

Introduction

Symbolic Regression (SR), which can be seen the discovery process of desired outputs that are inferred from particular inputs, can be used to “white-box” a “black-box” model. In the 17th century, Kepler developed his model for the motion of planets using years of astronomical observations. His data-driven trial and error approach, where he explicitly showed calculations that did not work, is still used in symbolic regression as a model discovery tool. The main goal of SR is to automatically derive or recover natural relations from data, without imposing restrictions on their form. In 2024 Liu et al. published a working paper, in which they propose a new type of Neural Network (NN), the so-called Kolmogorov-Arnold network (KAN), that, among other things can be used for symbolic regression. A comparison paper between KANs and Mult-layer Perceptron (MLP) found that KANs outperform MLPs for symbolic formula representation tasks.

In 1957, Williams was able to derive an eigenfunction expansion for isotropic linear-elastic materials, the so-called Williams series, which enabled the description of the asymptotic crack-field (stress and displacement fields) near the crack tip. At the German Aerospace Center (DLR), explainable AI is used, among other things, for detection of crack tips in materials.

The research question for this thesis was:

Can the basic building blocks of the Williams series be reliably recovered from data with the use of KANs?

Background

Kolmogorov-Arnold Networks

As an alternative to MLPs, KAN utilize the Kolmogorov-Arnold representation theorem to express multivariate functions. This leads to learnable 1D functions on the edges. Liu et al. present the differences as such:

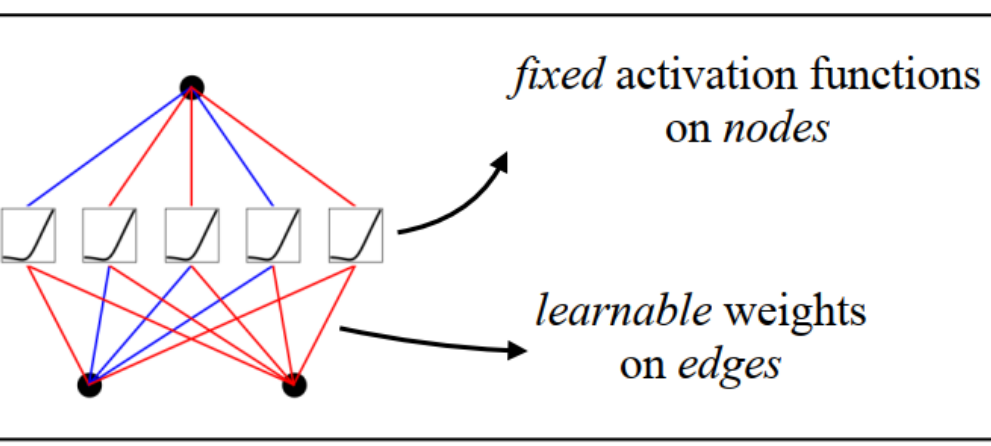
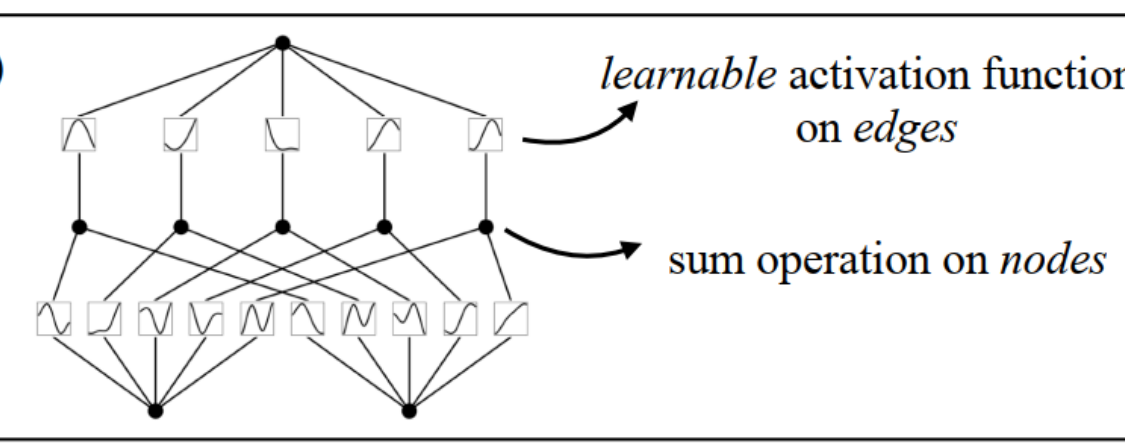
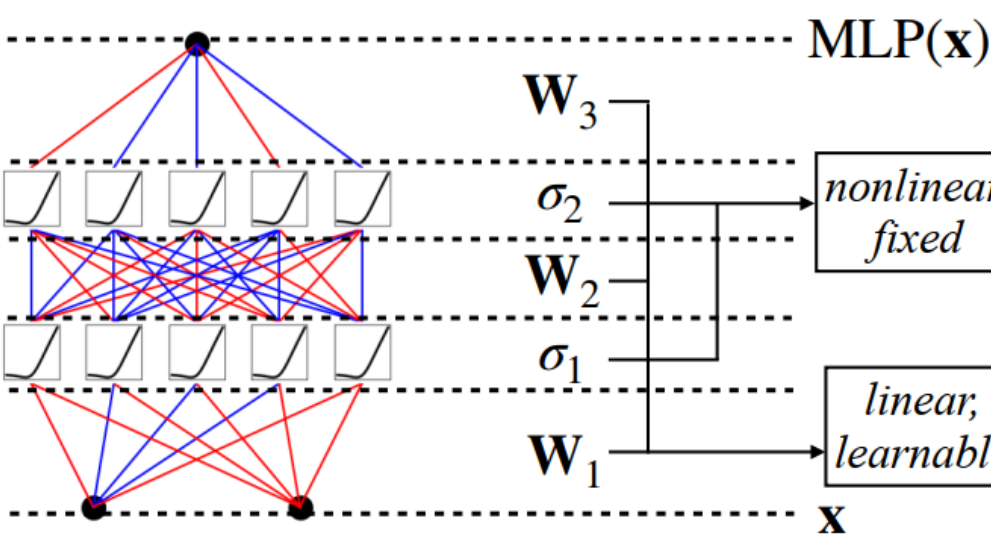
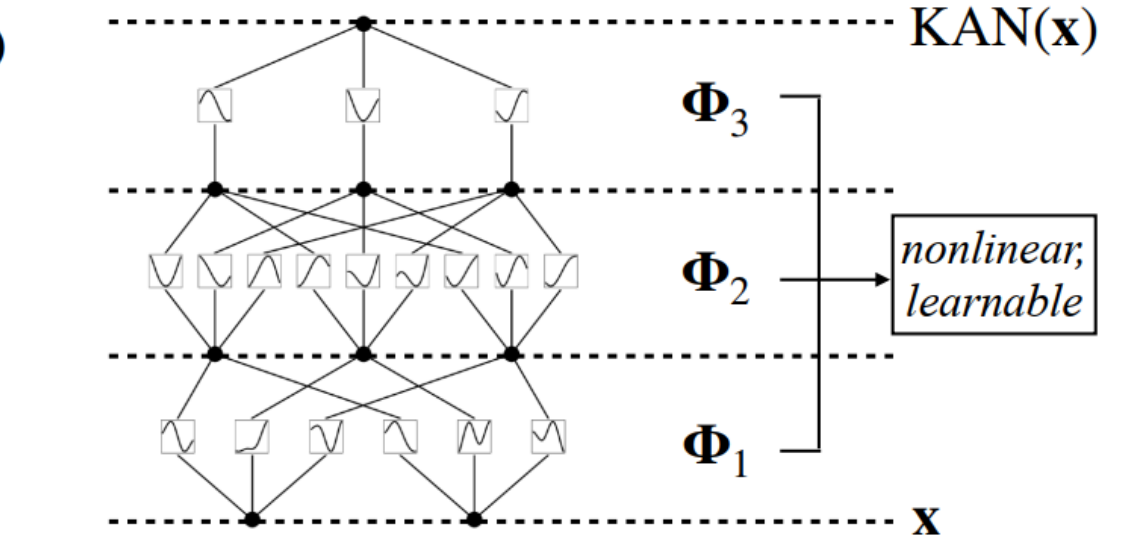
Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	Universal Approximation Theorem	Kolmogorov-Arnold Representation Theorem
Formula (Shallow)	$f(\mathbf{x}) \approx \sum_{i=1}^{N(e)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$	$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^n \phi_{q,p}(x_p) \right)$
Model (Shallow)	(a) 	(b) 
Formula (Deep)	$\text{MLP}(\mathbf{x}) = (\mathbf{W}_3 \circ \sigma_2 \circ \mathbf{W}_2 \circ \sigma_1 \circ \mathbf{W}_1)(\mathbf{x})$	$\text{KAN}(\mathbf{x}) = (\Phi_3 \circ \Phi_2 \circ \Phi_1)(\mathbf{x})$
Model (Deep)	(c) 	(d) 

