Sprecher Networks: A Generalized Neural Architecture for Multivariate Function Approximation

Christian Hägg Boris Shapiro

Abstract

We present *Sprecher Networks*, a family of trainable neural architectures inspired by the classical Kolmogorov–Arnold–Sprecher construction for approximating multivariate continuous functions. Unlike typical deep networks, these models rely on *monotonic and general splines* to implement learnable transformations with explicit shifts and summations. Our approach reproduces Sprecher's original one-layer "sum of shifted splines" formula and extends it to deeper, multi-layer compositions. We discuss theoretical motivations, implementation details, and potential advantages in interpretability and universality.

1 Introduction and Historical Background

Approximation of continuous functions by sums of univariate functions has been a recurring theme in mathematical analysis and neural networks. Kolmogorov's Superposition Theorem (1957), refined by Arnold, established that any multivariate continuous function $f : [0,1]^d \to \mathbb{R}$ can be represented as a finite sum of strictly increasing univariate functions applied to affine transformations of $x_1, \ldots, x_d$.

David Sprecher's 1965 Construction.

$$f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi\Big(\sum_{p=1}^{n} \lambda_p \, \phi\big(x_p + \eta\, q\big) + q \Big)$$

In his 1965 landmark paper, David Sprecher [1] proposed a simplified version of the Kolmogorov–Arnold representation that specifically replaces families of $\phi_{q,p}$ by a single monotonic function ϕ with suitable argument shifts. Concretely, Sprecher showed that this single-layer construction suffices to approximate any d-variate continuous function when $n \geq d$ and the parameters $(\eta, \lambda_1, \ldots, \lambda_n)$ and splines (ϕ, Φ) are chosen carefully. His result preserves the key insight of *shifting the input coordinates* by multiples of η and summing the partial evaluations under a final univariate map Φ.

2 Motivation and Overview of Sprecher Networks

Modern deep learning practice often focuses on conventional feedforward architectures. However, Sprecher's approach offers a highly interpretable alternative:

- Each layer in our network is organized around a fixed monotonic spline $\phi(\cdot)$ and a more general spline $\Phi(\cdot)$.
- It incorporates trainable shifts via a scalar η, and trainable weights λ that mix each input coordinate before passing them into ϕ.
- Finally, the network sums over shifted partial evaluations inside an outer spline, closely mimicking Sprecher's formula.

Our *Sprecher Networks* generalize this concept to *multiple layers* by stacking blocks, each of which applies

$$(x_i)_{i=1}^{d_{\text{in}}} \;\mapsto\; \Big[\Phi\big(\textstyle\sum_i \lambda_{i,q}\, \phi(x_i + \eta\, q) + q\big)\Big]_{q=0}^{d_{\text{out}}-1}$$

followed (in the final block) by summation over q to produce a single-dimensional output if desired. In the original 1965 paper, Sprecher used only one such layer (with $2n+1$ summands). Our approach allows

$$d_0 \to d_1 \to \cdots \to d_{L-1} \to 1$$

in multiple *layer blocks*, each embedding the shift-and-sum routine. By doing so, we recover a deeper analog of the Kolmogorov–Arnold–Sprecher construction with additional expressive power, while retaining a clear theoretical link to the classical result.

3 Core Architectural Details

3.1 Single Block Structure

A single *Sprecher block*, which transforms $\mathbb{R}^{d_{\text{in}}}$ to $\mathbb{R}^{d_{\text{out}}}$, consists of:

- **Monotonic spline** $\phi(\cdot)$, which is an increasing piecewise linear function on an interval $[a, b]$.
- **General spline** $\Phi(\cdot)$, a piecewise linear function without strict monotonic constraints, typically defined on a sufficiently large interval such as $[-10, 12]$.
- A matrix of *mixing weights* $\{\lambda_{i,q}\}$ of size $d_{\text{in}} \times d_{\text{out}}$.
- A scalar shift parameter $\eta > 0$ controlling how the index q shifts each coordinate.

