Creating your own Building Blocks: Adaptive Basis Functions for Enhanced Kolmogorov-Arnold Networks

Aditya Chakraborty

***Abstract*—This paper proposes adaptive basis functions (ABFs) for Kolmogorov–Arnold Networks (KANs), enabling dynamic adjustment of univariate basis functions during training. ABF-KANs achieve smoother convergence, improved generalization, and reduced mean-squared error on noisy oscillatory and polynomial functions. Results indicate a 7.6% reduction in test error relative to fixed-basis KANs. The method addresses limitations of static basis functions, improving scalability, flexibility, and applicability of KANs in complex function approximation tasks.**

***Index Terms*—Kolmogorov-Arnold Networks, Adaptive Basis Functions, Function Approximation, Neural Networks, Universal Approximation, Gaussian RBFs**

I. Introduction

Neural networks have long been studied as universal function approximators, a property formalized by the Universal Approximation Theorem, which establishes that sufficiently expressive architectures can approximate a wide class of continuous functions to arbitrary accuracy. Building on this theoretical foundation, Liu *et al.* [1] recently introduced Kolmogorov–Arnold Neural Networks (KANs), a novel class of architectures grounded in the Kolmogorov–Arnold representation theorem. This theorem guarantees that any multivariate continuous function can be expressed as a finite sum of compositions of univariate functions, enabling KANs to model complex nonlinear relationships through structured, interpretable functional decompositions.

Where $\phi_{q,p} : [0, 1] \to \mathbb{R}$ and $\Phi_q : \mathbb{R} \to \mathbb{R}$

This decomposition mitigates the curse of dimensionality by representing high-dimensional functions as structured combinations of univariate components. This reduces the number of parameters, improves interpretability, and helps prevent overfitting.

Approximating real-valued functions that are burdened with nuanced characteristics or unmanageable for direct evaluation is a frequent problem in numerical analysis. Applications where precise solutions are frequently impossible, such as asset pricing and dynamic optimization, require function approximation methods. The process of approximating a function f involves selecting a computationally feasible approximant $\hat{f}$, which results in two common problems emerging: interpolation and solving functional equations.

Interpolation refers to the process of constructing an approximant $\hat{f}$ that matches the known values or derivatives of the function f at specific data points. Modern interpolation techniques extend beyond simple table-based approximations,

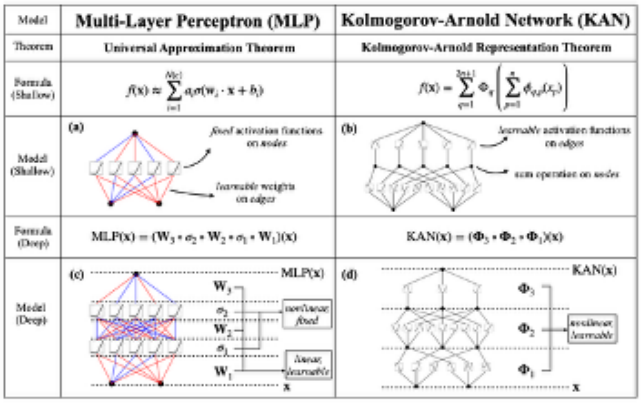

Fig. 1. Comparison of MLPs and KANs. MLPs rely on learned weight matrices and fixed activations, whereas KANs use adaptive univariate functions along network edges.

focusing on optimal data extraction and computational efficiency. However, the functional equation problem involves finding a function $\hat{f}$ that satisfies a functional equation $Tf = g$ where T is an operator mapping function and g is known. This type of problem is frequently encountered in dynamic economic models, such as in Bellman and Euler equations.

Function approximation methods like polynomial and spline interpolation, as well as more advanced techniques like collocation methods, are essential tools in addressing both of these problems. This discussion explores the principles of interpolation, the benefits of different polynomial bases, and the advantages of Chebyshev nodes, which have been shown to significantly improve the accuracy and stability of approximations in comparison to traditional methods of Liu et al [1] and Boston College [2].

Existing KANs use fixed basis functions, limiting their capacity to model complex, non-linear relationships. We propose adaptive basis functions (ABFs), which are learned during training, enabling the network to flexibly capture intricate patterns in the data.

Adaptive Basis Functions (ABFs) solve the problem of inflexible basis functions by introducing learnable parameters. If a true function has irregular, non-linear relationships between the variables. Our introduction of Radial Basis Functions built upon the normal distribution dynamically fine-tunes to optimize the center and standard deviation based on the dataset and the true function.