Figure 1: comparison MLPs vs KANs, source: Liu et al. (2024)

Methodology

Datasets: Dataset were simulated and normalized using the z-Score method. Most commonly, the function $f(r, \phi) = 3\sqrt{r} \sin \frac{\phi}{2}$ was used.

Modelgeneration: Models were initialized using the **pykan**-package provided by the authors of the paper.

Hyperparameteroptimization: Hyperparameters were optimized using the Optuna framework.

Pareto-front: Examining the pareto-front for depth/width of the NN and complexity of symbolic expression in the library to analyze performance and usefulness of weighted complexity for certain functions in libraries.

Results

Hyperparameteroptimization

The regularization term in the **pykan**-package is given as:

$$\mathcal{L}_{\text{reg}} = \lambda_{\text{reg}} (\lambda_{L1} \mathcal{L}_{L1} + \lambda_{\text{entropy}} \mathcal{L}_{\text{entropy}} + \lambda_{\text{coef}} \mathcal{L}_{\text{coef}} + \lambda_{\text{coefdifff}} \mathcal{L}_{\text{coefdifff}}) \cdot \quad (1)$$

In the tests λ_{coef} and $\lambda_{\text{coefdifff}}$ had negligible impact, and they were not considered in the standard implementation of the **pykan**-package as well as in the published paper. That is why they were disregarded going forward in this thesis as well.

