Informaticae Abstracta. ISSN: 3061-8533
Año 2025, Vol. 3, No. 2, pp. 4-24
JUL | DIC 2025
https://doi.org/10.36677/ia.v3n2.26890

Empirical Characterization of Lipschitz Continuity in k-nets with Precomputed Inner Functions

Huerta Mendoza, U. B.

Información del Artículo

Recibido: 5 de septiembre, 2025
Aceptado: 4 de noviembre, 2025

Palabras clave: Kolmogorov-Arnold networks, Lipschitz continuity, neural network regularity, function approximation, adversarial robustness

Resumen

The application of the Kolmogorov-Arnold representation theorem (KRT) in neural networks has historically been hindered by the non-smooth nature of its inner functions (ψ). While this has led to criticism of the theorem's relevance, the work of Sprecher and later Actor (2018) provided a constructive path forward by developing an algorithm for Lipschitz-continuous universal ψ. However, it remained an open question whether a network built upon this base could preserve that regularity after training.

1. Introduction

The value of the Kolmogorov-Arnold representation theorem (KRT) as basis for learning architectures has been a point of contention for decades. Several critical and survey works have situated KRT's influence as mostly philosophical or inspirational for modern machine learning, warning against misinterpreting it as a direct foundation for universal function approximation in neural networks [1, 2, 3, 4]. The main argument used to hinder KRT applicability is the often-found lack of "smoothness" of the inner ψ function. This is, the inner functions described in KRT can be highly non-smooth, even fractal [5], resulting in learned mappings that are not easily computationally tractable.

At the same time, Neural Networks (NNs) have become the preferred tool to characterize and analyze complex relationships in data, and the foundation of deep learning itself [6]. They've been successfully applied across essentially all scientific domains; a success built upon the well-established Multilayer Perceptron (MLP) architecture. The effectiveness of the MLP's brute force approach to approximating optimal functions for virtually any data source was mathematically formalized by the Universal Approximation Theorem, with no theoretical bounds beyond having a "sufficient" number of layers [7]. This number must be determined empirically for every application; a trial-and-error approach that extends to the selection of the MLP's activation functions.

Nonetheless, the widespread success of deep learning is often shadowed by the "black box"nature of MLPs, that difficult inference tracking due to its layer composition and structure [8]. Additionally, MLPs, like KRT representations, suffer from highly non-smooth outputs, an expected consequence of the universality of its approximation capabilities. This is reflected in high output sensitivity to changes in the inputs of trained models. Small changes in inputs of NNs with the intention of influencing output behavior are known as adversarial perturbations, or adversarial attacks [9].

Both paradigms, universal approximation and universal representation, are expected to deliver robust outputs under such attacks, since they represent a serious concern in the deployment of deep learning models in safety-critical systems such as autonomous driving [10].

One of the preferred methods to assess the robustness of NNs is the Lipschitz constant, defined as the maximum gradient of the function with respect to the distance metric [11]. Lipschitz-constrained neural networks limit the maximum deviation of the learned mapping in response to a change of the input, often at the expense of expressivity [12, 13]. An array of techniques are widely used to enforce this property, ranging from loss-based penalization to architectural constraints, such as 1-Lipschitz layers [14].

Despite skepticism, the applicability of the theorem was significantly advanced by the work of Sprecher [15], showing that the representation could be simplified such that distinct inner functions ψ_{qp} could be replaced by scaled and translated instances of a single, universal inner function ψ. This simplification makes the construction amenable to neural network development. Another notable step toward realizing this was the work of [16], who proposed an algorithm to construct a Lipschitz continuous ψ function via arc-length reparameterization. However, Actor explicitly left the construction of the outer function Φ as future work.

This work presents a Kolmogorov-Arnold neural network (*k-net*) [5] based on the Sprecher-Lorentz reformulation of the KRT that incorporates the precomputed, high-fidelity ψ function from Actor's algorithm parametrized by a Λ coefficient matrix. Throughout this work, we quantify the degree to which the regularity of Actor's precomputed inner function manifests as regularity of a trained network. Also, properties of the learned mapping, as well as standard predictive performance metrics, including its empirical Lipschitz constant, the smoothness of its internal representations, adversarial robustness, and gradient stability are evaluated. We also dissect the roles of ψ and its learnable coefficients Λ through an ablation study.

This work makes several contributions. First, Actor's work is extended by providing the first implementation that take advantage of his Lipschitz-continuous inner function. Second, a suite of internal regularity metrics designed to characterize the preservation of mathematical guarantees of the proposed *k-net* are introduced. Third, the exploration of the Λ coefficients as a viable path towards leveraging theoretical guarantees of the KRT is presented. Finally, a performance-regularity space demonstrates that the proposed architecture occupies a comparable position with MLPs.

This work is positioned as complementary to recent efforts in KRT based neural networks. While Liu et al. [8] focused on learnable B-spline edges for interpretability and Solis-Winkler et al. [5] defined *k-net* emphasizing algebraic symmetry through dual numbers, our work addresses the orthogonal question of whether and how theoretical regularity guarantees can be preserved and leveraged in implementations based on the theorem. The work built upon the rich mathematical foundations established by these and earlier works while pursuing the characterization of regularity in KRT-driven architectures.

This work presents the following structure: Section 2 details the mathematical background of the KRT, Actor's construction, and positions the work within the literature. Section 3 presents the model architecture. Section 4 describes the experimental design, including ablation variants and regularity metrics. Section 5 presents results. Section 6 discusses impli-

cations, and Section 7 concludes with directions for future work.

2. Background and Related Work

The Kolmogorov-Arnold Representation Theorem

The Kolmogorov-Arnold Representation Theorem (KRT), states that any multivariate continuous function can be represented as a finite sum of compositions of continuous univariate functions [17]. Formally, for any continuous function $f \in C([0,1]^n)$ mapping $\mathbb{R}^n \to \mathbb{R}$, there exist continuous univariate functions $\chi_q : \mathbb{I} \to \mathbb{R}$ (outer functions) and $\psi_{p,q} : \mathbb{R} \to \mathbb{R}$ (inner functions), where $p = 1, \dots, n$ and $q = 0, \dots, 2n+1$, such that:

$$f(x_1, \dots, x_n) = \sum_{q=0}^{2n} \chi_q \left(\sum_{p=1}^{n} \psi_{p,q}(x_p) \right) \tag{1}$$

The inner functions $\psi_{p,q}$ are independent of the function f. This theorem proved that a composition of univariate functions suffices for high-dimensional function approximation, opposed to the previous intuition that pointed to the necessity of functions of multiple variables [17].

Given it's universal scope, the original proof constructs inner functions that are often highly irregular: Hölder continuous but not necessarily differentiable, and in some cases exhibiting fractal-like behavior [18]. This poses severe challenges to the attempts to materialize the universal representation guarantees of the theorem computationally.

The applicability of the theorem was significantly advanced by Sprecher [15] and Lorentz [19]. Sprecher showed that all $n(2n+1)$ inner functions could be replaced by shifted and scaled versions of a single universal function ψ, and Lorentz that all outer functions could be replaced by a single function Φ. The reformulation becomes:

$$f(x_1, \dots, x_n) = \sum_{q=0}^{2n} \Phi \left(\sum_{p=1}^{n} \lambda_p \psi(x_p + q\epsilon) \right) \tag{2}$$

where λ_p are rationally independent constants satisfying $\sum_p \lambda_p = 1$, and ϵ is a small shift parameter. This Sprecher-Lorentz formulation forms the foundation of the proposed implementation.

This formulation separates the representation task into three stages. First, the input vector $\mathbf{x} \in \mathbb{R}^n$ undergoes $Q = 2n+1$ shifted projections through a universal inner function ψ. Second, these projections are weighted by coefficients $\Lambda = \{\lambda_{qp}\}$ to produce intermediate features $\{Z_q\}_{q=0}^{2n}$. Third, a shared outer network Φ processes each feature independently before summation yields the final prediction.

