LoKAN: Back prop-Free Node-Local Functional-Edge Learning for Equation Discovery

Anonymous authors
Paper under double-blind review

Abstract

Kolmogorov-Arnold network (KAN) model nonlinear relationships through functions on edges, but their many learnable edge parameters typically require repeated updates through backpropagation, making training costly. Such iterative optimization does not directly exploit the linear structure of edge coefficients under fixed basis functions to improve training efficiency. Fitting edges independently, in turn, ignores the coupling among incoming edges of the same node and generally cannot guarantee the joint optimum for a fixed support. To address these limitations, we propose LoKAN, a single-layer functional-edge model for equation discovery. Given task-specified candidate features, LoKAN constructs anchored piecewise-linear spline edges, jointly solves a ridge-regression system over the active incoming edges of each output node, iteratively prunes edges by their normalized contributions, and refits on the final support, without backpropagating through edge coefficients. In addition, LoKAN uses exact Gram-component partitioning to perform component-wise solves, reducing matrix storage requirements. Separate efficiency and scaling experiments show, respectively, that LoKAN achieves shorter per-fit training times than the compared backpropagation-trained KAN and spline group lasso, and that component-wise solves reduce Gram-matrix storage. Across the experiments, LoKAN demonstrates strong predictive performance and effectively recovers explicit candidate terms.

1 Introduction

Kolmogorov-Arnold network (KAN) offer a perspective on nonlinear function modeling that differs from conventional neural networks. They place learnable univariate functions on network edges and sum incoming contributions at each node. Consequently, nonlinearities are not represented solely by node activation functions, and the effects of input features on the output can be inspected as functions. This representation combines flexible function approximation with a degree of structural interpretability and has been used in function fitting, scientific computing, partial differential equations, and equation discovery. In equation discovery, when input features correspond to well-defined variables or candidate terms, edge functions can also provide direct evidence for analyzing how those features affect the target.

An inspectable edge representation, however, does not by itself ensure efficient and reliable parameter estimation. Different incoming edges of the same output node jointly explain one target, while candidate features such as raw variables, predefined interaction terms, derivatives, or temporal lags can be strongly correlated. With fixed basis functions, solving for edges separately and discarding their cross-Gram blocks changes the original node-level joint fitting problem; signal may then be allocated redundantly or unstably among correlated edges. Moreover, common KAN training procedures repeatedly update many edge functions and basis coefficients through backpropagation. As the numbers of edges, basis functions, or outputs increase, gradient computation and repeated parameter updates can incur substantial training costs.

We therefore propose LoKAN, a single-layer functional-edge model that estimates incoming edges jointly at each output node. Its task-specified candidate map may contain raw coordinates or interpretable polynomial terms, interactions, spatial derivatives, and temporal lags. LoKAN constructs a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

fixed anchored piecewise-linear spline basis for each candidate feature, making every edge function linear in its coefficients once the basis is fixed. For each output node, all active incoming edges are estimated in one regularized linear system. The active set is then progressively reduced according to the normalized contribution of each edge to the fitted output, followed by a refit on the final support. Different output nodes may select different edge sets, while correlated edges within a node are always estimated under the same objective.

This design retains cross-Gram terms between edges, allowing their coefficients to adjust jointly and yielding the unique minimizer of the node objective for a fixed structure and positive ridge scale. Since coefficient estimation with fixed bases reduces to solving a linear system, training requires neither a backpropagation graph for the edge coefficients nor repeated gradient updates. The expressive range of the model is determined jointly by the candidate map and spline resolution. With raw coordinates as candidates, the single-layer functional edges capture additive nonlinear effects of individual variables; explicit multivariate interactions require corresponding combination features in the candidate map. When candidate features represent the equation terms of interest and two-knot linear edges are used, selected edges and their coefficients after reversing standardization can further provide term-level equation estimates. LoKAN thus combines inspectable functional representations, joint estimation of correlated edges, and progressive structure selection in a backpropagation-free approach to equation discovery.