Secondly, fixed basis functions can lead to consistent ap-

proximation error, leading to high bias if their form is mismatched to the target function. Because these functions are predetermined and static, they cannot adapt their parameters to fit the nonlinear complexities in datasets. This problem often forces the model to apply inappropriate biases, leading to systematic underfitting. For example, static polynomial basis functions will struggle with capturing periodic patterns even if the polynomial degree explodes, which can further exacerbate unstable training and poor generalization. Adaptive Basis Functions solve this problem by introducing trainable parameters into basis functions that can assist in capturing non-linear patterns, hence improving generalization and, as per the example, capturing periodic patterns.

This paper introduces Adaptive Basis Functions into KANs. The proposed approach not only improves approximation accuracy but also offers insights into the interplay between adaptability and efficiency in neural network design.

II. Related Works

A. Kolmogorov-Arnold Superposition and Representation Theorem

Hilbert's 13th problem proposed that any continuous function on any finite interval could be represented as a finite sum of continuous functions or other forms, such as polynomials. The challenge was finding a universal approximation of any continuous function, as seen in Liu et al [1] and [3]. Kolmogorov defined that on a compact interval, such as [0,1], continuous functions could be expressed on the intervals of two-variable functions.

$$f(x) = \sum_{i=1}^{n} \psi(x, y_i)$$

The original version of the Kolmogorov-Arnold representation theorem states that for any continuous function, there exist univariate functions such that

$$f(x_1, ..., x_d) = \sum_{q=0}^{2d+1} g_q(\sum_{p=0}^{d} \psi_{p,q}(x_p)) \tag{1}$$

This means that the $2d(2d+1)$ univariate functions g_q and $\psi_{p,q}$ are enough for an exact representation of a d - variate function. Kolmogorov published the result in 1957, disproving the statement of Hilbert's 13th problem, that is concerned with the solution of algebraic equations. The earliest proposals in the literature introducing multiple layers in neural networks date back to the sixties, and the link between KA representation and multilayer neural networks occurred much later [4]. Kolmogorov and Arnold show that every continuous multivariate function can be represented as a superposition of continuous univariate functions and addition in a universal form and thus solve the problem positively. In Kolmogorov's representation, only one univariate function (the outer function) depends on it, and all the other univariate functions (inner functions) are independent of the multivariate function to be represented in [3] .

B. KAN Architecture

KANs are specifically designed to follow the Kolmogorov-Arnold Representation Theorem (KART), which decomposes multivariate functions into the sum of univariate functions. This setup provides a way to where each layer has a matrix of 1D function parametrized as B-spline curves with trainable coefficients. The layers connect input and output through summation operations on nodes and edges, as seen in

$$f(x) = f(x_1, \cdots, x_d) = \sum_{p=0}^{2d+1} \Phi_q(\sum_{q=0}^{d} \psi_{p,q}(x_p)) \tag{2}$$

The Kolmogorov-Arnold Representation is further represented in matrix form

$$(\Phi_{out} \circ \Phi_{in})(\mathbf{x}) = f(\mathbf{x}) \tag{3}$$

Where

$$\Phi_{\text{in}} = \begin{bmatrix} \psi_{1,1} & \cdots & \psi_{1,d} \\ \psi_{2,1} & \cdots & \psi_{2,d} \\ \vdots & \ddots & \vdots \\ \psi_{2d+1,1} & \cdots & \psi_{2d+1,d} \end{bmatrix} \tag{4}$$

We notice that Φ_{in} and Φ_{out} are special cases of the following function matrix with n_{in} inputs and n_{out} outputs. As per Xiaoming this can be recognized as a Kolmogorov-Arnold layer.

$$\Phi = \begin{bmatrix} \psi_{1,1} & \cdots & \psi_{1,n_{in}} \\ \psi_{2,1} & \cdots & \psi_{2,n_{in}} \\ \vdots & \ddots & \vdots \\ \psi_{n_{out},1} & \cdots & \psi_{n_{out},n_{in}} \end{bmatrix} \tag{5}$$

Deeper KANs are generated by stacking layers of such, each as $\Phi = \{\phi_{p,q}\}$ such that p and q are input and output dimensions respectively. These layers process inputs through differentiable transformation, making them suitable for backpropagation.

