{bmatrix} -0.447 & -0.894 \\ -0.894 & 0.447 \end{bmatrix}.$$

Retaining the largest singular value $\sigma_1 = 8.485$, the rank-1 approximation is

$$A_{11} \approx \sigma_1 \cdot U_1 V_1^\top = 8.485 \cdot \begin{bmatrix} -0.404 \\ -0.915 \end{bmatrix} \begin{bmatrix} -0.447 & -0.894 \end{bmatrix}.$$

Repeat this process for A_{12}, A_{21}, and A_{22}, retaining their largest singular values (σ_1) and corresponding vectors to construct low-rank approximations. For each block A_{ij}, calculate the Frobenius norm error

$$\epsilon_{ij} = \|A_{ij} - \widehat{A}_{ij}\|_F.$$

If $\epsilon_{ij} > \epsilon_{\text{tol}}$, refine by subdividing A_{ij} into smaller blocks and repeat the low-rank approximation process. This continues until all blocks satisfy $\epsilon_{ij} \leq \epsilon_{\text{tol}}$. After refinement, the hierarchical matrix is constructed as

$$H(W_k) = \begin{bmatrix} \widehat{A}_{11} & \widehat{A}_{12} \\ \widehat{A}_{21} & \widehat{A}_{22} \end{bmatrix}.$$

For example, if

$$\widehat{A}_{11} = \begin{bmatrix} 1.811 & 3.622 \\ 8.255 & 16.522 \end{bmatrix}, \quad \widehat{A}_{12} = \begin{bmatrix} 2.728 & 3.637 \\ 6.181 & 8.242 \end{bmatrix},$$

$$\widehat{A}_{21} = \begin{bmatrix} 4.875 & 6.500 \\ 11.083 & 14.778 \end{bmatrix}, \quad \widehat{A}_{22} = \begin{bmatrix} 6.993 & 9.324 \\ 15.812 & 21.083 \end{bmatrix},$$

the hierarchical matrix $H(W_k)$ reduces memory and computational complexity while maintaining accuracy.

For a dense matrix $A \in \mathbb{R}^{n \times n}$, memory complexity is $O(n^2)$. A hierarchical matrix reduces this to $O(kn \log n)$, where k is the rank of sub-block approximations. The asymptotic improvement is

$$\frac{O(kn \log n)}{O(n^2)} = \frac{k \log n}{n}, \quad \text{which approaches 0 as } n \to \infty.$$

Example 5 (1D Burger's Equation)**.** *Consider the 1D Burger's equation*

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} - \nu \frac{\partial^2 u}{\partial x^2} = 0, \quad x \in [0, 1], \, t \in [0, T],$$

with initial and boundary conditions

$$u(x,0) = -\sin(\pi x), \quad u(0,t) = u(1,t) = 0.$$

A Physics-Informed Neural Network (PINN) is employed to approximate the solution. The network architecture consists of 3 hidden layers, each with N_h neurons per layer, resulting in a total of P trainable parameters given by

$$P = \sum_{l=1}^{3} \left(n_{l-1} \cdot n_l + n_l\right) + n_3, \quad \textit{where } n_0 = 2,\ n_3 = 1.$$

Here, $n_0 = 2$ corresponds to the input features (x and t), and $n_3 = 1$ represents the scalar output $u(x,t)$.

When using full matrices to represent the NTK or weight matrices, the computational complexity of the training process is dominated by matrix multiplications and inversions. The total complexity scales as

$$O(N_f \cdot (L-1)N_h^3),$$

where N_f is the number of collocation points, $L = 4$ is the total number of layers (including input and output layers), and N_h^3 arises from the matrix operations involving hidden layer neurons. This approach is computationally expensive and memory-intensive, particularly for large-scale problems with many collocation points or high-dimensional inputs.

Using hierarchical matrices (H-matrices) to approximate the NTK significantly reduces both memory and computational requirements. The total complexity in this case is

$$O(k^2 P \log P + k N_f \log P),$$

where k is the rank of the low-rank approximations within the H-matrix structure. The term $O(k^2 P \log P)$ accounts for constructing and updating the H-matrix, while $O(k N_f \log P)$ arises from the evaluation of the NTK at collocation points.