Kolmogorov-Arnold Neural Networks (*k-nets*)

The Kolmogorov-Arnold Neural Networks (*k-nets*) are new compositional mappings derived from KRT that allow the selection of the inner layer functions for neural network construction, while maintaining rigorous adherence to the theorem [5]. They are defined as the following:

Let $f : [0,1]^n \to \mathbb{R}$ be a continuous function for which exists univariate function $\Phi : [0,1] \to \mathbb{R}$, $\phi_{qp} : [0,1] \to \mathbb{R}$ such that $f = \sum_q \Phi_q(\sum_p \phi(x_p)$, $1 \leq q \leq 2n+1$, $1 \leq p \leq n$, then a *k-net* is a composition mapping $Z = Z_0 \circ ... \circ Z_L$, where $Z_i : R^{d_i} \to R^{d_{i+1}}$, and $d_{i-1} = d_i$. The coordinates of each $Z_i = (\zeta_1, ..., \zeta_{d_i})$. The model layer structure is the vector $[d_0, ..., d_L]$ of dimensions of the domain of every Z_i for each layer.

Actor's Lipschitz Continuous Inner Functions

While the Sprecher-Lorentz reformulations streamline computing representations, they do not present a method to secure a universal inner function ψ with robustness guarantees. The original constructions, such as Köppen's refinement of Sprecher's function [20], remain Hölder continuous with unbounded slopes.

Actor [16] addressed this gap through a reparameterization based on the work by Hedberg [21] and Kahane [22]. The center of his argument is that any strictly monotonic continuous function is rectifiable (has finite arc length), and any rectifiable curve admits a Lipschitz-continuous reparameterization. Formally, if $\hat{\psi}$ is a Hölder-continuous inner function, one can construct a Lipschitz-continuous equivalent by:

$$\psi = \hat{\psi} \circ \sigma^{-1} \tag{3}$$

where $\sigma(x)$ is the normalized arc-length parameterization of $\hat{\psi}$:

$$\sigma(x) = \frac{1}{L} \sup_{P \in \mathcal{P}(x)} \sum_{i=1}^{\ell} \sqrt{(t_i - t_{i-1})^2 + (\hat{\psi}(t_i) - \hat{\psi}(t_{i-1}))^2} \tag{4}$$

where $\mathcal{P}(x)$ is the set of all partitions of $[0, x]$ and L is the total arc length over $[0, 1]$ [16].

The algorithm that computes ψ starts with Köppen's version of Sprecher's function which produces a Hölder-continuous base function $\tilde{\psi}^k$, that is refined through iterative interval subdivision. At refinement level k, the construction generates γ^k points through recursive application of a fractal-like refinement rule parameterized by the base γ and dimension n. Then, reparametrized. The resulting function is Lipschitz continuous. Figure 1 reproduces Actor's result showing the difference between the Hölder function and the Lipschitz reparameterization. The algorithm begins with Köppen's refinement of Sprecher's inner function. The the reader is referred to [16] for further details.

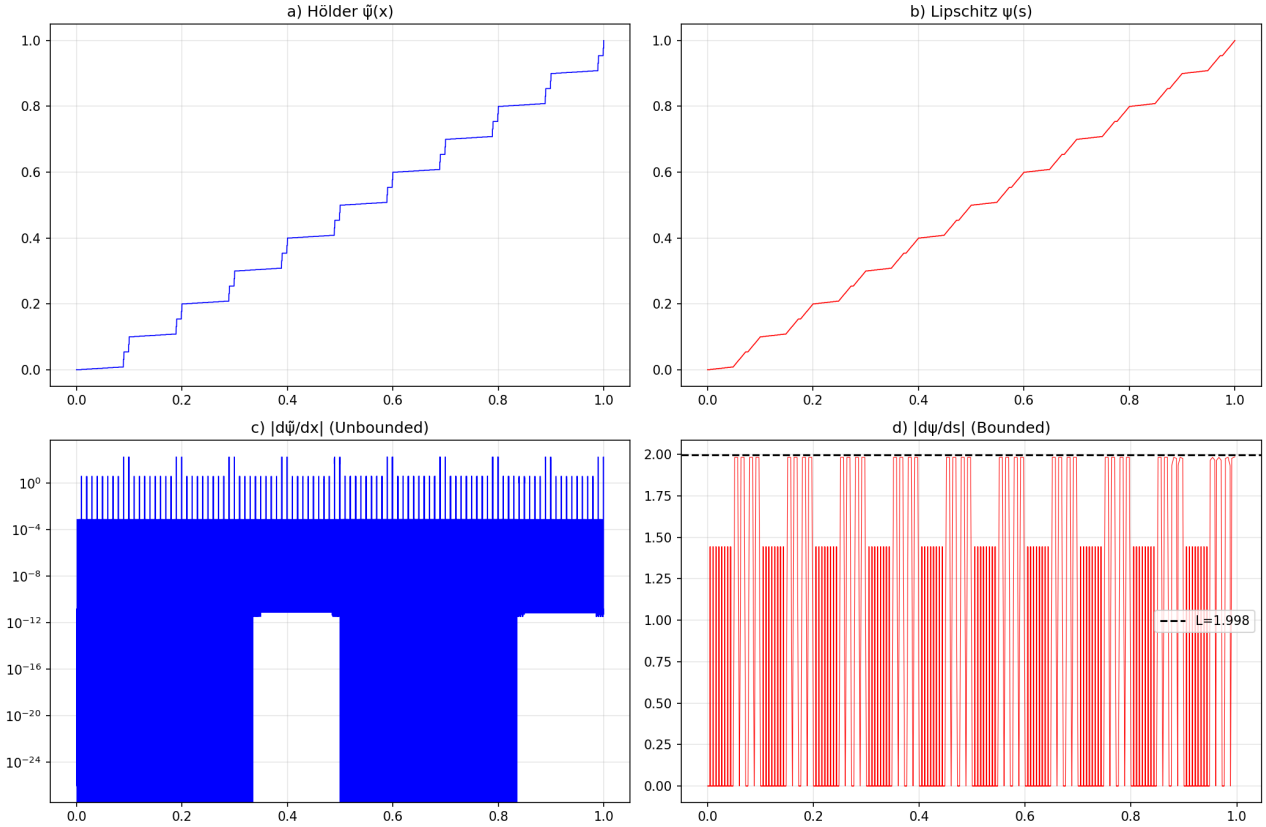

Figura 1: Comparison of inner functions ψ at refinement level $k = 6$. Arc-length reparameterization yields controlled slopes, ensuring Lipschitz continuity with constant $L_\psi \approx 1{,}998$.

Recent Implementations

The past years have seen renewed interest in Kolmogorov-Arnold inspired architectures, with distinct emphases.

KANs (Kolmogorov-Arnold Networks). Liu et al. [8] introduced an architecture where learnable univariate functions parameterized as B-splines are placed on the edges rather than nodes of the network. Their work emphasized interpretability and demonstrated strong performance on symbolic regression. However, their learnable B-spline functions lacked guarantees of regularity and is not a direct translation of the KRT.

***k-nets* with Dual Numbers.** Solis-Winkler et al. [5] recently presented the *k-net* neural network, that formalizes the training process within the algebraic framework of hyperdual numbers. One of the main contributions is demonstrating how forward-mode automatic differentiation via dual number arithmetic provides a mechanism for computing Jacobians while maintaining the KRT decomposition. Their inner functions are also B-splines, and their outer functions use simple activations (ReLU, sigmoid). This is a mathematically rigorous translation of the KRT into a deep learning architecture.