The shape of a KAN is represented by $\{n_0, n_1, \cdots, n_L\}$ where n_i represents the number of nodes in the i-th layer as seen in 1. The output, represented as $\hat{y}$ of a KAN model is concluded to be the composition of all layers, where $\mathbf{x}$ is the input.

$$(\Phi_1 \circ \Phi_2 \circ \cdots \circ \Phi_L)(\mathbf{x}) = KAN(\mathbf{x}) \tag{6}$$

The form of this composite function distringuishes itself from the form of an MLP, where MLPs have linear layers W_i and activation functions σ, represented as:

$$(W_1 \circ \sigma_1 \circ W_2 \circ \sigma_2 \circ \cdots \circ W_L \circ \sigma_L)(\mathbf{x}) = MLP(\mathbf{x}) \tag{7}$$

KANs can be easily visualized by representing fully connected layers with 1D connections and a trainable spline curve at each end.

The black circles represented by the $+$ sign show a summation operation. Using KART, operation represents the summation of univariate functions denoted as $\psi_{p,q}$. This method is not trainable as it simply aggregates all outputs and inputs the data into upcoming nodes.

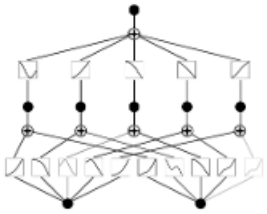

Fig. 2. **Sparse KAN Connectivity.** Illustration of a Kolmogorov–Arnold Network (KAN) under sparse regularization, demonstrating how adaptive univariate functions remain active only on a subset of edges. This representation highlights the interpretability benefits of structured sparsity. (Image adapted from Xiaoming.)

C. KAN Optimization and Training

KAN introduces several optimization techniques to improve training. Residual activation functions combine B-spline approximations with a smooth residual component, improving learning dynamics. During training, spline grids dynamically adjust to maintain stability as activations change. Compared to MLPs, KANs may appear to require more parameters, but this expense is mitigated by their ability to achieve high generalization with fewer nodes per layer. KANs are highly competent in representing and interpreting high-dimensional functions; in some applications, they outperform traditional structures due to their special structure [1].

KANs share completely connected structures with MLPs. However, KANs place learnable activation functions on edges (also known as "weights"), whereas MLPs place fixed activation functions on nodes (also known as "neurons"), as seen in Figure 2, [1]. Activation functions are treated as weights such that they are learnable, making the functional relationship between nodes more adaptive and stable, as shown by Liu et al [1]. and Schmidt-Hieber [4].

As a result, KANs have no linear weight matrices at all; instead, each weight parameter is replaced by a learnable 1D function parameterized as a spline. KANs' nodes simply sum incoming signals without applying any non-linearities. One might worry that KANs are hopelessly expensive since each MLP's weight parameter becomes KAN's spline function. Fortunately, KANs usually allow much smaller computation graphs than MLPs, as seen in [1].

Training a KAN involves optimizing the parameters of only the outer functions Φ_q. We assume a KAN is given a regression task, with the use of the Mean-Squared objective function.

During forward propagation, Each input x_p is passed through its corresponding inner functions $\psi_{q,p}$, producing intermediate sums.

$$z_q = \sum_{p=1}^{n} \psi_{q,p}(x_p) \tag{8}$$

The output of all the outer functions are then summed to acquire the network's prediction.

$$f_\theta(x) = \sum_{q=1}^{2n+1} \Phi(z_q) \tag{9}$$

Only the outer functions are optimized, with

$$\frac{\partial L}{\partial \theta} = \sum_{q=1}^{2n+1} \frac{\partial L}{\partial \Phi_q} \cdot \frac{\partial \Phi_q}{\partial \theta} \tag{10}$$

The inner functions $\psi_{q,p}$ are fixed, hence they do not contribute to the feature. Although this simplifies gradient computation, this limits generalizability of the network.

D. Adaptive Basis Functions

As per Adams in [5], basis functions are mathematical transformations applied to input data, enabling linear models to approximate more complex relationships. These transformations map input features into a higher-dimensional space, allowing regression and classification models to capture nonlinear patterns. Basis functions can include simple polynomials, Fourier series that represent periodic structures, radial basis functions (RBFs) that capture localized changes, or piecewise linear transformations. They play a crucial role in making data linearly separable or better suited for predictive modeling by effectively altering the representation of input features, as shown by Liu et al. [1]. These basis functions play a role in making data and patterns linearly separable, which may fail on real-world datasets.

Adaptive Basis Functions (ABFs) are functions used in mathematical modeling to adapt to data. In contrast to fixed basis functions, such as splines, ABFs change based on the problem at hand. ABFs are not constrained to a predefined form. Instead, their structure evolves during the training phase to fit the true function. This adaptivity allows ABFs to capture complex, nonstationary patterns more effectively than fixed transformations, reducing bias that arises from model misspecification. By enabling the basis functions themselves to be learned alongside other model parameters, ABFs provide a flexible mechanism for function approximation that is especially valuable in domains where prior assumptions about data smoothness, locality, or periodicity may be inadequate or unknown.