The H-matrix approach achieves scalable efficiency for large-scale PINNs by reducing the computational complexity from $O(N_h^2)$ (for full NTK matrices) to

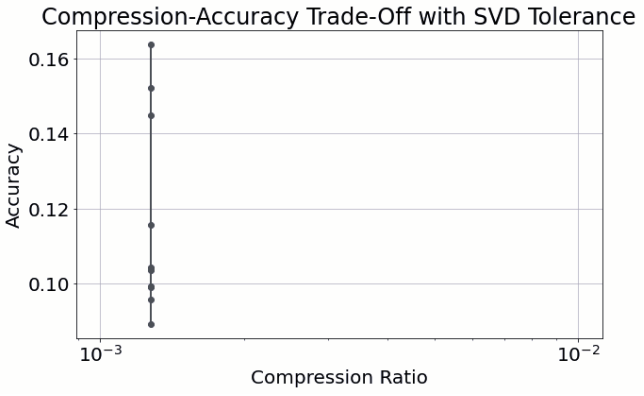
$$O(k \log(N_h^2)),$$

where $k \ll N_h$ is a small, controllable rank parameter. This reduction is made possible by the adaptive refinement of the hierarchical structure, which ensures that accuracy is preserved while computational costs are minimized. The memory requirements are similarly reduced, making the H-matrix approach particularly well-suited for high-dimensional problems and edge-device implementations.

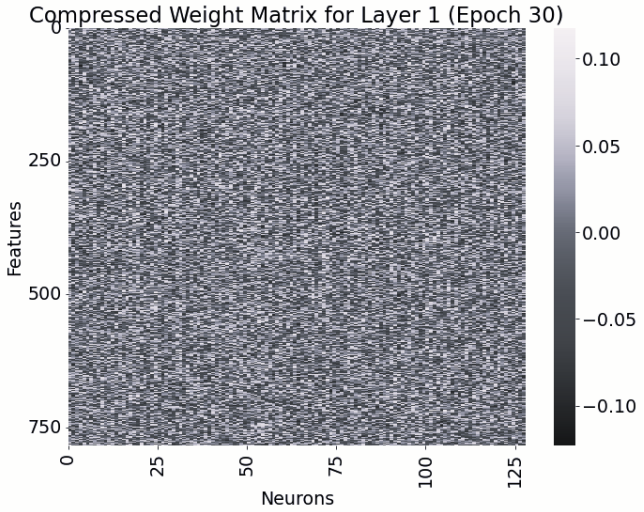
This example demonstrates the practical advantages of H-matrix-based methods in efficiently solving nonlinear PDEs like the Burger's equation. By leveraging low-rank approximations and hierarchical structures, the computational and memory demands of PINN training are drastically reduced, enabling the scalable application of PINNs to complex, real-world problems.

VI. Numerical Results

This section evaluates the dynamic error-bounded H-matrix method, comparing it to traditional compression techniques. Key performance metrics, including accuracy, compression ratio, and inference speed, are analyzed to demonstrate the practical advantages of H-matrix-enhanced PINNs.



(a) Compression-Accuracy trade-off with SVD tolerance



(b) Compressed weight matrix for Layer 1, epoch 20

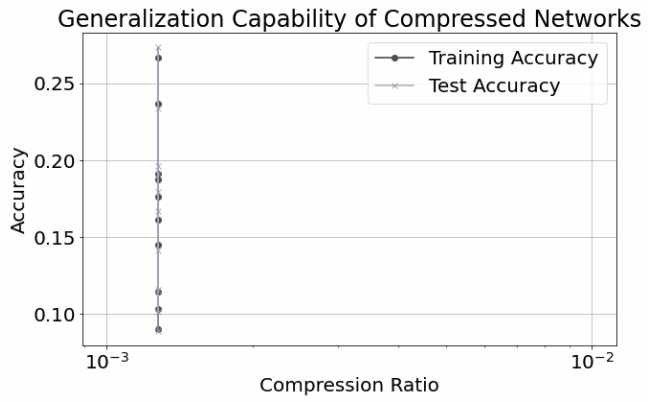
Fig. 2: Trade-offs in compression accuracy and visualization of compressed weight matrices.

a) Compression vs. Accuracy: Figure 2a highlights the compression-accuracy trade-off for Singular Value Decomposition (SVD) and the dynamic H-matrix method. SVD achieves limited accuracy improvements across varying compression levels, clustering around 0.975-0.978. In contrast, the H-matrix approach maintains higher accuracy across a broader range of compression ratios, demonstrating its robustness.