Our main contributions are as follows:

- We propose LoKAN, a single-layer functional-edge model designed for equation discovery from candidate features. Anchored splines represent feature effects, while joint node-level estimation, contribution-based pruning, and a final refit determine the functional structure of each output.
- For fixed candidate representations and supports, LoKAN jointly solves for the ridge-regression coefficients of each output. We characterize this joint solution, derive the cross-Gram matrix conditions for exact partitioning, and establish when aggregating consistent sufficient statistics is equivalent to a centralized solve.
- LoKAN provides a backpropagation-free training procedure and clarifies the candidate-representation conditions for numerical function fitting and equation-term recovery. Experiments on dynamical systems, candidate-term identification, and function-fitting tasks evaluate the model and the role of joint estimation.

2 Related Work

2.1 KANs and Edge-Function Models

KANs replace scalar connection weights with learnable univariate functions on edges and sum incoming contributions at each node (Liu et al., 2025b). Their motivation is related to classical representation and approximation results, although those results do not imply that a particular finite architecture is easy to train or yields an interpretable equation (Hornik et al., 1989; Leshno et al., 1993). KAN-based scientific modeling has since explored symbolic compilation and operator-aware representations (Liu et al., 2025a; Panahi et al., 2025). The KAN model-discovery study is especially close to our setting: it uses functional edges to represent nonlinear dynamics without requiring a sparse equation library. LoKAN instead fixes a task-specified candidate map and focuses on exact joint estimation and support selection for its single-layer edges.

Published KAN variants also examine alternative edge bases, including sinusoidal functions (Reinhardt et al., 2025). These works primarily study representation and gradient-trained architectures; LoKAN uses anchored piecewise-linear spline edges and estimates all active edges incident to an output jointly before contribution-based pruning and refitting.

2.2 Backpropagation-Free KAN Learning

Backpropagation-free learning has a longer history in random-feature and analytic-output-layer methods. Extreme learning machines fix or randomly choose hidden units and solve output coefficients by least squares, with ridge regularization used for noisy data (Huang et al., 2006). Such

methods establish the value of replacing iterative gradient updates with a linear solve when the feature map is fixed. LoKAN differs by learning edge coefficients jointly within each output node and selecting a sparse edge support.

Local-learning proposals provide another alternative to reverse-mode differentiation. Predictive-coding networks can approximate or exactly implement backpropagation under particular inference dynamics, while equilibrium propagation derives local updates from nearby energy states (Whittington & Bogacz, 2017; Song et al., 2020; Salvatori et al., 2023; Scellier & Bengio, 2017). LoKAN's setting is narrower: in a single layer with a separable output loss and fixed bases, its coefficient system is solved directly without either reverse-mode differentiation or iterative latent-state inference.

2.3 Joint Estimation and Structured Selection

Although KAN variants parameterize functions edge-wise, a node still fits a shared target, so correlated edge designs generate cross-Gram terms that belong to the joint normal equations. This is the familiar nonorthogonal-regression setting for which ridge regularization was developed, and additive-model backfitting can be understood as an iterative solution of coupled normal equations (Hoerl & Kennard, 1970; Buja et al., 1989; Stone, 1985).

Joint estimation is also related to structured selection and multi-response learning. Group lasso incorporates predefined feature blocks into sparse selection, sparse additive models combine smooth components with variable selection, and multi-task feature learning can share representations across responses (Yuan & Lin, 2006; Ravikumar et al., 2009; Argyriou et al., 2008). The lasso and elastic net provide complementary coefficient-level sparsity and correlated-feature regularization (Tibshirani, 1996; Zou & Hastie, 2005). LoKAN treats each candidate edge's spline coefficients as a group, retains cross-edge Gram blocks during estimation, and selects output-specific supports by normalized contribution followed by a fixed-support refit.

(Yuan & Lin, 2006; Tibshirani, 1996). This separates the coefficient solve from the chosen contribution-based selection rule.