Concretely, the block outputs d_{out} coordinates indexed by $q = 0, \ldots, d_{\text{out}} - 1$:

$$\text{Block}_{\phi,\Phi,\eta,\lambda}(\mathbf{x})_q = \Phi\Big(\sum_{i=1}^{d_{\text{in}}} \lambda_{i,q}\, \phi\big(x_i + \eta\, q\big) + q\Big).$$

3.2 Layer Composition and Final Summation

Let L be the number of blocks. Suppose the ℓ-th block maps $\mathbb{R}^{d_{\ell-1}}$ to $\mathbb{R}^{d_\ell}$. Denote its monotonic spline by $\phi^{(\ell)}$, its general spline by $\Phi^{(\ell)}$, its weights by $\{\lambda_{i,q}^{(\ell)}\}$, and shift scalar by $\eta^{(\ell)}$. If $\mathbf{h}^{(\ell-1)}$ is the output of the previous layer (with $\mathbf{h}^{(0)} = \mathbf{x}$), then

$$\mathbf{h}_q^{(\ell)} = \Phi^{(\ell)}\Big(\sum_{i=1}^{d_{\ell-1}} \lambda_{i,q}^{(\ell)}\, \phi^{(\ell)}\big(\mathbf{h}_i^{(\ell-1)} + \eta^{(\ell)}\, q\big) + q\Big), \quad q = 0, \ldots, d_\ell - 1.$$

If the final block has $d_L = 1$, we *sum* its q-outputs to get a scalar function value,

$$f(\mathbf{x}) = \sum_{q=0}^{d_L - 1} \mathbf{h}_q^{(L)}.$$

This recovers the structure of Sprecher's formula when $L = 1$, but generalized to multiple layers for $L > 1$.

3.2.1 Single Hidden Layer ($L = 1$)

For a SprecherNetwork with $d_0 = d_{\text{in}}$ inputs, one block of dimension d_1, and final summation, we have

$$f(\mathbf{x}) = \sum_{q=0}^{d_1 - 1} \Phi^{(1)}\Big(\sum_{i=1}^{d_0} \lambda_{i,q}^{(1)}\, \phi^{(1)}\big(x_i + \eta^{(1)}\, q\big) + q\Big).$$

This precisely reproduces Sprecher's 1965 construction

$$f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi\big(\sum_{p=1}^{n} \lambda_p\, \phi(x_p + \eta\, q) + q\big)$$

if $d_1 = 2d_0 + 1$ and we map $\phi^{(1)} = \phi,\ \Phi^{(1)} = \Phi$.

3.2.2 Two Hidden Layers ($L = 2$)

Let $d_0 = d_{\text{in}}, d_1$ be the size of the first hidden block, and d_2 the size of the second (final) block, which we sum over. Denoting the intermediate output as

$$\mathbf{h}_r^{(1)} = \Phi^{(1)}\Big(\sum_{i=1}^{d_0} \lambda_{i,r}^{(1)}\, \phi^{(1)}\big(x_i + \eta^{(1)}\, r\big) + r\Big), \quad r = 0, \ldots, d_1 - 1,$$

the second block becomes

$$\mathbf{h}_q^{(2)} = \Phi^{(2)}\Big(\sum_{r=0}^{d_1-1} \lambda_{r,q}^{(2)}\, \phi^{(2)}\big(\mathbf{h}_r^{(1)} + \eta^{(2)}\, q\big) + q\Big), \quad q = 0, \ldots, d_2 - 1.$$

Finally, the *network output* is

$$f(\mathbf{x}) = \sum_{q=0}^{d_2-1} \mathbf{h}_q^{(2)}.$$

Equivalently, we can write this as

$$f(\mathbf{x}) = \sum_{q=0}^{d_2-1} \Phi^{(2)}\Big(\sum_{r=0}^{d_1-1} \lambda_{r,q}^{(2)}\, \phi^{(2)}\Big(\Phi^{(1)}\big(\sum_{i=1}^{d_0} \lambda_{i,r}^{(1)}\, \phi^{(1)}(x_i + \eta^{(1)}\, r) + r\big) + \eta^{(2)}\, q\Big) + q\Big).$$

This expression directly extends Sprecher's idea of shifted sums of univariate transformations across two intermediary layers.