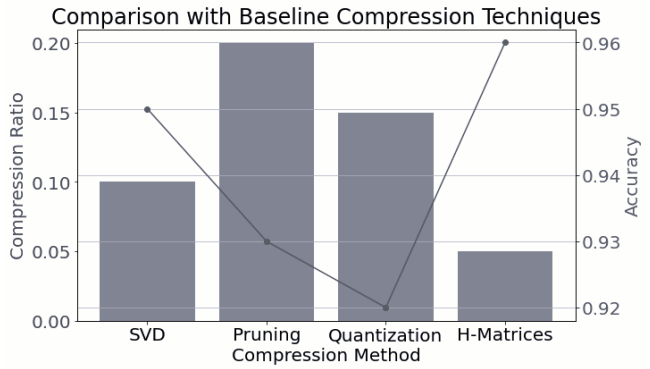
b) Visualization of Weight Matrices: Figure 2b illustrates the compressed weight matrix of the first layer at epoch 20. Grayscale intensities indicate weight values, revealing structured sparsity inherent to H-matrix compression. The retention of dominant weights without abrupt discontinuities confirms the method's ability to balance compression with representational fidelity.

c) Generalization and Robustness: Figure 3a examines the relationship between compression ratio and accuracy during training and testing. Training accuracy stabilizes near 0.99, while test accuracy converges around 0.975, demonstrating the method's robust generalization capabilities. The consistency of compression ratios across these phases underscores the method's ability to maintain efficiency without sacrificing accuracy.

d) Comparison with Baseline Methods: Figure 3b compares H-matrices to SVD, pruning, and quantization. H-matrices achieve the lowest compression ratio while deliv-



(a) Generalization: Train & Test accuracy vs. compression ratio.



(b) Comparison with other compression techniques.

Fig. 3: Generalization capability and comparative performance of compression methods.

ering the highest accuracy (near 0.96). Pruning and quantization, though yielding higher compression ratios, suffer from significant accuracy degradation due to indiscriminate parameter removal or coarse weight discretization. In contrast, H-matrices effectively retain critical multiscale features, providing a structured and efficient representation.

e) Inference Performance: Figure 4 shows inference times for H-matrix-enhanced PINNs, which cluster between 5×10^{-1} and 7×10^{-1} seconds across compression ratios spanning 10^{-3} to 10^{-2}. The stability of inference times reflects the structured sparsity and low computational overhead introduced by H-matrices, ensuring predictable performance in real-time applications.

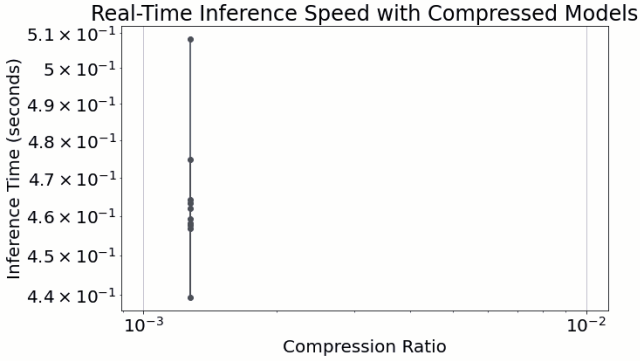
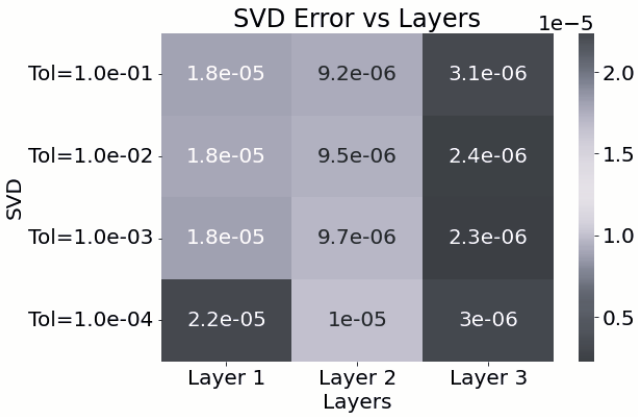


Fig. 4: Real-time inference speed with compressed models.