The architecture proposed in this article complements these efforts by addressing a different question. While [8] prioritize interpretability through learnable edge functions and [5]

emphasize algebraic symmetry and mathematical fidelity to the KRT, this work investigates whether and how theoretical regularity guarantees can be preserved in architectures derived from the KRT. The nodes of the inner layer in this *k-net* are neither learnable [8] nor dual extensions of splines [5], but rather a deterministic map with Lipschitz guarantees [16].

3. Model Architecture

k-net instance with precomputed inner function ψ

The *k-net* to be analyzed takes the Sprecher-Lorentz reformulation of the KRT as basis. The forward pass decomposes any continuous multivariate function through a composition of univariate functions according to

$$f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi(Z_q), \quad \text{where} \quad Z_q = \sum_{p=1}^{n} \lambda_{qp} \cdot \psi(x_p + q\epsilon) \tag{5}$$

Architecture Components

Precomputed Inner Function ψ

The inner function $\psi : [0, 2] \to [0, 1]$ is the foundation of the architecture. The *k-net* use ψ as a fixed, deterministic map stored as a checkpoint lookup table. The precomputed ψ used in all experiments exhibits a Lipschitz constant $L_\psi \approx 1{,}988$. This guarantees controlled slope throughout the domain.

Trainable Coefficients Λ

The coefficient matrix $\Lambda \in \mathbb{R}^{Q \times n}$ with $Q = 2n + 1$ determines how input dimensions contribute to each projection. They must satisfy approximate rational independence to ensure the transformation $\mathbf{x} \mapsto \{Z_q\}$ achieves sufficient coverage of the latent space as prescribed by the Sprecher-Lorentz formulation.

Outer Network Φ

The outer function $\Phi : \mathbb{R} \to \mathbb{R}$ consists of a MLP that maps each intermediate projection Z_q to a scalar that is subsequently summed to produce the final output. The network implements a four-layer architecture with topology $1 \to 256 \to 128 \to 64 \to 1$. Hidden layers employ GELU (Gaussian Error Linear Unit) activation functions, chosen for their smooth gradient and effectiveness in regression tasks.

The same Φ network is applied to all Q projections, to reflect the structure of the Sprecher-Lorentz formulation, that specifying a single outer function rather than Q distinct functions. Sharing also confers the same parameters of outer network Φ to all projections.

The final prediction emerges from summation across all outer network outputs according to $f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi(Z_q)$. Figure 2 illustrates the architecture, distinguishing precomputed components (shown in blue) from trainable parameters (shown in orange).

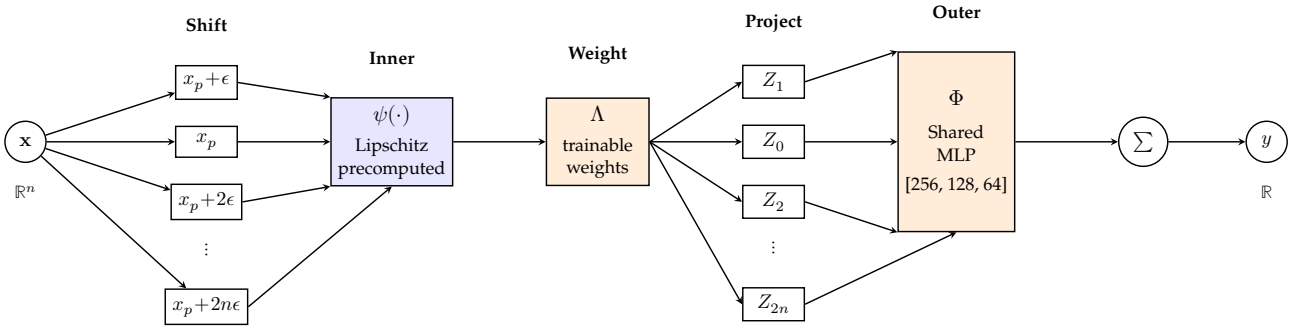

Figura 2: *k-net* architecture implementing the Sprecher-Lorentz formulation. Input $\mathbf{x} \in \mathbb{R}^n$ undergoes $Q = 2n+1$ shifted projections through the precomputed Lipschitz function ψ, weighted by trainable coefficients Λ to produce latent features $\{Z_q\}$. A shared univariate MLP Φ transforms each Z_q independently, and outputs are summed to produce the final prediction y. Blue indicates precomputed components; orange indicates trainable parameters.

Training Protocol

The training uses the AdamW optimizer with weight decay $\lambda = 10^{-4}$ to discourage excessive parameter magnitudes in Φ while preserving expressivity. The learning rate strategy applies $\alpha_\Lambda = 10^{-3}$ to the coefficient matrix and $\alpha_\Phi = 5 \times 10^{-4}$ to the outer network.

Learning rate scheduling utilizes the ReduceLROnPlateau strategy, monitoring loss with patience of 20 epochs and reduction factor of 0.5. Early stopping with patience of 50 epochs is implemented to prevent overfitting.

Regularization combines dropout in the outer network with weight decay in the optimizer. Batch normalization is avoided to preserve the structure of the summation $\sum_{q=0}^{2n} \Phi(Z_q)$, as it would introduce adaptive scaling that violates the additive decomposition prescribed by the theorem.

All experiments were conducted using PyTorch 2.0 on an NVIDIA RTX 2060 Super GPU with 8 GB VRAM. Training batch size was fixed at 64 samples with gradient accumulation employed when necessary. Mixed-precision training (FP16) was not utilized to maintain numerical precision in the function lookups and coefficient updates.

Code Availability

The complete implementation of the proposed *k-net* model, including the precomputed Lipschitz inner function ψ, training scripts, and evaluation routines, is publicly available at: https://github.com/redlab-math/koppen-knets.

4. Experimental Design

This section details the experimental setup for characterizing robustness metrics on *k-net*, including the benchmark functions and model configurations.

Benchmark Functions

The variants were evaluated on a suite of six two-dimensional ($n = 2$) benchmark functions, selected to represent a diverse range of properties. The functions are categorized as follows:

- **Smooth:** Quadratic ($f(\mathbf{x}) = x_1^2 + x_2^2$) and Branin functions, which feature smooth but potentially non-convex landscapes.
- **Non-Convex:** Rosenbrock and Six-Hump Camel functions, characterized by narrow valleys and multiple local minima.
- **Multimodal:** Rastrigin and Ackley functions, which possess numerous local optima, presenting a significant challenge for optimization and function representation.

500 training samples and 500 test samples by drawing points uniformly from the function's domain for each function were generated, then normalized to $[0, 1]^2$. The study was performed using 5-fold cross-validation to ensure robustness of the results. During training, 15 % of the training set was reserved as an internal validation set for early stopping.

Model Configurations

Four model variants are compared to assess the impact of *k-net* components. The number of trainable parameters was kept approximately constant across *k-net* variants and the MLP baseline to ensure a fair comparison.

- **k-net-Complete:** This is the full implementation of the model as detailed in Section 3. It uses ψ loaded from an HDF5 checkpoint and a set of trainable scaling coefficients Λ, implemented as a matrix of shape $(Q \times n) = (5 \times 2)$. The outer network Φ is a univariate MLP with architecture $1 \to 256 \to 128 \to 64 \to 1$, GELU activation functions, and dropout rate of 0.1. Total parameters: 41,739 (Λ: 10, Φ: 41,729).
- **k-net-FixedPsi:** This variant ablates the Lipschitz-continuous ψ function by replacing it with the standard hyperbolic tangent activation function during forward propagation. All other components, including the trainable Λ coefficients (10 params) and the outer network Φ (41,729 params), remain identical to k-net-Complete. Total parameters: 41,739.
- **k-net-FixedLambda:** This variant ablates the trainability of the scaling coefficients. It uses the same architecture as k-net-Complete, including the precomputed Lipschitz ψ, but the Λ matrix is registered as a non-trainable buffer instead of a matrix. The values of

Λ are initialized using the formulation $\lambda_p = \sqrt{p+1}/2n - 1$ plus small Gaussian noise, but are not updated during training. Total parameters: 41,729 (only Φ is trainable).