3.2.3 Three Hidden Layers ($L = 3$)

Now let $d_0 \to d_1 \to d_2 \to d_3$ be the layer dimensions, with the final layer summed over q. In the recursive style:

$$\mathbf{h}_r^{(1)} = \Phi^{(1)}\Big(\sum_{i=1}^{d_0} \lambda_{i,r}^{(1)}\, \phi^{(1)}(x_i + \eta^{(1)}\, r) + r\Big),$$

$$\mathbf{h}_s^{(2)} = \Phi^{(2)}\Big(\sum_{r=0}^{d_1-1} \lambda_{r,s}^{(2)}\, \phi^{(2)}(\mathbf{h}_r^{(1)} + \eta^{(2)}\, s) + s\Big),$$

$$\mathbf{h}_q^{(3)} = \Phi^{(3)}\Big(\sum_{s=0}^{d_2-1} \lambda_{s,q}^{(3)}\, \phi^{(3)}(\mathbf{h}_s^{(2)} + \eta^{(3)}\, q) + q\Big),$$

and

$$f(\mathbf{x}) = \sum_{q=0}^{d_3-1} \mathbf{h}_q^{(3)}.$$

The equivalent formulation with nested sums is

$$f(\mathbf{x}) = \sum_{q=0}^{d_3-1} \Phi^{(3)}\Big(\sum_{s=0}^{d_2-1} \lambda_{s,q}^{(3)}\, \phi^{(3)}\Big(\Phi^{(2)}\big(\sum_{r=0}^{d_1-1} \lambda_{r,s}^{(2)}\, \phi^{(2)}\Big(\Phi^{(1)}\big(\sum_{i=1}^{d_0} \lambda_{i,r}^{(1)}\, \phi^{(1)}(x_i + \eta^{(1)}\, r) + r\big) + \eta^{(2)}\, s\Big) + s\big) + \eta^{(3)}\, q\Big) + q\Big).$$

Remark 1. *These expansions show how each layer's output is a shift-and-sum transformation of the previous layer, with two splines $\phi^{(\ell)}, \Phi^{(\ell)}$. The final layer sums over its output index q to yield a single scalar. If $d_3 > 1$, one could keep the vector $\mathbf{h}^{(3)}$ as the final multi-channel output instead.*

4 Relation to Sprecher (1965) and Its Generalization

In his 1965 paper [1], David Sprecher proved a version of the Kolmogorov–Arnold theorem by writing

$$f(\mathbf{x}) = \sum_{q=0}^{2n} \Phi\Big(\sum_{p=1}^{n} \lambda_p \, \phi\big(x_p + \eta \, q\big) + q\Big),$$

for suitable choices of monotonic ϕ and continuous Φ. Our single-layer Sprecher Network with $d = n$ inputs and $d_{\text{out}} = 2n + 1$ outputs, *followed by a sum over* q, reproduces precisely that formula.

Going Deeper. While Sprecher's original construction used one layer, we can compose multiple shift-and-sum blocks to form a deeper architecture. Each block can reduce or transform the intermediate dimension, thereby creating new function classes that remain "Sprecher-like" but with greater representational flexibility.

5 Implementation Highlights

5.1 Trainable Splines

Each block has two spline modules:

- $\phi^{(\ell)}$, a monotonic piecewise-linear spline defined on $[0, 1]$ or a slightly wider interval, initialized so that it is increasing.
- $\Phi^{(\ell)}$, a general piecewise-linear spline defined on, say, $[-10, 12]$ or any sufficiently wide domain. This spline is not necessarily monotonic and can be initialized near the identity function.

Both spline types have many knots (e.g. 100–300), which goes well beyond the original theorem's proof, which only required continuity and monotonicity constraints for ϕ.