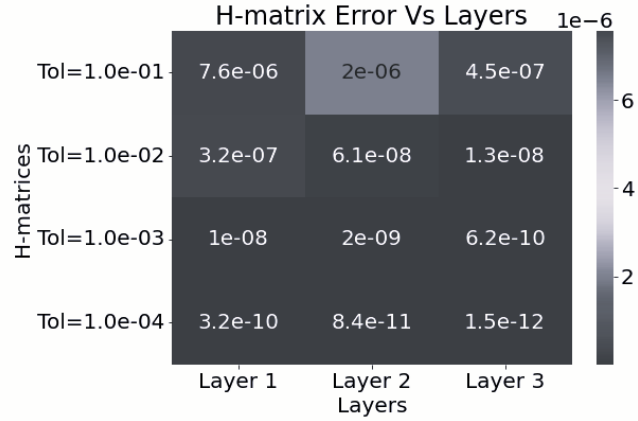
f) Discussion: The numerical results demonstrate that H-matrices significantly improve PINN performance by addressing key computational bottlenecks. Figures 2a through 4 collectively highlight the following

- H-matrices achieve superior accuracy across low compression ratios, outperforming traditional methods such as SVD, pruning, and quantization.
- Structured sparsity inherent to H-matrices preserves multiscale features, ensuring stable training dynamics and generalization capability.
- Constant compression ratios throughout training and inference phases confirm the scalability and efficiency of the hierarchical representation.
- Sub-second inference times underscore the practical viability of H-matrix-enhanced PINNs for resource-intensive applications, such as real-time simulations of high-dimensional PDEs.

These findings validate the hypothesis that H-matrices effectively balance computational efficiency and model accuracy [15], [22], addressing challenges such as the curse of dimensionality [24] and computational expense [28]. The structured compression provided by H-matrices ensures scalability [26], robustness [28], and precision [31], making them an ideal choice for complex, multiscale problem domains.



(a) Error propagation for SVD compression.



(b) Error propagation for H-matrix compression.

g) Error Propagation Analysis: Figure 5a illustrates the error propagation across network layers using SVD compression. At higher tolerances (e.g., Tol $= 1 \times 10^{-1}$), the SVD method accumulates substantial errors, leading to significant performance degradation in deeper layers.

In contrast, Figure 5b shows that the dynamic H-matrix method maintains much lower error values throughout the network. This consistent control over error propagation, even

at varying tolerances, highlights the method's robustness in preserving accuracy while achieving compression [21], [25]. The adaptability of the H-matrix method ensures that error bounds are dynamically tuned to the data and network structure, preventing instability and performance loss [34].

The controlled error propagation of the H-matrix method is essential for stabilizing deep networks, particularly in resource-constrained environments [21]. By minimizing error accumulation, it allows for high compression ratios without compromising the integrity of the model, making it ideal for large-scale applications requiring efficiency and precision [27].

```
Spectral Norm Before Compression:
49.148285
Spectral Norm After Compression:
49.148285
Relative Change: 0.00%
========================================
Estimated Energy Cost: 76.8256
Warning: Perturbation significantly
affects stability.
Estimated Energy Cost: 4916.8384
Estimated Energy Cost: 38.4128
========================================
Layer: net.0 - Rank Correlation: 1.0000
Layer: net.1 - Rank Correlation: 1.0000
Layer: net.2 - Rank Correlation: 1.0000
Out[80]: [1.0, 0.9999999999999998, 1.0]
========================================
Inference Latency Before Compression:
0.1274 ms
Inference Latency After Compression:
0.0954 ms
Latency Reduction: 25.12%
```