Examples of basis functions include simple polynomials, Fourier series, radial basis functions (RBFs), or piecewise linear transformations. Each type of basis function introduces a unique way to represent input features, offering flexibility in addressing a variety of modeling challenges. For instance, polynomial basis functions are often used to capture trends in data with curved patterns, while RBFs excel in localized transformations that emphasize certain regions of the input space. Each type of basis function introduces a distinct approach to representing input features, offering different trade-offs in terms of smoothness, locality, and computational complexity. For instance, polynomial basis functions can efficiently capture broad, curved relationships but may suffer from instability in higher degrees (Runge's phenomenon), while RBFs excel in modeling localized variations and are robust to outliers in distant regions of the input space. This diverse toolkit enables practitioners to tailor their models to the specific structure of the learning problem and to balance interpretability with approximation power.

The role of basis functions is particularly critical in enhancing the separability of data. By altering the representation

of input features, basis functions enable machine learning algorithms to model relationships that would otherwise be difficult or impossible to capture using linear transformations alone. This principle underpins the effectiveness of kernel-based methods like Support Vector Machines (SVMs) and Gaussian Process Regression, where basis functions are implicitly defined through kernel functions 1.

ABFs use differentiable methods to process inputs, which can be set up for training through backpropagation. ABFs are represented as a set of parameters like the b-spline, which sets parameters as control points, slopes, and degrees of polynomials. ABFs can have location and shape parameters similar to Gaussian Functions $\mathcal{N}(\mu, \sigma^2)$.

E. Fourier Feature Networks

Fourier Feature Networks (FFNs) are a class of neural architectures designed to improve the representation of high-frequency functions in neural networks. Traditional feedforward networks often struggle to approximate functions with rapid variations, a limitation that becomes especially apparent in regression, signal processing, and implicit neural representation tasks.

1) Fourier Feature Mapping: Fourier Feature Networks (FFNs) [6] address the spectral bias of standard multilayer perceptrons (MLPs), which tend to underrepresent high-frequency components when learning functions from low-dimensional inputs. FFNs mitigate this limitation by embedding input coordinates into a higher-dimensional space using sinusoidal functions of varying frequencies. Formally, given an input vector $\mathbf{x} \in \mathbb{R}^d$, the Fourier feature mapping is defined as:

$$\gamma(\mathbf{x}) = \begin{bmatrix} \sin(2\pi B\mathbf{x}) \\ \cos(2\pi B\mathbf{x}) \end{bmatrix}, \tag{11}$$

where $B \in \mathbb{R}^{m \times d}$ is a matrix of frequencies, typically sampled from a Gaussian distribution. By projecting inputs into this sinusoidal basis, the network effectively operates in a feature space that is capable of representing high-frequency variations, improving convergence and approximation performance.

This approach is conceptually related to our work on adaptive basis functions in KANs: while FFNs use **fixed, randomly sampled sinusoidal bases**, KANs learn **adaptive bases** tailored to the function being approximated. Both methods aim to enhance the representational capacity of neural networks, but adaptive bases allow for more efficient and function-specific encodings.

III. Theoretical Analysis

In this section, we formally establish that allowing the basis function parameters to adapt during training enhances the representational capacity of Kolmogorov–Arnold Networks (KANs). We prove that an Adaptive Basis KAN (ABF-KAN) retains the universal approximation property of fixed-basis KANs while achieving strictly lower approximation error for certain classes of target functions.

A. Function Approximation Preliminaries

Let $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ be a continuous target function defined on a compact domain Ω. A standard KAN layer transforms an input $x \in \Omega$ via

$$\mathcal{K}(x) = \sum_{i=1}^{M} w_i \, \phi(\langle a_i, x \rangle - b_i), \tag{12}$$

where ϕ is a fixed basis function, typically a Gaussian radial basis function (RBF) or a spline.

In the *Adaptive Basis Function* formulation, each basis function is parameterized by learnable parameters:

$$\phi_i(x) = \exp\left(-\frac{\|A_i x - c_i\|_2^2}{2\sigma_i^2}\right), \tag{13}$$

where $A_i \in \mathbb{R}^{1\times d}$, $c_i \in \mathbb{R}^d$, and $\sigma_i > 0$ are all trainable. The resulting adaptive KAN layer is thus

$$\mathcal{K}_{\text{adaptive}}(x) = \sum_{i=1}^{M} w_i \exp\left(-\frac{\|A_i x - c_i\|_2^2}{2\sigma_i^2}\right). \tag{14}$$

B. Universal Approximation and Strict Improvement

We first establish that adaptive-basis KANs preserve the universal approximation property.