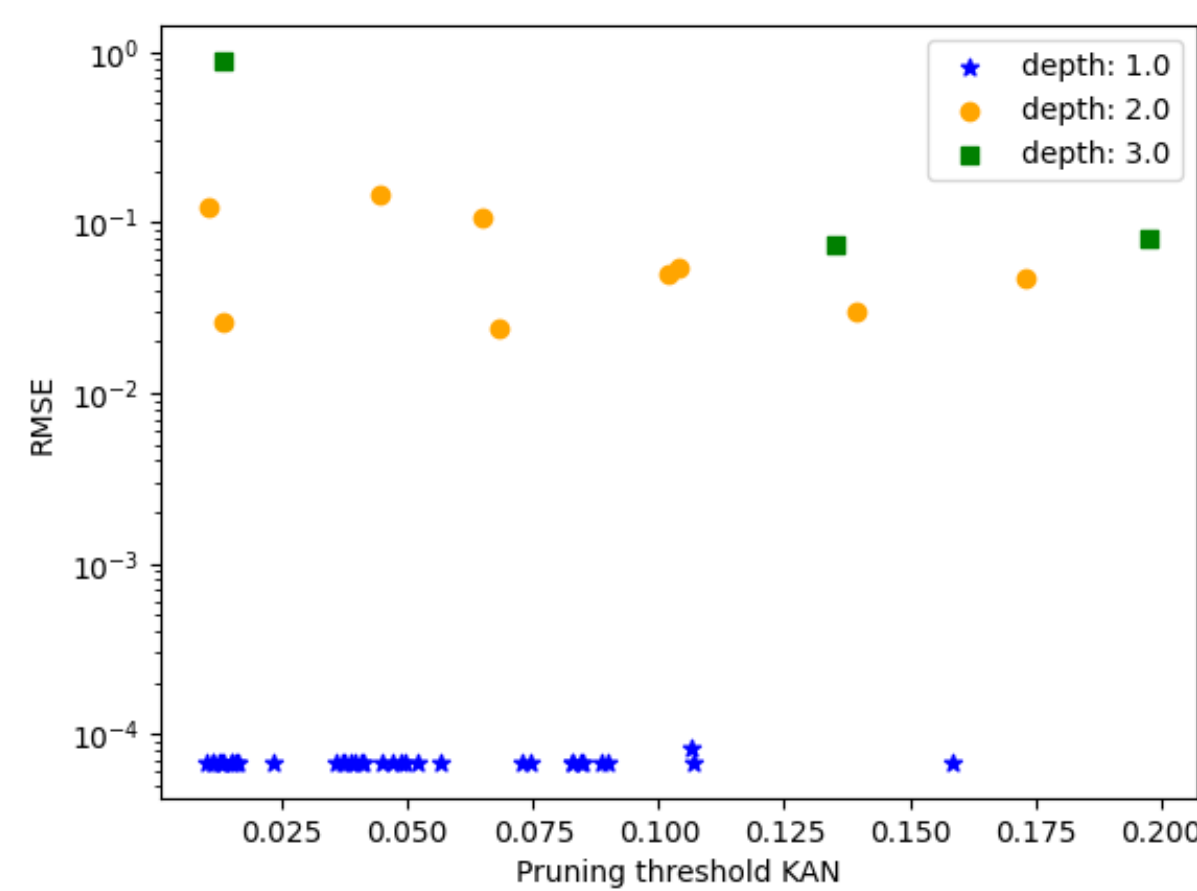


Figure 2: distribution of loss in regards to pruning threshold, sorted by depth of the NN

Paretofront

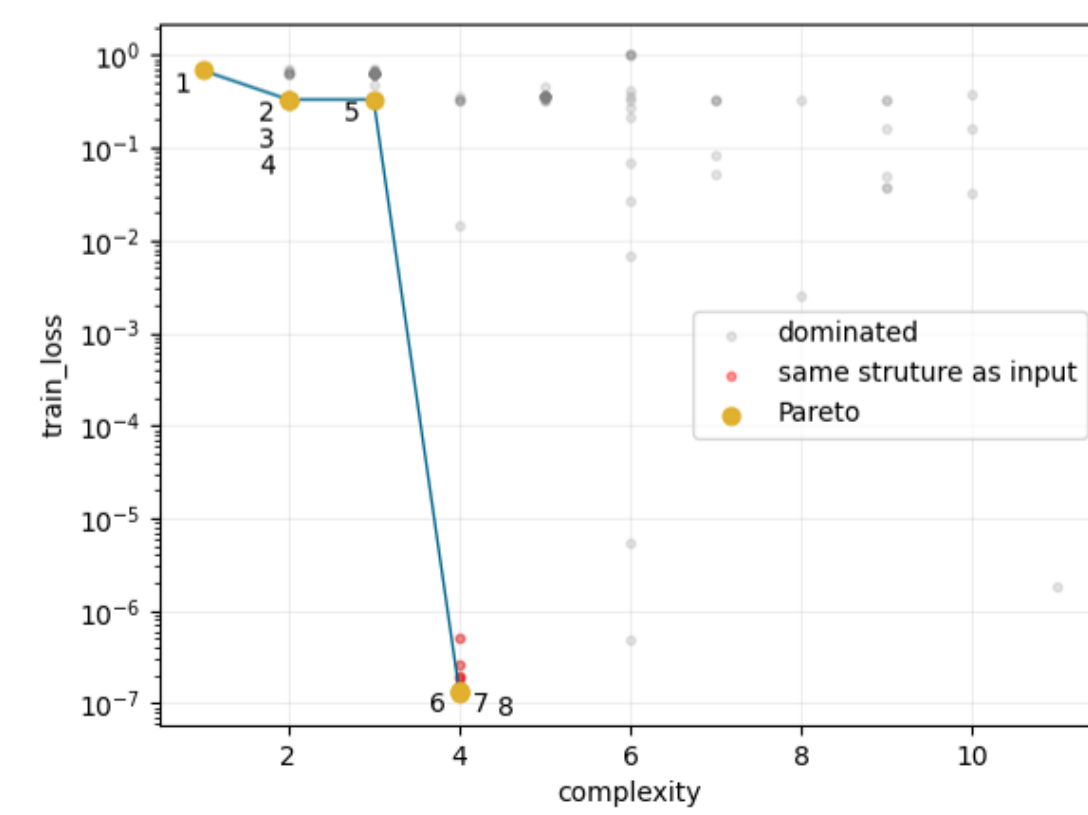


Figure 3: Paretofront of weighted library complexities

Successful trainings are more dependent on the architecture of the neural network than the pruning threshold. It is more difficult to successfully train a KAN with more layers than optimal.

Only 151 out of 350 models trained successfully to symbolic recovery. Of those 151 only 12 reached a result that resembles the structure of the input data. That means only about 3.5% of all training ended satisfactory. Notably, higher complexities did not achieve better training losses than sparser functions.