2.4 Fixed Libraries and Adaptive Representations for Equation Discovery

Equation discovery seeks interpretable equations for static relationships, dynamical systems, and partial differential equations. SINDy selects terms from a candidate library, PDE-FIND extends sparse identification to spatial derivatives, and SINDYc adds known controls or forcing inputs (Brunton et al., 2016a; Rudy et al., 2017; Brunton et al., 2016b). Extensions address noisy derivatives, implicit or rational dynamics, and uncertainty quantification through weak formulations, SINDy-PI, and sparse Bayesian regression (Messenger & Bortz, 2021b;a; Kaheman et al., 2020; Zhang & Lin, 2018). SR3 provides a unified optimization framework for sparse regularizers (Champion et al., 2020). These methods can recover explicit support and coefficients when the library contains the relevant terms; a sparse solver cannot introduce a missing interaction or functional form.

Symbolic regression searches expression structures, from genetic-programming discovery to physics-guided methods such as AI Feynman; equation-learning networks encode arithmetic operators, while PDE-Net and DeepMoD combine learned representations with equation structure or sparse selection (Schmidt & Lipson, 2009; Udrescu & Tegmark, 2020; Sahoo et al., 2018; Long et al., 2019; Both et al., 2021). Physics-informed neural networks offer a complementary route for solving and identifying differential equations by embedding residual constraints in a learned surrogate (Raissi et al., 2019). LoKAN occupies an intermediate position: it fits anchored univariate spline edges over supplied candidate features and jointly solves the active edges of each output. Raw inputs yield additive numerical functions, while explicit equation-term candidates with two-knot linear edges permit term-level estimates after reversing standardization.

3 Method

LoKAN organizes equation discovery into three connected steps: candidate representation, node-level estimation, and structure selection. Candidate features specify interpretable variables or terms for the incoming edges, while fixed spline bases allow each edge to represent a nonlinear effect. Given an active edge set, all coefficients entering the same output node are estimated by a joint

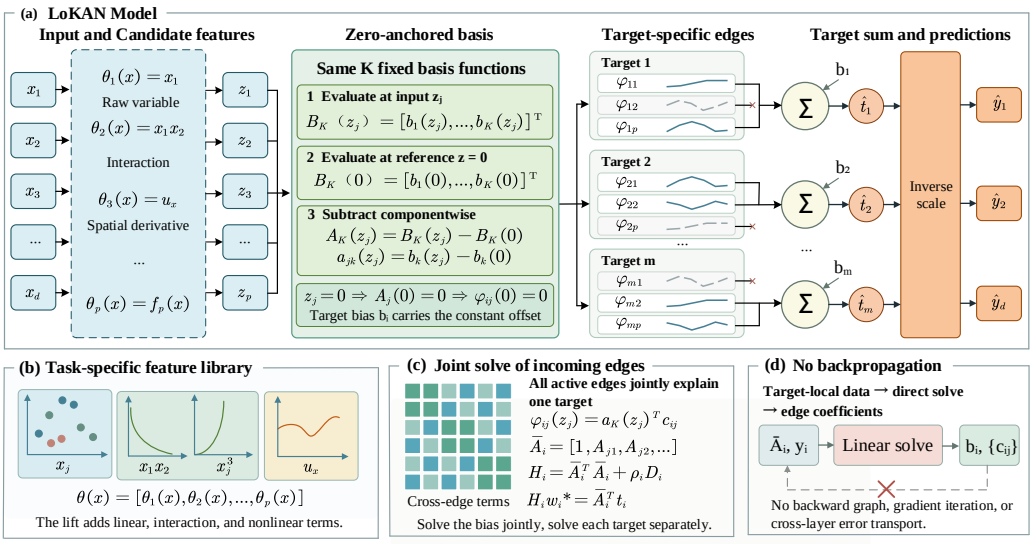


Figure 1: Overview of LoKAN. (a) Model topology and functional edges. (b) Candidate feature construction. (c) Joint fitting and edge pruning for each target node. (d) Direct linear solving without backpropagation.

ridge solve, retaining the allocation of signal among correlated edges. The active set is then updated according to the fitted contribution of each edge, followed by a final refit. The single-layer architecture separates parameters across output nodes while preserving coupling among the edges incident to each node. This combines flexible functional representation with an exact coefficient solve for each fixed structure.