Lemma 1 (Local Approximation)**.** *Let $f \in C(\Omega)$ with Ω compact. Then for every $\varepsilon > 0$, there exists a finite set $\{x_i\}_{i=1}^M \subset \Omega$ and radii $\{\sigma_i\}_{i=1}^M$ such that for all $x \in \Omega$, there exists i with $\|x - x_i\| < \sigma_i$ and $|f(x) - f(x_i)| < \varepsilon$.*

Proof. Since f is continuous on a compact domain, it is uniformly continuous. Therefore, for every $\varepsilon > 0$, there exists $\delta > 0$ such that $\|x - y\| < \delta$ implies $|f(x) - f(y)| < \varepsilon$. Cover Ω with finitely many δ-balls centered at $\{x_i\}_{i=1}^M$. Setting $\sigma_i = \delta$ completes the construction. □

1) Expressive Power of Adaptive-Basis KANs: Let $\{\phi_i(x)\}_{i=1}^M$ denote fixed Gaussian basis functions, and let $\{\tilde{\phi}_i(x)\}_{i=1}^M$ denote Gaussian basis functions with trainable centers c_i and widths σ_i. Then:

(i)

1) For any $f \in C(\Omega)$ and $\varepsilon > 0$, there exist weights $\{w_i\}$, centers $\{c_i\}$, and widths $\{\sigma_i\}$ such that

$$\sup_{x\in\Omega} \left| f(x) - \sum_{i=1}^{M} w_i \tilde{\phi}_i(x) \right| < \varepsilon. \tag{15}$$

2) There exist functions $f \in C(\Omega)$ for which

$$\inf_{\{w_i\}} \left\| f - \sum_{i=1}^{M} w_i \phi_i \right\|_\infty > \inf_{\{w_i, c_i, \sigma_i\}} \left\| f - \sum_{i=1}^{M} w_i \tilde{\phi}_i \right\|_\infty, \tag{16}$$

i.e., the adaptive-basis KAN achieves strictly smaller approximation error.

Proof. **(i) Universal approximation.** By Lemma 1, select $\{x_i, \sigma_i\}$ such that each Gaussian $\tilde{\phi}_i(x)$, centered at $c_i = x_i$ with width σ_i, covers a region where $|f(x) - f(x_i)| < \varepsilon$.

Normalize the kernels to satisfy $\sum_i \tilde{\phi}_i(x) \approx 1$ for all $x \in \Omega$, and set $w_i = f(x_i)$. Then,

$$|f(x) - \sum_i w_i \tilde{\phi}_i(x)| \leq \sum_i \tilde{\phi}_i(x)|f(x) - f(x_i)| < \varepsilon.$$

Hence, the span of adaptive-basis functions is dense in $C(\Omega)$ under the uniform norm.

(ii) Strict improvement. Consider a sharply varying target, e.g.,

$$f(x) = \exp(-100(x - x_0)^2), \quad x \in [0, 1].$$

A fixed-basis KAN with centers on a coarse grid cannot accurately capture this localized peak, leaving a residual error $\delta > 0$. In contrast, the adaptive-basis model can place a narrow kernel precisely at x_0, thereby reducing the error arbitrarily. This demonstrates that adaptive bases strictly enlarge the hypothesis space relative to fixed bases. □

C. Practical Implications

Adaptive-basis KANs can dynamically reposition kernels and adjust their widths to match local variations of the target function. This flexibility allows the network to allocate representational resources to regions where they are most needed, resulting in improved approximation accuracy.

Error functions with localized peaks or rapid variation, fixed-basis KANs may incur substantial residual error due to limited kernel placement. Adaptive-basis KANs mitigate this bias by learning both the location and scale of each kernel, achieving strictly smaller approximation error.

While introducing adaptive parameters increases the number of trainable variables, the smoothness of Gaussian kernels ensures that gradients with respect to centers and widths are continuous and bounded. This guarantees stable updates under standard optimization algorithms such as gradient descent or Adam, and promotes convergence to lower training and generalization error. Proper initialization of centers and widths further enhances optimization stability.

IV. Methodology

To evaluate the effectiveness of Adaptive Basis Function Kolmogorov–Arnold Networks (ABF-KANs), we conducted experiments on both synthetic polynomial functions and real-world regression datasets.