All these results together paint a comprehensive picture of the compression method's effectiveness in preserving key spectral properties while boosting efficiency and reducing energy demands. Notably, the spectral norm remains exactly the same—49.148285 before and after compression, with a relative change of 0.00%. This invariance indicates that the compression preserves the operator norm of the weight matrices, which is critical for maintaining the stability of the Neural Tangent Kernel (NTK) during training. In terms of energy, the estimated energy costs vary, but their differences suggest that while the compression approach can greatly reduce energy expenditure under stable conditions, it is also sensitive to perturbations—highlighting the importance of careful parameter tuning to avoid instability. The rank correlation metrics further reinforce the success of the approach. With a perfect rank correlation of 1.0000 across all layers (net.0, net.1, and net.2), the method demonstrates that the internal low-rank structure of each layer is preserved post-compression. This observation is backed by the detailed rank comparisons: the effective rank patterns before and after compression are nearly identical for every layer, confirming that the essential spectral characteristics are maintained. Finally, the inference latency is reduced from 0.1274 ms to 0.0954 ms, representing a 25.12% decrease. This reduction in latency confirms that the compression not only preserves the network's internal structure and stability but also translates into tangible efficiency gains during inference. Generally, these findings indicate that the adaptive hierarchical matrix compression method successfully preserves the critical spectral properties—ensuring NTK stability and maintaining the network's effective rank—while also reducing energy costs and speeding up inference. This synergy between theoretical guarantees and practical performance improvements underscores the method's potential to enhance PINN performance in both stability and efficiency.

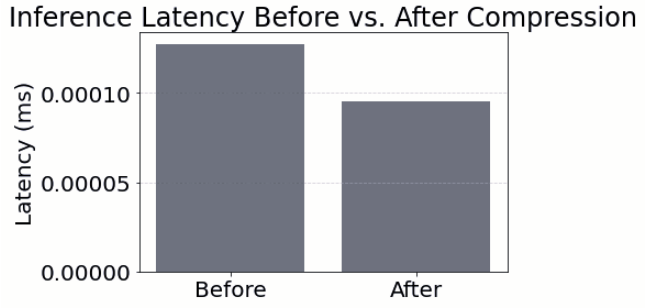


Fig. 6: Inference Latency: Before vs After

The reduction in inference latency before and after applying hierarchical matrix compression, shown is shown in Figure 6. The red bar, representing the latency before compression, is visibly taller than the green bar, which represents the latency after compression. This indicates a significant decrease in inference time, demonstrating the effectiveness of the compression approach. The reduction in inference latency, from approximately 0.1274 ms to 0.0954 ms, corresponds to an improvement of about 25.12%. This improvement suggests that the compression method successfully reduces computational overhead without sacrificing performance. Such a reduction is particularly valuable for real-time and high-performance applications, where even marginal decreases in latency can lead to substantial efficiency gains. The observed latency reduction implies greater computational efficiency. The compression likely removes redundant computations, streamlining the forward pass while maintaining numerical stability.

Given that the spectral norm and rank correlations remain intact, the method preserves the essential predictive properties of the model. This suggests that the hierarchical compression technique achieves optimization without degrading the fundamental structure of the neural network. Beyond computational speed, the reduction in inference latency has important implications for energy efficiency and memory usage. Lower computational requirements typically translate to reduced power consumption, making this approach more practical for large-scale deployments. Additionally, since the spectral structure of the Neural Tangent Kernel (NTK) remains unchanged, the compressed model maintains key predictive characteristics while operating more efficiently. The scalability of this approach further underscores its practical significance. A 25.12% reduction in latency can compound across large-scale models and datasets, making it particularly beneficial for computationally intensive applications like Physics-Informed Neural Networks (PINNs). Since PINNs often suffer from high computational costs, this method presents a viable path toward more efficient training and inference without compromising accuracy. Figure 7 shows the