3.1 Regularization, candidate features, and single-layer functional edges

Let $\{(x_r, y_r)\}_{r=1}^{N}$ be training samples, where $x_r \in \mathbb{R}^d$ and $y_r \in \mathbb{R}^m$. A task-specified candidate map $\theta(x) = [\theta_1(x), \ldots, \theta_p(x)]^\top$ may contain raw coordinates for additive nonlinear modeling, or physically meaningful polynomial terms, interactions, spatial derivatives, or temporal lags. The candidate map determines the range of equation terms under consideration; the edge functions learn how each supplied feature affects a target. The estimation and selection procedure below applies to all of these candidate types.

Feature and target means and scales are estimated from fitting data alone. Define

$$z_{rj} = \frac{\theta_j(x_r) - \mu_j^\theta}{\sigma_j^\theta}, \qquad t_{ri} = \frac{y_{ri} - \mu_i^y}{\sigma_i^y}, \tag{1}$$

and write $t_i = (t_{1i}, \ldots, t_{Ni})^\top$ for standardized target i. A scale that is nearly zero is set to one for numerical stability. Validation, test, and inference samples use the statistics determined during fitting.

For $K > 2$, let $B_K(z) \in \mathbb{R}^K$ be a fixed piecewise-linear hat basis with knots uniformly spaced over $[-3, 3]$. We use the following anchored basis to accommodate both linear candidate terms and nonlinear edge functions:

$$a_K(z) = \begin{cases} (-z/2,\ z/2)^\top, & K = 2, \\ B_K(\mathrm{clip}(z, -3, 3)) - B_K(0), & K > 2, \end{cases} \qquad \phi_{ij}(z) = a_K(z)^\top c_{ij}, \quad c_{ij} \in \mathbb{R}^K. \tag{2}$$

The edge ϕ_{ij} connects candidate feature j to output node i. Both cases satisfy $\phi_{ij}(0) = 0$, where zero refers to the standardized feature; a node intercept absorbs the constant offset. The multi-knot basis gives a local nonlinear edge over a bounded interval, whereas $K = 2$ yields a linear edge on the real line and is suited to coefficient estimation for explicit candidate terms.

Let $S_i \subseteq \{1, \dots, p\}$ denote the active incoming edges of output i. Its standardized prediction is

$$\widehat{t}_i(x) = b_i + \sum_{j \in S_i} \phi_{ij} \left(\frac{\theta_j(x) - \mu_j^\theta}{\sigma_j^\theta} \right). \tag{3}$$

Outputs use the same candidate-map definition and basis construction, but have separate supports S_i, intercepts b_i, and edge coefficients c_{ij}. For fixed S_i, let $A_j \in \mathbb{R}^{N \times K}$ have row r equal to $a_K(z_{rj})^\top$. Concatenate the active blocks in a fixed order as $A_i = [A_j]_{j \in S_i}$ and set $\bar{A}_i = [\mathbf{1}, A_i]$ and $w_i = (b_i, c_i^\top)^\top$, where c_i concatenates the corresponding c_{ij} in the same order. Predictions on the fitting data are then $\bar{A}_i w_i$.

To stabilize correlated or linearly dependent basis columns, we apply ridge regularization to edge coefficients while leaving the intercept unpenalized. With $q_i = 1 + K|S_i|$, define

$$\rho_i(S_i) = \lambda \left(1 + \frac{\operatorname{tr}(\bar{A}_i^\top \bar{A}_i)}{q_i} \right), \qquad D_i = \operatorname{diag}(0, I_{K|S_i|}), \qquad \lambda > 0. \tag{4}$$

The mean diagonal of the Gram matrix adjusts the penalty to the scale of the current design as edges are removed, while the additive one keeps the penalty positive even for nearly inactive basis columns. Anchoring concentrates constant offsets in the unpenalized intercept, so the ridge term acts only on functional-edge coefficients. The fixed-support objective is

$$J_i(w_i; S_i) = \frac{1}{2} \|t_i - \bar{A}_i w_i\|_2^2 + \frac{\rho_i(S_i)}{2} w_i^\top D_i w_i. \tag{5}$$

Regularization stabilizes coefficient estimation for a given support; contribution scoring and pruning in Section 3.3 select the edge set. Thus, each coefficient-estimation step is a well-defined convex problem.