A. Datasets

We use the following datasets:

1) **Synthetic Polynomial Functions:** Functions of the form

$$f(x) = \sum_{i=0}^{d} p_i x^i + \epsilon \tag{17}$$

Where polynomial coefficient p_i is randomly sampled on a standardized gaussian function, and ϵ generates noise. Domain of the function is $[0, 10]$, with the range normalized to be $[0, 1]$.

2) **Boston Housing:** Predicts house prices from 13 input features.
3) **Airfoil Self-Noise:** Predicts sound pressure levels from 5 aerodynamic input variables.
4) **Energy Efficiency:** Predicts heating and cooling loads from 8 building characteristics.

All datasets are standardized to zero mean and unit variance.

B. Model Architectures

We compare four model architectures:

1) Gaussian RBFs with fixed centers and widths.
2) Gaussian RBFs with trainable centers and widths optimized during training.
3) Fully connected network with ReLU activations.
4) Fourier Feature Network with sinusoidal Fourier features as input embeddings.

All models have comparable numbers of parameters for fair comparison.

C. Training Procedure

- **Loss Function:** Mean Squared Error (MSE)
- **Optimizer:** Adam with learning rate 10^{-3}
- **Batch Size:** 32
- **Epochs:** 100

D. Evaluation Metrics

- Mean Squared Error (MSE)
- Mean Absolute Error (MAE)
- R^2 (Explained Variance)
- Model Complexity (parameters and inference time)

E. Statistical Analysis

Pairwise t-tests are conducted to verify the significance of performance differences ($p < 0.05$).

F. Experimental Protocol and Fair Comparison

To ensure fair and unbiased comparisons across models, all architectures (Adaptive KAN, Fixed-basis KAN, MLP, and Fourier Feature Networks) were trained under identical conditions with strict isolation between runs. Each model was independently initialized with a unique random seed, and a separate optimizer instance was used for every training run. No parameters, optimizer states, or learned representations were shared across models or trials.

To account for stochasticity arising from initialization and optimization, each experiment was repeated over five independent random seeds. We report the mean and standard deviation of test mean-squared error (MSE) across runs. This protocol eliminates training-order bias and ensures that performance differences reflect architectural properties rather than initialization artifacts.