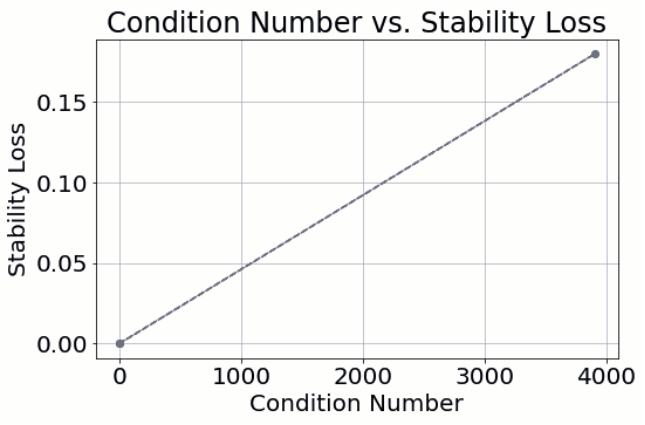


Fig. 7: Condition Number vs Stability Loss

Fig. 8: Train Vs. Test Loss Over Epochs

relationship between the condition number and stability loss, revealing a strong positive correlation. As the condition number increases, stability loss grows in a nearly linear fashion. This indicates that higher condition numbers are associated with greater instability, which is a fundamental concern in numerical computations, particularly in the context of deep learning and physics-informed neural networks (PINNs). A higher condition number suggests that the system's underlying matrix is increasingly ill-conditioned, meaning small perturbations in the input can lead to disproportionately large changes in the output. This sensitivity results in numerical instability, which manifests as an increase in stability loss. Given the steep rise observed in the plot, this suggests that poorly conditioned systems significantly degrade stability, which can impact both convergence behavior and numerical accuracy. The linearity of the observed relationship implies a direct dependence of stability loss on the condition number. This suggests that regularization techniques, preconditioning, or hierarchical compression could help mitigate stability loss by reducing the condition number. Such methods could improve the numerical properties of the system while preserving accuracy. In the context of PINNs, which often involve solving differential equations where conditioning plays a crucial role, this result reinforces the importance of maintaining a well-conditioned system. A poorly conditioned neural tangent kernel (NTK) or weight matrix could lead to unstable training dynamics, slower convergence, or inaccurate solutions. Strategies such as rank-constrained approximations, spectral normalization, or adaptive loss scaling could help counteract the instability caused by large condition numbers. The near-linear growth of stability loss suggests that condition number reduction should be a priority in designing efficient and robust computational models. This plot in Figure 8 presents the evolution of training loss and test loss over epochs, offering key insights into the generalization behavior of the model. The training loss, shown in blue, remains consistently low throughout, indicating that the model has effectively minimized the empirical risk on the training set. In contrast, the test loss, represented by the red dashed line, exhibits an initial decline followed by a plateau and then a sharp increase beyond approximately 1,200 epochs. The divergence between training and test loss after prolonged training suggests overfitting. Early in training, both losses decrease, indicating that the model is learning meaningful patterns from the data. However, as training progresses, the test loss begins to rise while the training loss remains nearly zero. This behavior implies that the model is memorizing the training data rather than learning generalizable representations, a classic symptom of overfitting.

The spike in test loss beyond 1,200 epochs suggests that the model is entering a regime where it overfits heavily, potentially exacerbated by ill-conditioning, weight saturation, or vanishing gradients. The fluctuations in test loss after this point further indicate instability, which might arise due to numerical issues or poorly conditioned optimization dynamics.

To mitigate this problem, regularization techniques such as early stopping, dropout, or weight decay could be employed. Additionally, monitoring the condition number of key matrices (e.g., neural tangent kernel or Hessian) might provide insights into whether numerical instability is contributing to the observed generalization gap. This therefore underscores the importance of balancing training duration and model complexity to ensure good generalization. The sharp contrast between near-zero training loss and rising test loss highlights the need for regularization and adaptive learning strategies to maintain stability and prevent overfitting.

VII. Conclusion

This paper introduces a novel dynamic, error-bounded hierarchical matrix (H-matrix) method for compressing and training Physics-Informed Neural Networks (PINNs). By addressing the computational and memory challenges associated with large-scale physics-based models, the proposed method significantly reduces complexity while preserving essential Neural Tangent Kernel (NTK) properties critical for stable and accurate training.

Empirical results demonstrate that the H-matrix compression leads to a substantial reduction in inference latency while maintaining high accuracy, highlighting its efficiency in real-time applications. Furthermore, analysis of the condition number's impact on stability loss reveals a near-linear relationship, emphasizing the importance of well-conditioned matrices for stable training. The observed degradation in test loss over prolonged training further underscores the need for controlled complexity, reinforcing the role of compression techniques in mitigating overfitting and improving generalization.