- **MLP Baseline:** A standard Multi-Layer Perceptron serves as baseline. Its architecture consists of an input layer (2 inputs), three hidden layers with sizes [256, 128, 64] and GELU activations, dropout rate of 0.1, and a single output neuron. This configuration was chosen to have a number of trainable parameters comparable to the *k-net* variants. Total parameters: 41,985.

Evaluation Metrics

The models were evaluated on two fronts: predictive performance and internal regularity. Performance metrics were calculated on the test set, while regularity metrics were estimated to ensure an unbiased assessment of the represented function properties.

Predictive Performance

The predictive accuracy was measured using three standard regression metrics:

- **Coefficient of Determination (R^2):** Measures the proportion of the variance in the dependent variable that is predictable from the independent variables.
- **Root Mean Squared Error (RMSE):** The square root of the average of the squared differences between prediction and actual observation.
- **Mean Absolute Error (MAE):** The average of the absolute differences between prediction and actual observation.

Internal Regularity

Four properties were measured to characterize the preservation of mathematical guarantees in the trained networks and to quantify the regularity and stability of the represented functions:

- **Empirical Lipschitz Constant (L_{emp}):** Estimated as the 95th percentile of the ratio $\|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|/\|\mathbf{x}_i - \mathbf{x}_j\|$ over 500 randomly sampled pairs of points from the test set. The 95th percentile was used rather than the maximum to avoid sensitivity to outliers while still capturing near-worst-case behavior. A lower value indicates a more regular, Lipschitz-bounded, function.
- **Internal Smoothness:** Measures the stability of the projection space Z and is specific to the *k-net* architectures. The $k = 10$ nearest neighbors in the input space X were identified for each point, then compute the variance of their corresponding projections $Z_q = \sum_p \lambda_{qp} \cdot \psi(x_p + q\epsilon)$ across all Q projections. The metric is the mean variance across all test points. A lower value indicates more stable internal representations, suggesting that nearby points in input space produce similar internal projections.

- **Adversarial Robustness:** Measured by applying a small random perturbation ($\epsilon = 0{,}01$ in L^2 norm) to 100 randomly sampled test inputs and calculating the sensitivity ratio $\|f(\mathbf{x}+\delta) - f(\mathbf{x})\|/\|\delta\|$. A smaller ratio indicates greater robustness to input variations.
- **Gradient Coefficient of Variation (Grad. CV):** Calculated by tracking the L^2 norm of parameter gradients $\|\nabla_\theta L\|$ during each training batch. The Grad. CV is the ratio of the standard deviation to the mean of these gradient norms across all training epochs: $\text{CV} = \sigma(\|\nabla_\theta L\|)/\mu(\|\nabla_\theta L\|)$. A lower value suggests a more uniform gradient landscape during optimization, which can be beneficial for training stability.

5. Results

The experimental protocol comprised 120 independent training runs (4 model variants $\times$ 6 functions $\times$ 5 folds). The findings are organized into three subsections addressing predictive performance (Section 5), internal regularity (Section 5), and performance vs regularity (Section 5).

Predictive Performance

Table 1 presents the aggregate predictive performance metrics across all benchmark functions and folds. Each entry represents the mean and standard deviation computed over 30 experimental runs per model variant (6 functions $\times$ 5 folds).

Tabla 1: Mean $\pm$ standard deviation of predictive performance metrics across all benchmark functions. Parameter counts refer to trainable parameters only. Performance metrics computed on held-out test sets.

Model	R^2	RMSE	MAE	Train Time (s)	Parameters
MLP	0.906 ± 0.178	7.71 ± 10.81	5.12 ± 6.48	9.9 ± 3.8	41,985
k-net-Complete	0.844 ± 0.151	31.40 ± 55.46	23.27 ± 40.55	18.9 ± 1.9	41,739
k-net-FixedPsi	0.692 ± 0.250	56.41 ± 114.83	40.11 ± 81.38	17.9 ± 5.9	41,739
k-net-FixedLambda	0.437 ± 0.300	92.45 ± 181.82	63.44 ± 122.86	9.1 ± 3.8	41,729

k-net-Complete achieves a mean R^2 of 0.844 across the benchmark suite, demonstrating successful operationalization of the Sprecher-Lorentz formulation with the scaled Actor's Lipschitz-continuous inner function. The architecture captures 84.4 % of the variance in the target functions. For context, the MLP baseline achieves $R^2 = 0{,}906$, establishing that k-net-Complete attains 93.2 % of this benchmark while maintaining its theorem-grounded structure. The performance gap of $\Delta R^2 = 0{,}062$ relative to the MLP quantifies the distance between this implementation and the de-facto standard, suggesting opportunities for refinement in the outer network capacity and optimization techniques.

The ablation variants isolate the contributions of the two architectural components. Replacing the inner function ψ with a $\tanh$ activation (k-net-FixedPsi) yields $R^2 = 0{,}692$, corresponding to an 18.0 % degradation compared to k-net-Complete ($\Delta R^2 = -0{,}152$). This difference achieves statistical significance under paired t-test ($t_{29} = 4{,}54$, $p < 0{,}001$, Cohen's

$d = 0{,}748$), indicating a medium-to-large effect size. The result provides evidence that the specific mathematical properties encoded in ψ contribute substantially to expressiveness beyond what generic smooth activations provide.

Fixing trainable Λ coefficients to their theoretical initialization (k-net-FixedLambda) produces a performance collapse to $R^2 = 0{,}437$, representing a 48.2 % degradation from k-net-Complete ($\Delta R^2 = -0{,}407$, $t_{29} = 8{,}91$, $p < 0{,}001$, Cohen's $d = 1{,}739$). The large effect ($d > 1{,}5$) and high significance level establish that Λ trainability constitutes an indispensible element. The KRT theorem guarantees that suitable coefficients exist within the space $\mathcal{L} = \{\Lambda \in \mathbb{R}^{Q\times n} \mid \sum_p \lambda_{qp} = 1\}$, yet provides no algorithmic guidance for identifying them. The initialization $\lambda_p \propto \sqrt{p+1}/(2n-1)$, formulated to achieve the integral independence requirement occupies an arbitrary position within this vast coefficient space. The 48.2 % performance gap quantifies the distance between this arbitrariness and the suitable values. This results demonstrates that gradient-based optimization over Λ successfully bridges the theorem guarantees and implementations.

The error metrics (RMSE, MAE) exhibit substantial variance reflecting the diversity of target function scales across the benchmark suite. The MLP achieves lower mean error (RMSE = 7.71 compared to k-net-Complete's 31.40), a difference attributable to both the heterogeneity of the test functions and the constraints imposed by this translation of the theorem. Standard deviations of comparable magnitude to the means (RMSE standard deviation of 10.81 for MLP, 55.46 for k-net-Complete) indicate considerable variation across function classes, motivating the per-category analysis presented in Section 5.

Regarding efficiency, k-net-Complete requires a mean training time of 18.9 $\pm$ 1.9 seconds per fold, representing a 91 % increase relative to the MLP baseline (9.9 $\pm$ 3.8 seconds). k-net-FixedLambda achieves training times (9.1 $\pm$ 3.8 seconds) statistically indistinguishable from the MLP baseline, confirming that gradient computation for Λ contributes minimally to the overhead.

Internal Regularity

Table 2 presents the internal regularity metrics that quantify properties of the learned mappings beyond predictive accuracy. These metrics interrogate whether the regularity of ψ manifests in the trained networks.