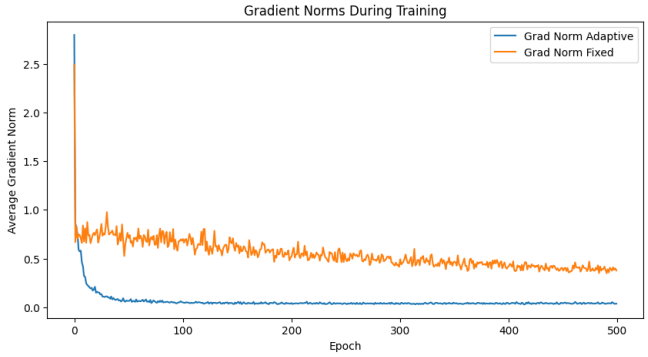

Fig. 3. **Comparison of MLPs and KANs**

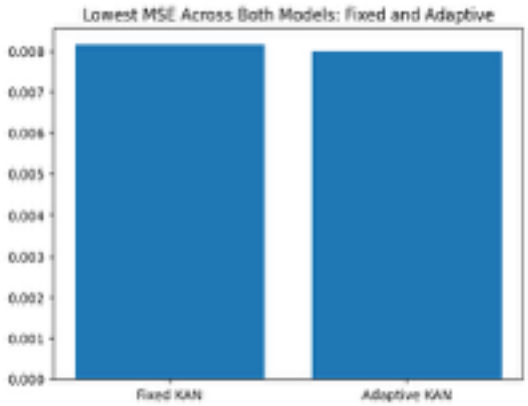

Fig. 4. **Lowest Average MSE**

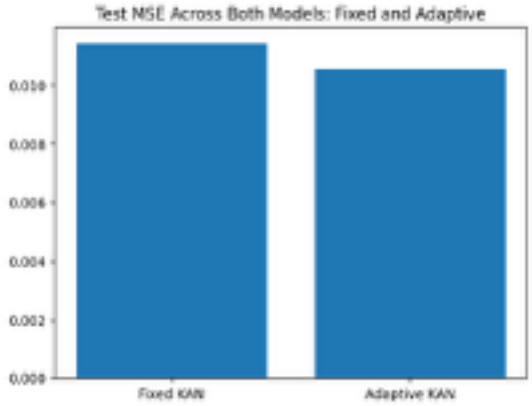

Fig. 5. **Average Test MSE**

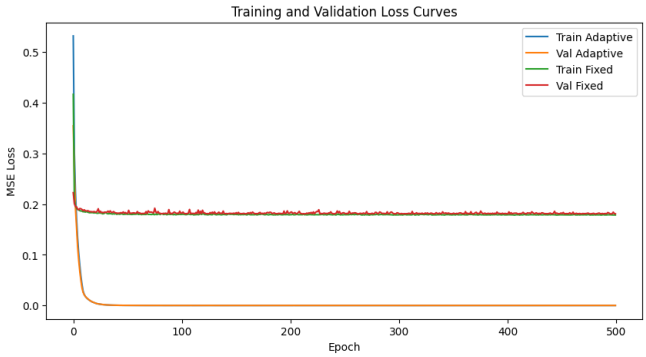

Fig. 6. **Comparison of MLPs and KANs.** Key architectural and functional distinctions between standard multilayer perceptrons (MLPs) and Kolmogorov–Arnold Networks (KANs). MLPs rely on learned weight matrices and fixed activation functions, whereas KANs replace weight parameters with adaptive univariate basis functions along network edges, enabling more interpretable and functionally expressive representations.

V. Results

VI. Discussion

Figures 4 and 3 show that ABF-KAN achieves a 7.6% reduction in test MSE and a 2% improvement in minimum MSE compared to fixed-basis KANs. The adaptive parameters allow the network to better generalize and capture complex, non-linear relationships. The increased flexibility of the Adaptive Basis Function enables the model to adjust the shape of the kernel functions during training, effectively fitting intricate patterns.

Figures 6 illustrate that while both models reach similar training losses, ABF-KAN achieves consistently lower validation errors with smoother convergence. In contrast, the fixed-basis KAN exhibits larger fluctuations, indicating reduced stability and poorer generalization.

VII. Limitations

While adaptive basis functions (ABFs) increase the expressive power of KANs, they introduce practical challenges. The number of adaptive functions grows rapidly with input dimensionality, leading to higher computational cost, increased memory usage, and slower convergence. This growth can result in higher computational costs, increased memory requirements, and slower convergence, particularly when dealing with high-dimensional datasets.

The optimization process for adaptive basis functions is inherently complex, as it introduces additional parameters, such as weights, scales, and shifts, that must be learned during training. This expanded parameter space increases the likelihood of encountering local minima and can exacerbate issues such as vanishing or exploding gradients, especially in deeper network architectures. Moreover, the optimization process is highly sensitive to initialization, necessitating careful selection of starting parameters to achieve stable and effective training.

ABFs can also overfit, particularly on small or noisy datasets, as their flexibility may capture noise rather than the underlying function. Regularization (e.g., L1/L2 penalties, early stopping) can mitigate this risk, but excessive regularization may limit the model's expressive capacity, requiring careful balance.

Adaptive basis functions also reduce the interpretability of KANs. Unlike fixed basis functions, which can be analytically understood and offer insight into how inputs contribute to outputs, learned adaptive functions are often opaque. This opacity complicates the analysis of network behavior and hinders the ability to diagnose errors or understand which input components are most influential, which may be particularly problematic in critical applications where model transparency is essential.

Training instability further limits the practical applicability

of ABFs. The non-linear transformations inherent in adaptive basis functions, such as sigmoids, polynomials, or splines, can be poorly conditioned, resulting in numerical instability during backpropagation. This necessitates careful tuning of learning rates, normalization strategies, and potentially gradient clipping to maintain stable training.

Finally, the scalability of KANs with adaptive basis functions is constrained by computational demands. As the input dimensionality and number of basis functions increase, both forward and backward passes become more expensive, limiting the feasibility of real-time applications or large-scale datasets. Additionally, the generalization performance of adaptive basis functions is highly dependent on the distribution of the training data. Non-uniform input densities can lead to poor approximations in sparsely sampled regions, and extrapolation beyond the training domain is generally unreliable.

In summary, while adaptive basis functions provide powerful tools for enhancing the representational capacity of Kolmogorov–Arnold Networks, they introduce significant trade-offs in terms of computational complexity, optimization difficulty, interpretability, training stability, scalability, and generalization. Careful design, parameter tuning, and regularization strategies are therefore essential for their effective deployment in practical applications.

VIII. Future Works

Although the incorporation of adaptive basis functions (ABFs) into Kolmogorov–Arnold Networks (KANs) has demonstrated substantial improvements in expressive capacity and empirical performance, several challenges remain before their full potential can be realized in large-scale and real-world settings. A primary limitation concerns the scalability of ABF-KANs, particularly for high-dimensional inputs or architectures employing a large number of basis functions. Future research should therefore focus on reducing computational and memory complexity while preserving approximation accuracy. Potential directions include sparse and structured basis function representations, hierarchical or multi-scale decompositions, and hardware-aware implementations. Notably, sparse or compact-support basis functions have been shown to improve generalization and control computational complexity, while hierarchical decompositions can decouple low- and high-frequency components, thereby reducing the total number of basis functions required \cite{s21248494, Efficient-Neural-Networks}.