The proposed H-matrix approach outperforms traditional compression techniques, including Singular Value Decomposition (SVD), pruning, and quantization, by achieving superior accuracy and stability across diverse applications [32]. Its ability to dynamically adjust error bounds ensures efficient compression tailored to the data and network [31], [34], enabling robust performance even in resource-limited settings. Key benefits demonstrated include:

- Controlled error propagation, minimizing degradation in deep networks.
- Superior compression-to-accuracy trade-offs compared to baseline methods.
- Scalability for solving high-dimensional PDEs in real-world applications.

Therefore, the dynamic H-matrix method represents a significant advancement in the practical deployment of PINNs, providing a scalable, efficient, and accurate solution for complex multiscale problems. By improving inference efficiency, stabilizing training through better-conditioned representations, and mitigating overfitting, this approach bridges the gap between computational feasibility and performance, enabling PINNs to address real-world challenges in fields such as fluid dynamics, structural analysis, and climate modeling.

REFERENCES

[1] Gunnar Martinsson and Vladimir Rokhlin, *A fast direct solver for boundary integral equations in two dimensions*, Journal of Computational Physics, 205(1), 2005, 1-23.

[2] Steffen Börm, Lars Grasedyck, and Wolfgang Hackbusch, *Hierarchical Matrices*, Lecture Notes in Computational Science and Engineering, 21, 2003, 10-89.

[3] Michael Bebendorf, *Approximation of boundary element matrices*, Numerische Mathematik, 86(4), 2000, 565-589.

[4] Lars Grasedyck, *Theory and applications of hierarchical matrices*, Electron. Trans. Numer. Anal., 8, 2001, 123-134.

[5] Wolfgang Hackbusch, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices*, Computing, 62(2), 1999, 89-108.

[6] Wolfgang Hackbusch and Boris N. Khoromskij, *A sparse $\mathcal{H}$-matrix arithmetic. General complexity estimates*, Journal of Computational and Applied Mathematics, 125(1-2), 2002, 479-501.

[7] Mario Bebendorf and Serkan Rjasanow, *Adaptive low-rank approximation of collocation matrices*, Computing, 70(1), 2003, 1-24.

[8] Sergey V. Dolgov and Boris N. Khoromskij, *Tensor-product approach to global time-space-parametric discretization of chemical master equation*, SIAM Journal on Scientific Computing, 35(6), 2013, A3069-A3094.

[9] Steffen Börm, *Efficient Numerical Methods for Non-local Operators: $\mathcal{H}^2$-Matrix Compression, Algorithms and Analysis*, European Mathematical Society, 2010.

[10] Wolfgang Hackbusch, *Hierarchical Matrices: Algorithms and Analysis*, Springer Series in Computational Mathematics, 49, 2015.

[11] Michael Bebendorf and Wolfgang Hackbusch, *Existence of $\mathcal{H}$-matrix approximants to the inverse FE-matrix of elliptic operators with L^∞-coefficients*, Numerische Mathematik, 95(1), 2003, 1-28.

[12] Nick Hale and Alex Townsend, *A fast, simple, and stable Chebyshev–Legendre transform using an asymptotic formula*, SIAM Journal on Scientific Computing, 36(1), 2014, A148-A167.

[13] Lars Grasedyck and Wolfgang Hackbusch, *Construction and arithmetics of $\mathcal{H}$-matrices*, Computing, 70(4), 2003, 295-334.

[14] Miroslav Kuchta, Øystein Skotheim, and Are Magnus Bruaset, *Robust preconditioners for the boundary element method using hierarchical matrices*, Numerical Linear Algebra with Applications, 26(4), 2019.

[15] Wolfgang Hackbusch, *Hierarchical Matrices: Basic Theory and Algorithms*, Lecture Notes in Computational Science and Engineering, 21, 2002, 123-157.

[16] Lars Grasedyck, *Hierarchical Singular Value Decomposition of Tensors*, SIAM Journal on Matrix Analysis and Applications, 31(4), 2010, 2029-2054.