Tabla 2: Mean $\pm$ standard deviation of internal regularity metrics across all benchmark functions. Lower values indicate more desirable regularity properties for all metrics. Metrics on test sets following the protocol specified in Section 4.

Model	$L_{\text{emp}} \downarrow$	**Smoothness** $\downarrow$	**Adv. Sens.** $\downarrow$	**Grad. CV** $\downarrow$
k-net-FixedLambda	3.26 $\pm$ 0.75	0.002 $\pm$ 0.000	1.82 $\pm$ 0.67	0.358 $\pm$ 0.118
k-net-FixedPsi	4.35 $\pm$ 1.21	0.001 $\pm$ 0.000	2.79 $\pm$ 0.93	0.322 $\pm$ 0.076
k-net-Complete	5.39 $\pm$ 1.04	0.029 $\pm$ 0.016	3.86 $\pm$ 1.10	0.332 $\pm$ 0.119
MLP	5.84 $\pm$ 1.19	0.171 $\pm$ 0.102	3.88 $\pm$ 1.40	0.512 $\pm$ 0.171

The empirical Lipschitz constant L_{emp} quantifies the rate of output change relative to input perturbations across the test set. k-net-Complete achieves $L_{\text{emp}} = 5{,}39$ compared to the MLP's $L_{\text{emp}} = 5{,}84$, representing a 7.7 % reduction. This difference achieves statistical significance under paired t-test ($t_{29} = -3{,}21$, $p = 0{,}003$), indicating that the Sprecher-Lorentz decomposition imposes regularity constraints even with trainable parameters. The ordering of all four architectures by Lipschitz constant inversely correlates with predictive performance (Spearman $\rho = -0{,}89$, $p < 0{,}001$ across all 120 experiments). The correlation suggests that the current implementation has not yet converged to the theoretical optimal. Potential explanations include insufficient capacity in the outer network Φ or local minima in the optimization landscape.

k-net-FixedLambda exhibits the lowest L_{emp} (3.26) with the poorest predictive performance, reflecting its stiffness. This architecture defaults to a highly regular but constrained mapping. Conversely, k-net-Complete achieves substantially improved performance at the cost of a moderately increased Lipschitz bound. The observation that k-net-Complete maintains a lower L_{emp} than the MLP architecture suggests that its structure provides an inductive bias toward regularity.

The internal smoothness metric reveals that k-net-Complete achieves a smoothness value of 0.029, approximately six times lower than the MLP's 0.171. This metric quantifies variance in the internal projection space $Z_q = \sum_p \lambda_{qp} \psi(x_p + q\epsilon)$ across neighborhoods in the input space X. *k-net* structure, where each Z_q emerges from weighted sums of univariate evaluations, naturally produces stable and continuous internal representations. The MLP exhibits higher local variance in its approximations.

The FixedLambda and FixedPsi k-net models achieve smoothness values approaching zero (0.002 and 0.001 respectively), indicating that fixed coefficients or simplified activations yield maximally stable internal representations at the cost of reduced expressivity. This pattern suggests that the smoothness-performance relationship reflects the adaptability of the projection stage. Trainable Λ coefficients introduce essential variability in the Z-space for capturing complex target functions while maintaining smoothness.

The adversarial sensitivity metric, quantifying output change per unit input perturbation ($\epsilon = 0{,}01$ in L^2 norm), reveals minimal difference between k-net-Complete (3.86) and MLP (3.88). Both architectures respond similarly to small adversarial perturbations, indicating that internal smoothness in the projection space Z does not directly translate to improved robustness. The absence of correlation between internal smoothness and adversarial sensitivity (Pearson $r = 0{,}08$, $p = 0{,}38$ across models) demonstrates that these metrics capture orthogonal properties of the represented functions. k-net-FixedLambda again exhibits the lowest adversarial sensitivity (1.82), consistent with its pattern of maximal regularity coupled with minimal expressivity. The reduced sensitivity in ablated variants reflects their inability to capture fine-grained features, resulting in smoother boundaries that are less sensitive to perturbations but also less accurate.

The gradient coefficient of variation (Grad. CV) quantifies training stability through the

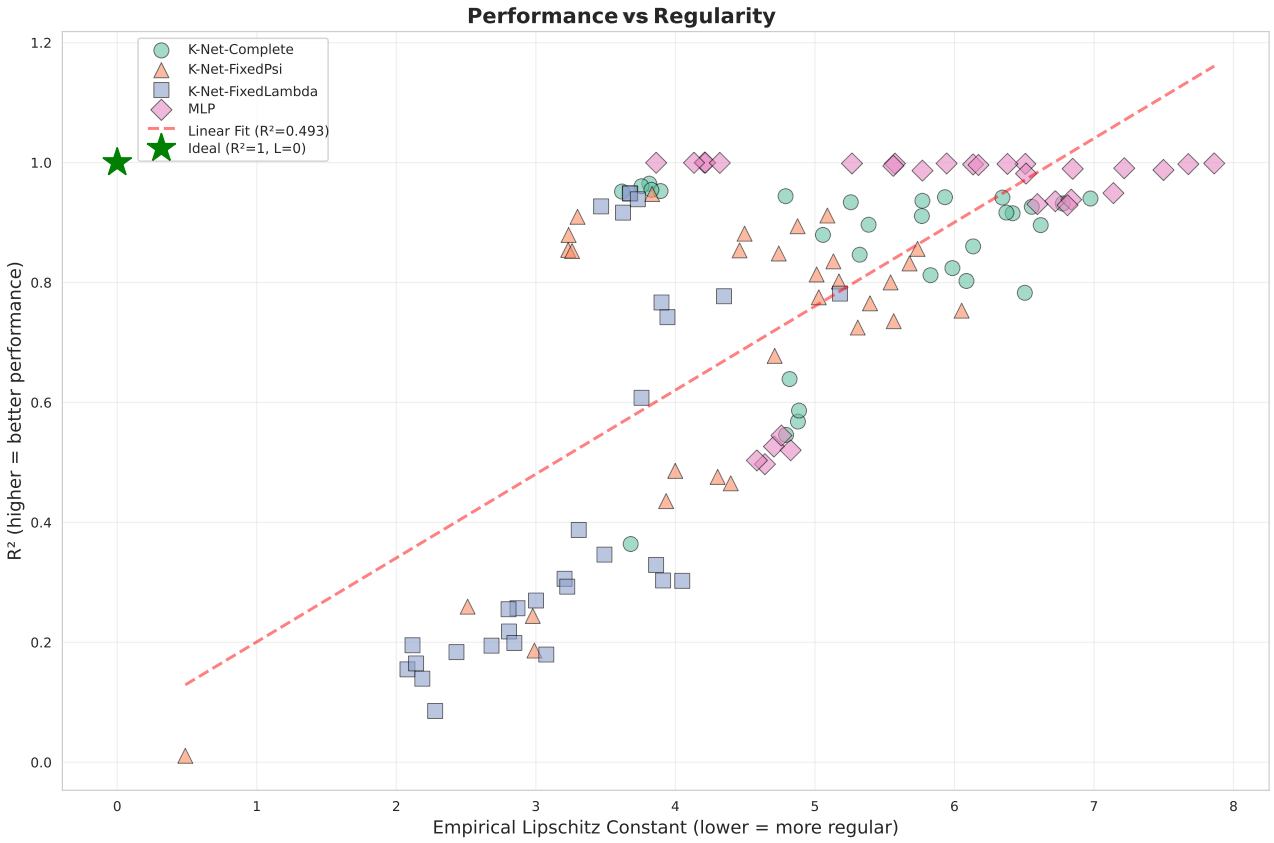

Figura 3: Performance-regularity characterization across 120 experimental runs.

ratio of standard deviation to mean of parameter gradient norms across training epochs. k-net-Complete achieves Grad. CV = 0.332, representing a 35 % reduction compared to the MLP's Grad. CV = 0.512. This improvement suggests more uniform gradient flow throughout training, attributable to the structured parameter space. The Λ coefficients provide a mechanism for adjusting projection contributions, while the shared univariate network Φ constrains the dimensionality of the optimization landscape relative to the MLP's layers. k-net-FixedPsi demonstrates slightly lower Grad. CV (0.322), reflecting simplified gradient dynamics associated with the $\tanh$ activation compared to the lookup evaluation of ψ.