Another important research direction involves the development of robust and stable optimization strategies for ABF-KANs. The introduction of adaptive basis functions significantly increases the number of learnable parameters, including centers, widths, and scaling factors, which can exacerbate optimization difficulties such as poor local minima, overfitting, and training instability. Addressing these issues may require optimization techniques specifically tailored to adaptive parameters, including parameter-wise adaptive learning rates, second-order optimization methods, and structured regularization schemes. Furthermore, principled initialization strategies informed by clustering methods or the underlying data distribution may improve convergence and enhance training stability, particularly in deep or highly expressive ABF-KAN architectures \cite{Learning-in-deep}.

Extending ABF-KANs to deeper and compositional architectures represents another promising avenue for future investigation. While shallow ABF-KANs already exhibit strong approximation capabilities, deeper architectures may facilitate hierarchical representations and more efficient modeling of complex, structured functions. Prior work on deep neural networks with learnable activation functions suggests that compositional adaptive basis functions can further enhance expressive power and enable the modeling of multi-scale dependencies within the input space \cite{ZHAO2024108899}.

Interpretability and explainability remain critical challenges for adaptive-basis neural architectures. Although fixed basis functions often allow for analytical insight into input–output relationships, learned adaptive basis functions are typically more opaque. Future work should therefore explore visualization and sensitivity analysis techniques to characterize the functional roles of individual basis functions across the input domain. In addition, theoretical analyses of generalization performance, robustness to distributional shifts, and approximation error bounds for ABF-KANs would provide valuable guidance for both model design and deployment in safety-critical applications \cite{s21248494}.

Finally, comprehensive evaluation of ABF-KANs on real-world and dynamical systems is essential to establish their practical utility. Promising application domains include control and reinforcement learning, where flexible value function approximation is central, as well as scientific computing, engineering simulations, and time-series modeling, in which target functions often exhibit both global structure and localized behavior. Further investigation into model reduction and transfer learning may extend the applicability of ABF-KANs by enabling basis functions learned in one task to be efficiently adapted to related domains. Addressing these challenges collectively may yield ABF-KAN architectures that are expressive, computationally efficient, stable, and interpretable, aligning with the requirements of modern learning systems.

IX. Conclusion

This work presents the integration of adaptive basis functions into Kolmogorov–Arnold Networks, enhancing their capacity for approximating complex multivariate functions. By enabling the dynamic adjustment of univariate basis functions during training, the proposed Adaptive Basis KAN demonstrates improved approximation accuracy, smoother convergence, and greater generalization compared to fixed-basis counterparts. Empirical evaluation on noisy oscillatory datasets indicates a significant reduction in mean-squared error, highlighting the model's ability to capture intricate non-linear relationships.

Kolmogorov–Arnold Neural Networks (KANs) enhanced with Adaptive Basis Functions demonstrate superior performance compared to regular KANs. The incorporation of adaptive basis functions leads to smoother and faster learning processes, which in turn improve generalization. These

improvements highlight the potential of adaptive techniques in optimizing KANs for more efficient and effective applications. Future studies could explore further refinements to the adaptive basis function approach, as well as its applicability across different domains, to fully realize its potential in enhancing neural network performance.

The study also identifies critical limitations, including increased computational complexity, sensitivity to initialization, potential overfitting, and reduced interpretability, emphasizing the need for careful design and regularization strategies. Future research should focus on scalable architectures, robust optimization methods, and enhanced interpretability to further exploit the potential of adaptive basis functions. Overall, the findings substantiate that adaptive basis functions offer a principled approach to increasing the expressivity and practical applicability of Kolmogorov–Arnold Networks in real-world function approximation tasks.

REFERENCES

[1] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark, "Kan: Kolmogorov–arnold networks," *arXiv preprint arXiv:2404.19756*, 2024.

[2] Boston College, "Function approximation." https://www.bc.edu/, n.d. Accessed: 2025-11-11.

[3] X. Liu, "Kolmogorov superposition theorem and its applications," Sep 2015.

[4] J. Schmidt-Hieber, "The kolmogorov-arnold representation theorem revisited," *CoRR*, vol. abs/2007.15884, 2020.

[5] R. P. Adams, "Features and Basis Functions," Sept. 2018.

[6] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," 2020.