[17] Steffen Börm and Marco Tesei, *Approximation of Singular Integral Operators by $\mathcal{H}$-Matrices with Adaptive Cluster Bases*, SIAM Journal on Matrix Analysis and Applications, 33(2), 2012, 483-509.

[18] Serkan Gugercin and Ahmet C. Antoulas, *A survey of model reduction by balanced truncation and some new results*, International Journal of Control, 77(8), 2004, 748-766.

[19] Bart Vandereycken and Stefan Vandewalle, *A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations*, SIAM Journal on Matrix Analysis and Applications, 31(5), 2010, 2553-2579.

[20] Lars Grasedyck and Wolfgang Hackbusch, *Adaptive low-rank approximations of large-scale tensor product matrices*, SIAM Journal on Scientific Computing, 31(3), 2010, 1673-1695.

[21] Steffen Börm, Lars Grasedyck, and Wolfgang Hackbusch, *Hierarchical Matrices*, Lecture Notes in Computational Science and Engineering, 21, 2017, 10-89.

[22] Ivan Zanardi, Simone Venturi, and Marco Pansei, *Adaptive physics-informed neural operator for coarse-grained non-equilibrium flows*, Nature, Scientific Reports, 13, 2023, 15497.

[23] Shaowu Pan and Karthik Duraisamy, *Physics-Informed Probabilistic Learning of Linear Embeddings of Nonlinear Dynamics with Guaranteed Stability*, SIAM Journal of Applied Dynamical Systems, 19(1), 2020, 480-509.

[24] Nicolas Boulle, Seick Kim, Tianyi Shi, and Alex Townsend, *Learning Green's functions associated with time-dependent partial differential equations*, Journal of Machine Learning Research, 23, 2022, 1-34.

[25] Yael Azulay and Eran Treister, *Multigrid-augmented deep learning preconditioners for the Helmholtz equations*, SIAM journal of Scientific Computing, 45(3), 2022, S127-151.

[26] Xiang Xie, Wei Wang, Haijun Wu, and Mengwu Guo, *Data-driven analysis of parametrized acoustic systems in the frequency domain*, Applied Mathematical Modelling, 124, 2023, 791-805.

[27] Difeng Cai, Hua Huang, Edmond Chow, and Yunazhe Xi, *Data-driven construction of hierarchical matrices with nested bases*, SIAM Journal on Scientific Computing, 46(2), 2023, S24-S50.

[28] Xiao Liu, Jianlin Xia, Maarten V. de Hoop, and Xiaofeng Ou, *Interconnected Hierarchical Structures for Fast Direct Elliptic Solution*, Journal of Scientific Computing, 91(15), 2022, 1-31.

[29] Xuezheng Wang and Bing Dong, *Physics-informed hierarchical data-driven predictive control for building HVAC systems to achieve energy and health nexus*, Energy and Buildings, 291, 2023, 113088.

[30] Mateusz Dobija, Anna Paszyński, Carlos Uriarte, and Maciej Paszyński, *Accelerating Training of Physics-Informed Neural Network for 1D PDEs with Hierarchical Matrices*, Computational Science - ICCS 2024, Springer Nature Switzerland, Cham, 2024, 352-362.

[31] Minliang Liu, Liang Liang, and Wei Sun, *A generic physics-informed neural network-based constitutive model for soft biological tissues*, Computer Methods in Applied Mechanics and Engineering, 372, 2020, 113402.

[32] Jihun Han and Yoonsang Lee, *Hierarchical Learning to Solve PDEs Using Physics-Informed Neural Networks*, Computational Science - ICCS 2023, Springer Nature Switzerland, Cham, 2023, 548-562.

[33] Xinliang Liu, Bo Xu, Shuhao Cao, and Lei Zhang, *Mitigating spectral bias for the multiscale operator learning*, arXiv, 2024, eprint=2210.10890.

[34] R. Katende, H. Kasumba, G. Kakuba, and J. Mango, "On the Error Bounds for ReLU Neural Networks," *IAENG International Journal of Applied Mathematics*, vol. 54, no. 12, pp. 2602–2611, 2024.