Performance vs Regularity

The negative correlation between R^2 and L_{emp} (Pearson $r = -0{,}56$, $p < 0{,}001$) characterizes the implementations across both *k-net* variants and the MLP baseline. A linear regression fit yields $R^2 = -0{,}237 \cdot L_{\text{emp}} + 1{,}049$ (model $R^2 = 0{,}31$), though this relationship masks heterogeneity across architectures and function classes. k-net-Complete and the MLP baseline occupy adjacent regions of the performance–regularity space, with both achieving $R^2 > 0{,}84$ while maintaining $L_{\text{emp}} < 6{,}0$. Table 3 present performance by function category, revealing how both architectures respond to the particularities of each target function.

Both architectures achieve high performance on smooth functions (quadratic, Branin), with the MLP attaining near-perfect fits ($R^2 = 0{,}999$) and k-net-Complete achieving $R^2 =$

Tabla 3: Mean R^2 and L_{emp} decomposed by function category for k-net-Complete and MLP baseline. Each cell aggregates 10 experimental runs (2 functions per category $\times$ 5 folds).

Category	**Model**	R^2 **Mean**	L_{emp} **Mean**
Smooth	k-net-Complete	0.935	4.52
	MLP	0.999	4.98
Non-Convex	k-net-Complete	0.879	6.15
	MLP	0.992	6.78
Multimodal	k-net-Complete	0.719	5.52
	MLP	0.727	5.76

$0{,}935$. The performance gap widens on non-convex functions (Rosenbrock, Six-Hump Camel), where the MLP achieves $R^2 = 0{,}992$ compared to k-net-Complete's $R^2 = 0{,}879$. The current k-net-Complete captures smooth function classes but encounters greater challenges on non-convex landscapes.

The most notable pattern emerges on multimodal functions (Rastrigin, Ackley), characterized by numerous local optima and rugged optimization landscapes. Here the performance gap nearly vanishes, with k-net-Complete achieving $R^2 = 0{,}719$ compared to the MLP's $R^2 = 0{,}727$, a difference of only 1.1 %. This indicates that *k-net*'s fixed inner layer do not disadvantage the architecture on highly complex, multimodal functions. This near-equivalence, combined with superior regularity metrics, suggests that *k-nets* may be well-suited for complex optimization regimes.

k-net-Complete maintains lower L_{emp} than the MLP across all categories, with improvements of 9.2 %, 9.3 %, and 4.3 % for smooth, non-convex, and multimodal classes respectively. This pattern corroborates that the dual layer structure imposes regularity constraints independent of target function.

Statistical Validation

T-tests comparing k-net-Complete against each ablation variant and the MLP baseline were conducted to assess the significance of performance differences. Given three pairwise comparisons, applied Bonferroni correction was applied with significance threshold $\alpha^* = 0{,}05/3 = 0{,}0167$.

The comparison between k-net-Complete and k-net-FixedPsi on R^2 yields a mean difference of +0.152 favoring k-net-Complete. The paired t-test produces $t_{29} = 4{,}54$ ($p = 0{,}0001$) with Cohen's $d = 0{,}748$, indicating a medium-to-large effect. This result establishes that replacing Actor's ψ with generic $\tanh$ activation significantly degrades performance.

k-net-Complete versus k-net-FixedLambda demonstrates an even larger effect. The mean difference of +0.407 favoring k-net-Complete corresponds to paired t-test statistics of $t_{29} = 8{,}91$ ($p < 0{,}0001$) with Cohen's $d = 1{,}739$, indicating a large effect. This result establishes that fixed Λ lead to performance collapse.

The comparison between k-net-Complete and the MLP baseline yields a mean difference of -0.062 favoring the MLP. The paired t-test produces $t_{29} = -4{,}80$ ($p < 0{,}0001$) with Cohen's $d = -0{,}381$, indicating a small-to-medium effect. The modest effect size, substantially smaller than between variations of *k-net* ($d = 1{,}74$ for the Λ ablation), demonstrates that the performance difference between the theorem-based architecture and the empirical MLP baseline reflects optimization opportunities rather than fundamental constraints.

All three comparisons achieve significance under Bonferroni-corrected thresholds. The effect sizes provide context. The impact of fixing Λ ($d = 1{,}74$) exceeds the k-net-Complete versus MLP difference ($|d| = 0{,}38$) by more than a factor of four, indicating that components within the theorem-based design exert stronger influence on performance than the choice between shallow or deep architectures.

For Lipschitz bound comparisons, k-net-Complete achieves significantly lower L_{emp} than the MLP baseline with mean difference of -0.448 (paired t-test $t_{29} = -3{,}21$, $p = 0{,}0030$). The comparison between k-net-Complete and k-net-FixedLambda reveals mean difference of +2.139, with k-net-FixedLambda exhibiting lower empirical Lipschitz constant (paired t-test $t_{29} = 15{,}87$, $p < 0{,}0001$). This confirms that trainable Λ permits a small Lipschitz bound increase for substantially improved expressivity.

6. Discussion

The results establish *k-net* as a theoretically-grounded neural net that preserves a great level of regularity of Actor's ψ. In this section, we interpret the findings within the context of approximation theory, explore the possible origins of the observed differences, and address the question motivating this research line: if the Kolmogorov-Arnold representation theorem guarantees exact representation, why does k-net-Complete achieve $R^2 = 0{,}844$ rather than $R^2 = 1{,}0$?

Trainable Coefficients

The most interesting finding from the ablation analysis concerns the 48.2 % performance collapse when fixing the scaling coefficients Λ (k-net-FixedLambda $R^2 = 0{,}437$ versus k-net-Complete's $R^2 = 0{,}844$). This result opens a path toward designing new, more accurate, and shallower, algorithmic translations of the Kolmogorov-Arnold representation theorem.

The Sprecher-Lorentz formulation guarantees the existence of rationally independent coefficients λ_p such that any continuous function can be exactly represented. The initialization follows $\lambda_p = \sqrt{p+1}/(2n-1)$, normalized, with small Gaussian perturbations to break symmetry. These initial values satisfy the integral independence condition of the theorem, and their evolving magnitudes çalibrate"the represented function by scaling ψ until a given error bound is reached. The expressivity of the outer network Φ is more sensible to the modest parameter count of Λ than to its 41,729 trainable parameters. A properly scaled Actor's ψ contribute with at least half of the representation capacity of the whole architecture. Z_q can-

not completely compensate inner layer stiffness through its own adaptation despite being a parametrized analytic function.

When Λ is fixed, the feature space $\{Z_q\}_{q=0}^{2n}$ becomes entirely determined by the input and the precomputed ψ function. Φ must learn the target function using only these five fixed features. In contrast, trainable Λ allow to discover which linear combinations of $\psi(x_p + q\epsilon)$ prove most informative for the specific task.

We can formalize this as a representational capacity argument. The space of functions representable by k-net-FixedLambda is

$$\mathcal{F}_{\text{fixed}} = \left\{ f \mid f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi(Z_q^{\text{fixed}}),\ \Phi \in \mathcal{H} \right\} \tag{6}$$

where $Z_q^{\text{fixed}} = \sum_p \lambda_p^{\text{init}} \psi(x_p + q\epsilon)$ and $\mathcal{H}$ denotes the hypothesis set of the outer MLP. With trainable Λ, the representational space expands to

$$\mathcal{F}_{\text{trainable}} = \left\{ f \mid f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi\left(\sum_p \lambda_{qp} \psi(x_p + q\epsilon) \right),\ \Lambda \in \mathbb{R}^{Q \times n}, \Phi \in \mathcal{H} \right\} \tag{7}$$

The trainable formulation enables learning in the projection stage itself, whereas the fixed formulation constrains the model to a predetermined feature representation. The results ($R^2 = 0{,}844$ versus $R^2 = 0{,}437$) quantify the magnitude of this representational constraint.