RMSE	kan_width	reduced expression	cfunc
$1.27 \cdot 10^{-7}$	[[2,0],[5,3],[1,0]]	$\sqrt{r} \sin(\varphi)$	4
$1.38 \cdot 10^{-7}$	[[2,0],[5,4],[1,0]]	$\sqrt{r} \cos(\varphi)$	4
$1.38 \cdot 10^{-7}$	[[2,0],[0,3],[1,0]]	$\sqrt{r} \sin(\varphi)$	4
$1.39 \cdot 10^{-7}$	[[2,0],[3,5],[1,0]]	$\sqrt{r} \cos(\varphi)$	4
$1.42 \cdot 10^{-7}$	[[2,0],[5,1],[1,0]]	$\sqrt{r} \sin(\varphi)$	4
$1.46 \cdot 10^{-7}$	[[2,0],[2,4],[1,0]]	$\sqrt{r} \sin(\varphi)$	4
$1.52 \cdot 10^{-7}$	[[2,0],[2,3],[2,3],[1,0]]	$\sqrt{r} \cos(\varphi)$	4

Table 1: top 7 models with weighted library

RMSE	kan_width	reduced expression	cfunc
$1.56 \cdot 10^{-4}$	[[2,0],[0,2],[1,0]]	$\arccos(\sqrt{r} \sin(\varphi))$	4
$4.28 \cdot 10^{-4}$	[[2,0],[0,2],[0,2],[1,0]]	$\tan(\sqrt{r} \sin(\varphi))$	7
$5.49 \cdot 10^{-3}$	[[2,0],[2,1],[2,1],[1,0]]	$\text{atanh}(\exp(\dots))$	17
$6.64 \cdot 10^{-3}$	[[2,0],[1,5],[1,5],[1,0]]	$\arccos(\dots)$	22
$1.44 \cdot 10^{-2}$	[[2,0],[5,5],[5,5],[1,0]]	$\tan(\dots \sin(\dots))$	13
$2.40 \cdot 10^{-2}$	[[2,0],[3,2],[3,2],[1,0]]	$\exp(\dots)$	33
$2.95 \cdot 10^{-2}$	[[2,0],[3,0],[1,0]]	$\sin(\text{atan}(\dots))$	9

Table 2: top 7 models with constant library

A weighted library yielded better success. Using a library where all function have the same complexity weighting did not result in any successful training. The target reduced expressions were $\sqrt{r} \sin(\varphi)$ or $\sqrt{r} \cos(\varphi)$.

Conclusion

KANs can be used to successfully find the building blocks of complex formulas like the Williams series. However, calibrating them for general symbolic regression, where prior knowledge about the data distribution is not available seems difficult with the approach used in this thesis. The training is very sensitive to hyperparameter and even the seed can have an important role in determining successful or unsuccessful training. That means using a singular model and pruning it to the desired structure might be to inconsistent for general use. Other research has been put into progressively expanding the model until a certain threshold is reached. That might prove to be a more viable approach than the one used in this thesis.

Sources

- [BG25] Marco Bühler and Gonzalo Guillén-Gosálbez. *KAN-SR: A Kolmogorov-Arnold Network Guided Symbolic Regression Framework*. Sept. 2025. DOI: [10.48550/arXiv.2509.10089](https://doi.org/10.48550/arXiv.2509.10089).
- [Kun13] Meinhard Kuna. *Finite Elements in Fracture Mechanics: Theory - Numerics - Applications*. Vol. 201. Solid Mechanics and Its Applications. Dordrecht: Springer Netherlands, 2013. ISBN: 978-94-007-6679-2. DOI: [10.1007/978-94-007-6680-8](https://doi.org/10.1007/978-94-007-6680-8).
- [Liu+25a] Ziming Liu et al. *KAN: Kolmogorov-Arnold Networks*. 2025. arXiv: [2404.19756](https://arxiv.org/abs/2404.19756) [cs.LG]. URL: <https://arxiv.org/abs/2404.19756>.
- [Liu+25b] Ziming Liu et al. “Kolmogorov-Arnold Networks Meet Science”. In: *Physical Review X* 15.4 (Dec. 2025), p. 041051. ISSN: 2160-3308. DOI: [10.1103/4t7t-v191](https://doi.org/10.1103/4t7t-v191).