3.2 Joint node-level estimation without backpropagation

For fixed S_i, Equation equation 5 is quadratic in w_i. Setting its gradient to zero gives

$$\nabla_{w_i} J_i = \bar{A}_i^\top (\bar{A}_i w_i - t_i) + \rho_i(S_i) D_i w_i = 0,$$

and hence

$$H_i w_i^\star = \bar{A}_i^\top t_i, \qquad H_i = \bar{A}_i^\top \bar{A}_i + \rho_i(S_i) D_i. \tag{6}$$

For any nonzero $v = (v_b, v_c^\top)^\top$, $v^\top H_i v = \|\bar{A}_i v\|_2^2 + \rho_i(S_i)\|v_c\|_2^2 > 0$: the second term is positive if $v_c \neq 0$, while the first equals $N v_b^2 > 0$ if $v_c = 0$ and $v_b \neq 0$. Thus, when $N > 0$ and $\lambda > 0$, H_i is positive definite and Equation equation 6 gives the unique global minimizer for the fixed representation, support, and regularization. The solution is obtained by solving the linear system without explicitly forming H_i^{-1}. Moreover, completing the square yields, for any alternative coefficient vector w_i,

$$J_i(w_i; S_i) - J_i(w_i^\star; S_i) = \frac{1}{2} (w_i - w_i^\star)^\top H_i (w_i - w_i^\star) \geq 0, \tag{7}$$

with equality only at $w_i = w_i^\star$.

The need for a joint solve follows from the summation at a KAN node. Eliminating the intercept with $M = I_N - \mathbf{1}\mathbf{1}^\top / N$ makes Equation equation 5 equivalent, as a function of c_i, to

$$\frac{1}{2} \|M t_i - M A_i c_i\|_2^2 + \frac{\rho_i(S_i)}{2} \|c_i\|_2^2. \tag{8}$$

In this single-layer, fixed-basis setting, all incoming edge functions are summed before comparison with the same target. Their parameters are stored separately, but their estimates need not be independent. Partitioning A_i by edge and defining $G_{jk} = A_j^\top M A_k$ gives

$$c_i^\top A_i^\top M A_i c_i = \sum_{j \in S_i} c_{ij}^\top G_{jj} c_{ij} + 2 \sum_{\substack{j<k \\ j,k \in S_i}} c_{ij}^\top G_{jk} c_{ik}.$$

The second sum expresses how distinct edges jointly explain one target. Correlated candidate features produce nonzero cross-Gram blocks; independent edge solves discard these blocks and can

allocate the same signal to multiple edges. The joint solve retains the original node objective and reaches its fixed-support optimum. Appendix A.2 gives a two-edge example.

Across outputs, by contrast, no trainable coefficients are shared. For fixed supports, the multi-output objective therefore decomposes exactly as $\mathcal{J}(\{w_i\}, \{S_i\}) = \sum_{i=1}^{m} J_i(w_i; S_i)$. The output node is consequently a valid independent optimization unit, even when all of its incoming edges must be solved together.

This node-local structure also explains the absence of backpropagation. For residual $e_i = t_i - \bar{A}_i w_i$, the exact derivative for an active edge is

$$\nabla_{c_{ij}} J_i = -A_j^\top e_i + \rho_i(S_i) c_{ij}, \qquad \frac{\partial J_i}{\partial b_i} = -\mathbf{1}^\top e_i. \tag{9}$$

These derivatives involve only the node residual and the corresponding input basis. LoKAN satisfies the first-order conditions of all edges entering a node simultaneously through Equation equation 6, rather than repeatedly adjusting edge parameters through reverse-mode differentiation. Output objectives are separable, so each node forms a system from its own active edges; pruning may give different nodes different supports and ridge scales.