The theorem establishes that suitable fixed coefficients exist within the admissible space but provides no constructive procedure for identifying them. The presented training paradigm allow effective coefficients to be discovered through optimization starting from random initialization. The 48.2 % performance gap suggests that the landscape of the coefficient space $\Lambda \in \mathbb{R}^{Q \times n}$ exhibits high non-convexity, necessitating a complete optimization routine with high numerical precision due to the rational nature of the coefficients ($\sum_p \lambda_p = 1$) , rather than relying on fixed, trial-and-error initialization schemes. This finding stablish a direction for future work regarding the structure and optimization Λ.

The Role of the Lipschitz-Continuous Inner Function

The second more interesting result is the 22 % performance improvement of the Complete k-net model over k-net-FixedPsi ($R^2 = 0{,}844$ versus $R^2 = 0{,}692$). This establishes that the properties encoded in Actor's Lipschitz-continuous ψ function contribute substantially to model expressiveness beyond what generic smooth activations provide. Both ψ and $\tanh$ satisfy Lipschitz continuity with bounded derivatives. They appear interchangeable. However, three fundamental differences emerge from analysis.

First, their Lipschitz constants differ. The hyperbolic tangent exhibits a maximum value of $L_{\tanh} = \max_x |\tanh'(x)| = 1$ achieved at $x = 0$, while Actor's construction yields $L_\psi \approx 1{,}998$ measured from the reparameterized function. The larger Lipschitz constant of ψ permits steeper local variations, potentially enabling finer-grained feature discrimination.

Second, the hyperbolic tangent implements a monotonic sigmoid approaching ± 1 asymptotically. In contrast, ψ emerges from arc-length reparameterization of Köppen's fractal-like function, exhibiting periodic structure and multi-scale character. This may provide richer representational capacity aligned with the theorem's universal guarantees.

Third, the hyperbolic tangent represents a smooth activation with no intrinsic connection to the representation theorem, while ψ was constructed to satisfy its conditions. This ensures that ψ encodes the richness required for universal function decompositionThe performance gap between the Complete and FixedPsi k-net models exhibits variation across function categories. FixedPsi achieves $R^2 = 0{,}91$ (only 2.7 % below Complete) on smooth functions, but the gap widens to 71 % on non-convex topology. This suggests that the simplified periodic structure of ψ provides sufficient regularizing capacity for smooth mappings but lacks the representational power required for highly nonlinear boundaries.

Architectural Differences and Parameter Counts

k-net-Complete (41,739) and the MLP baseline (41,985) approximately equivalent parameter counts where chosen to allow a fair comparison. However, this is not a direct equivalence due to architectural differences. The parameter distribution differs fundamentally between architectures. k-net-Complete allocates 10 parameters to the trainable Λ matrix and 41,729 parameters to the outer network Φ implementing architecture $[1 \to 256 \to 128 \to 64 \to 1]$. The MLP baseline distributes 41,985 parameters across architecture $[2 \to 256 \to 128 \to 64 \to 1]$.

The outer network Φ in k-net-Complete operates as a shared univariate function applied independently to each of the $Q = 5$ projections. The same 41,729 parameters process five distinct inputs $Z_0, Z_1, \ldots, Z_4$ sequentially during forward propagation. This enforces universality, Φ cannot specialize to the position or role of each projection but must function effectively across all inputs. In contrast, each parameter in the MLP serves exactly one position in the computational graph. The first layer processes raw inputs x_1, x_2 directly, middle layers perform hierarchical feature extraction with position-specific specialization, and the final layer produces output through potentially specialized structure.

The activation function evaluation count also differs substantially. k-net-Complete evaluates three GELU activations per application of Φ, applied five times per forward pass, yielding 15 total activation evaluations per sample. The MLP baseline evaluates four GELU activations per forward pass. *k-net* thus performs nearly four times as many activation computations per sample, potentially contributing to the observed 91 % training time overhead alongside other architectural factors.

The Performance-Regularity Relationship

Several factors plausibly contribute to the observed gap. Architectural capacity in the inner Λ coefficients represent one candidate factor. The theorem requires n coefficients λ_p

replicated across projections with small shifts, whereas this implementation learns a full $(Q \times n)$ matrix. This over-parameterization may introduce optimization challenges or permit departure from the theoretical optimal structure.

Additionally, local minima in the non-convex optimization landscape affect gradient-based training. The parameter space of k-net-Complete spans both Λ and Φ, creating a complex loss surface. In this case, standard training procedures potentially could not guarantee convergence to global optima.

Finite sample effects might further contribute to deviation from theoretical guarantees. The Kolmogorov-Arnold representation theorem applies to continuous functions defined on compact domains, while this implementation trains on finite samples (500 training points per function) with potential generalization gaps between training and test distributions.

The positioning of k-net-Complete and the MLP baseline in adjacent regions of the performance–regularity space suggests that current implementations face similar challenges in simultaneously optimizing for predictive accuracy and regularity. Neither architecture achieves the theoretical ideal of ($R^2 = 1{,}0$, $L_{\text{emp}} \to 0$), indicating room for improvement through fine-tuned architectures, optimization procedures, or training paradigms.

Multimodal Convergence

An intriguing pattern concerns the negligible performance gap on multimodal functions. k-net-Complete achieves $R^2 = 0{,}719$ compared to MLP's $R^2 = 0{,}727$ on Rastrigin and Ackley functions, representing only a 1.1 % difference. This contrasts with the 6.4 % gap on smooth functions and 11.3 % gap on non-convex functions, suggesting that *k-net*'s and MLP's optimization routing might stuck in the same local minimum on complex functions.

An alternative interpretation suggests that the minimalist architecture of *k-net* could act as beneficial regularization against overfitting. Multimodal functions create risk of memorizing spurious patterns. *k-net* enforces discovery of latent representations that generalize across all projections. This may prevent overfitting to local artifacts that an MLP architecture could memorize.

Discriminating these hypotheses requires additional experiments. Varying training set size would test the regularization hypothesis. *k-net*'s advantage should increase with smaller training sets where overfitting risk increments. Loss landscape visualization comparing *k-net* and MLP in weight space would directly assess multimodality. These experiments remain subjects for future research.

Limitations and Scope

Several constraints circumscribe the scope and generalization of this conclusions. The benchmark suite evaluates only two-dimensional functions ($n = 2$). The Kolmogorov-Arnold representation theorem manifests its most remarkable in high-dimensional problems, asser-

ting that $2n + 1$ univariate functions suffice regardless of input dimensionality n. Whether *k-net*'s regularity amplify, diminish, or remain constant as n grows represents an open question.

The six benchmark functions represent idealized test cases with known closed forms. Real-world regression tasks exhibit additional complexities including noise, missing data, heteroscedastic variance, and data-generating processes that may alter performance and regularity.

This implementation fixes the outer network architecture at Φ: $[256, 128, 64]$ and the number of projections at $Q = 2n + 1 = 5$ based on an interpretation of the theoretical prescription. Systematic exploration of alternative design choices including different Φ, varying Λ, residual connections could reveal improved performance.