5.2 Shifts and Weights

Each block has a scalar $\eta^{(\ell)}$ controlling how the block's index q shifts its input coordinates, plus a matrix $\lambda_{i,q}^{(\ell)}$ mixing each input. These free parameters are updated via gradient-based optimizers to minimize a loss function such as

$$\text{(MSE on training data)} + \alpha \times \text{(penalty on spline second derivatives)},$$

where the penalty encourages smoother splines.

5.3 Parameter Counting

Because each block contains two splines (each with potentially hundreds of knot parameters), the total trainable parameters are typically dominated by the spline coefficients. If each $\phi^{(\ell)}$ has K_ϕ knots and each $\Phi^{(\ell)}$ has K_Φ knots, then each layer block has about $K_\phi + K_\Phi$ spline parameters, plus $d_{\ell-1} \, d_\ell$ weight parameters and one $\eta^{(\ell)}$. Summing over L blocks yields the total.

6 Universal Approximation Perspective

6.1 Sprecher-Type Theorems

If we restrict to one layer with $d_{\text{out}} = 2d + 1$, monotonic ϕ, and suitable constants (λ_i, η), then we reproduce the original Sprecher formula. From Sprecher's 1965 argument we know this single-layer network can approximate any continuous $f : [0, 1]^d \to \mathbb{R}$ to arbitrary accuracy, provided ϕ and Φ are sufficiently flexible.

6.2 Deeper Extensions

A natural conjecture is that stacking these blocks retains universal approximation capability while potentially requiring fewer summation nodes. Intuitively, multi-layer compositions can capture complex structure more efficiently. However, a complete theoretical treatment of depth vs. width for Sprecher Networks remains an active topic.

7 Empirical Demonstrations

In practice, we train Sprecher Networks on small data sets sampled from a target function f. Two classic examples are:

- **1D case:** A polynomial or sinusoidal function on $[0, 1]$. The network learns ϕ and Φ splines that approximate f very accurately, effectively acting like a learnable spline interpolant.
- **2D case:** A surface such as $f(x, y) = \exp(\sin(\pi x) + y^2)/7$. Once trained for sufficient epochs, good approximation is observed after $O(10^5)$ gradient updates. Layerwise spline plots offer an interpretable view of how the network transforms inputs.

8 Conclusion

We have introduced and motivated the design of *Sprecher Networks*, a modern, trainable extension of Sprecher's single-layer shift-and-sum formula for function approximation. By using two splines (ϕ and Φ) per layer block, along with careful index shifting and summation, we retain the theoretical grounding of the Kolmogorov–Arnold approach, thereby enabling deeper networks of potentially greater expressive power.

Further Related Work: Recent work in neural network design has re-examined the Kolmogorov–Arnold theorem from fresh perspectives. In particular, *Kolmogorov–Arnold Networks (KANs)* [4] propose learnable spline functions on edges instead of nodes, showing that such architectures can sometimes surpass MLPs in accuracy and interpretability for certain tasks. While KANs share the general KA motivation, the "Sprecher Network" approach here uses one monotonic and one non-monotonic spline *in each layer*, summing over the shifted inputs. It would be interesting in future work to compare these designs more thoroughly and unify them under the broader KA umbrella.

Future Directions: Potential research avenues include:

- *High-dimensional* spline adaptations (tensor-product or multi-dimensional ϕ, Φ).
- Incorporation of *residual* or *skip connections* for more powerful deep architectures.
- *Theoretical* comparisons of depth vs. width in Sprecher-style networks.

References

[1] Sprecher, D. A. (1965). "On the Structure of Continuous Functions of Several Variables," *Transactions of the American Mathematical Society*, **115**, 340–355.

[2] Kolmogorov, A. N. (1957). "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition," *Doklady Akademii Nauk SSSR*, **114**.

[3] Arnold, V. I. (1963). "On functions of three variables," *Doklady Akademii Nauk SSSR*, **48**.

[4] Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T.Y., Tegmark, M. (2025). "KAN: Kolmogorov-Arnold Networks," *ICLR 2025 (to appear)*. arXiv preprint arXiv:2404.19756.