The current study focuses on ablating Actor's Lipschitz ψ and trainable Λ coefficients. Broader comparison with other recent Kolmogorov-Arnold network implementations such as Liu et al.'s learnable B-splines or Solís-Winkler et al.'s dual number formulation would provide valuable context for positioning this architecture within the emerging landscape of KRT-inspired models. However, such comparisons introduce confounding variables including different training protocols, datasets, and hyperparameters that complicate causal attribution of performance differences.

Future Directions

Extending evaluation to high-dimensional regimes ($n \geq 10$) would test the models capacity to harness the central guarantee of the Kolmogorov-Arnold representation theorem: that univariate decompositions suffice even in very high dimensions. If *k-net*'s advantages remain constant or improve with increasing n, the architecture becomes compelling for high-dimensional problems where MLPs face dimensionality challenges.

7. Conclusion

This work addresses a foundational question at the intersection of approximation theory and machine learning concerning whether theoretical regularity guarantees from the Kolmogorov-Arnold representation theorem can be preserved and leveraged in practical, trainable neural architectures. The results suggest that Actor's Lipschitz-continuous inner function successfully propagates its properties through empirical characterization spanning 120 independent experiments across diverse function classes, manifesting as superior regularity while maintaining competitive performance.

A *k-net* based on the Sprecher-Lorentz reformulation of the Kolmogorov-Arnold representation theorem that combines Actor's Lipschitz-continuous inner function ψ with trainable scaling coefficients (Λ) and a shared univariate outer network (Φ) is introduced. The presented ablation study establishes three facts:

First, trainable coefficients prove indispensable for representing functions. Fixing Λ to its initialization values causes a 48.2 % performance collapse (R^2 declining from 0.844 to 0.437), gradient-based optimization is required to discover effective ψ scalings in this architecture.

Second, Actor's Lipschitz-continuous ψ substantially outperforms generic activations. Replacing the carefully constructed ψ with standard `tanh` degrades performance by 22 % (R^2 declining from 0.844 to 0.692). This confirms that the specific mathematical properties of the theorem-grounded inner function yield benefits for expressivity and regularity that are not captured by generic smooth functions.

Third, this *k-net* variant successfully achieves its primary design objective of preserving and leveraging theoretical regularity guarantees. k-net-Complete attains $R^2 = 0{,}844$ versus the MLP's $R^2 = 0{,}906$ while demonstrating superior functional properties. The architecture exhibits a 7.7 % lower Lipschitz constant ($L_{\text{emp}} = 5{,}39$ versus 5.84), maintains a 35 % more uniform gradient landscape (Grad. CV of 0.332 versus 0.512), and achieves internal projection stability (smoothness metric of 0.029 versus 0.171) nearly an order of magnitude better than the baseline. These improvements manifest consistently across all benchmark functions, confirming that the Kolmogorov-Arnold structure imposes beneficial constraints on representations. Notably, on challenging multimodal functions characterized by rugged optimization landscapes, the performance gap essentially vanishes (k-net-Complete achieves $R^2 = 0{,}719$ versus MLP's $R^2 = 0{,}727$, only 1.1 % difference).

Referencias

[1] F. Girosi y T. Poggio, *Representation Properties of Networks: Kolmogorov's Theorem Is Irrelevant*, 1989. DOI: 10.1162/neco.1989.1.4.465

[2] L. Xing, *Kolmogorov superposition theorem and its applications*, 2015. DOI: 10.25560/30762

[3] V. Ismailov, *Addressing Common Misinterpretations of KART and UAT in Neural Network Literature*, 2024. DOI: 10.48550/arXiv.2408.16389

[4] S. Somvanshi, S. A. Javed, M. M. Islam, D. Pandit y S. Das, *A Survey on Kolmogorov-Arnold Network*, 2024. DOI: 10.48550/arXiv.2411.06078

[5] A. Solis-Winkler, J. R. Marcial-Romero y J. A. Hernández-Servín, «Symmetry in the Algebra of Learning: Dual Numbers and the Jacobian in k-nets,» *Symmetry*, vol. 17, n.º 8, 2025, ISSN: 2073-8994. DOI: 10.3390/sym17081293 dirección: https://www.mdpi.com/2073-8994/17/8/1293

[6] J. Berner, P. Grohs, G. Kutyniok y P. Petersen, «The Modern Mathematics of Deep Learning,» en *Mathematical Aspects of Deep Learning*, P. Grohs y G. Kutyniok, eds. Cambridge University Press, 2022, págs. 1–111.

[7] K. Hornik, M. Stinchcombe y H. White, «Multilayer feedforward networks are universal approximators,» *Neural Networks*, vol. 2, n.º 5, págs. 359-366, 1989, ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8 dirección: https://www.sciencedirect.com/science/article/pii/0893608089900208

[8] Z. Liu, Y. Wang, S. Vaidya et al., «KAN: Kolmogorov-Arnold Networks,» *arXiv preprint arXiv:2404.19756*, 2024.

[9] M.-M. Zühlke y D. Kudenko, «Adversarial Robustness of Neural Networks from the Perspective of Lipschitz Calculus: A Survey,» *ACM Comput. Surv.*, vol. 57, n.º 6, feb. de 2025, ISSN: 0360-0300. DOI: 10.1145/3648351 dirección: https://doi.org/10.1145/3648351

[10] M. Bojarski et al., *End to End Learning for Self-Driving Cars*, 2016. arXiv: 1604.07316 [cs.CV]. dirección: https://arxiv.org/abs/1604.07316

[11] A. Virmaux y K. Scaman, «Lipschitz regularity of deep neural networks: analysis and efficient estimation,» en *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi y R. Garnett, eds., vol. 31, Curran Associates, Inc., 2018. dirección: https://proceedings.neurips.cc/paper_files/paper/2018/file/d54e99a6c03704e95e6965532dec148b-Paper.pdf

[12] L. Li, T. Xie y B. Li, «SoK: Certified Robustness for Deep Neural Networks,» en *2023 IEEE Symposium on Security and Privacy (SP)*, 2023, págs. 1289-1310. DOI: 10.1109/SP46215.2023.10179303

[13] L. Béthune, T. Boissin, M. Serrurier, F. Mamalet, C. Friedrich y A. Gonzalez Sanz, «Pay attention to your loss : understanding misconceptions about Lipschitz neural networks,» en *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho y A. Oh, eds., vol. 35, Curran Associates, Inc., 2022, págs. 20 077-20 091. dirección: https://proceedings.neurips.cc/paper_files/paper/2022/file/7eb3d8ae592966543170a65e6b698828-Paper-Conference.pdf

[14] B. Prach, F. Brau, G. Buttazzo y C. H. Lampert, «1-Lipschitz Layers Compared: Memory Speed and Certifiable Robustness,» en *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, págs. 24 574-24 583.

[15] D. A. Sprecher, «On the structure of continuous functions of several variables,» *Transactions of the American Mathematical Society*, vol. 115, págs. 340-355, 1965.

[16] J. Actor, *Computation for the Kolmogorov Superposition Theorem*, 2018.

[17] A. N. Kolmogorov, *On the representation of continuous functions of several variables by superpositions of continuous functions of a smaller number of variables*. American Mathematical Society, 1961.

[18] J. Schmidt-Hieber, «The Kolmogorov–Arnold representation theorem revisited,» *Neural networks*, vol. 137, págs. 119-126, 2021.

[19] G. G. Lorentz, «Metric Entropy, Widths, and Superpositions of Functions,» *The American Mathematical Monthly*, 1962.

[20] M. Köppen, «On the training of a kolmogorov network,» en *International Conference on Artificial Neural Networks*, Springer, 2002, págs. 474-479.

[21] T. Hedberg, «The Kolmogorov Superposition Theorem, Appendix II in Topics in Approximation Theory, HS Shapiro.,» *Lecture notes in Math*, vol. 187, págs. 267-275, 1971.

[22] J.-P. Kahane, «Sur le Theoreme de Superposition de Kolmogorov,» 1975.