3.3 Contribution pruning and final fixed-support refit

The joint solve determines coefficients for a given support but does not itself select edges. For each output, LoKAN starts from $S_i^{(0)} = \{1, \ldots, p\}$. At round ℓ, it solves Equation equation 6 on $S_i^{(\ell)}$ and evaluates the fitted contribution of each active edge:

$$v_{rij}^{(\ell)} = a_K(z_{rj})^\top c_{ij}^{(\ell)}, \qquad s_{ij}^{(\ell)} = \frac{\left[N^{-1} \sum_{r=1}^{N} \left(v_{rij}^{(\ell)}\right)^2\right]^{1/2}}{\max\{\mathrm{std}(t_i), 10^{-8}\}}. \tag{10}$$

The norm of a coefficient vector depends on basis scaling and parameterization, whereas Equation equation 10 measures the edge's realized output amplitude on the fitting samples. Normalization by target standard deviation gives the threshold a relative-contribution interpretation.

Given threshold τ, the next support retains active edges with $s_{ij}^{(\ell)} \geq \tau$. If this rule removes every edge, the highest-scoring edge is retained. The supports therefore form a nested decreasing sequence. Pruning ends when the support stabilizes or a prescribed budget of R rounds is reached; the experimental configuration uses $R = 4$. With the final support S_i^{final} fixed, the ridge scale in Equation equation 4 is recomputed for its design matrix and Equation equation 6 is solved once more. This ensures that reported and inferred coefficients correspond to the retained edges.

The knot count K, threshold τ, and base ridge parameter λ are chosen using task configurations and validation data. Final test evaluation follows frozen selection. Inference applies the selected candidate map, fitting-set standardization, support, and coefficients in Equation equation 3, then returns to original units via $\widehat{y}_i = \mu_i^y + \sigma_i^y \widehat{t}_i$. No solve or pruning is required at inference time.

3.4 Computational cost, exact partitions, and expressivity

Constructing and caching the candidate basis costs $O(NpK)$ time and space. For a solve with $q_i = 1 + K|S_i|$ columns, forming $\bar{A}_i^\top \bar{A}_i$ costs $O(Nq_i^2)$ time; a dense solve costs $O(q_i^3)$ time and $O(q_i^2)$ matrix space. Total training cost also depends on the active widths across pruning rounds and on the number of configurations examined during validation. Shrinking $|S_i|$ can lower later solve and inference costs, while inference only evaluates and sums retained edges. Training cost no longer depends on a prescribed number of gradient epochs, although its practical benefit depends on sample size, candidate width, and hardware.

Fix S_i and the positive ridge scale $\rho_i(S_i)$ computed for the full node. Construct a graph on the active edges, joining j and k whenever $G_{jk} \neq 0$. Its connected components form the unique finest edge-group partition whose separate solves reproduce the joint solution for every linear-system right-hand side. Any union of whole components is also exact; splitting a component fails for at least one right-hand side. Consequently, edge-wise independent solves are exact for every right-hand side if and only if all cross-edge Gram blocks vanish. Appendix A.2 proves the claim. LoKAN uses the full output node as its basic solve unit; the proposition characterizes when a finer exact unit exists.

Along the sample dimension, the fixed-design statistics $\bar{A}_i^\top \bar{A}_i$ and $\bar{A}_i^\top t_i$ are sums of the corresponding statistics over disjoint data shards. With common standardization and a ridge scale computed from the aggregated statistics, the resulting system equals the centralized one (Appendix A.4). Output-node separation, exact edge-group partitions, and data-shard aggregation therefore offer distinct levels of potential parallel computation.

The candidate map and spline resolution jointly determine expressivity. Raw-coordinate candidates with multi-knot edges represent additive nonlinear effects. Adding scientifically meaningful combinations such as x_1x_2 to θ permits the same framework to estimate interactions; with raw coordinates alone, the single-layer function class remains additive. When each candidate corresponds to an explicit equation term and $K = 2$, define $\gamma_{ij} = (c_{ij,2} - c_{ij,1})/2$. Equations equation 1-equation 3 then yield $\widehat{y}_i = \beta_{i0} + \sum_{j \in S_i} \beta_{ij}\theta_j(x)$, where

$$\beta_{ij} = \frac{\sigma_i^y}{\sigma_j^\theta}\gamma_{ij}, \qquad \beta_{i0} = \mu_i^y + \sigma_i^y b_i - \sum_{j \in S_i} \beta_{ij}\mu_j^\theta. \tag{11}$$

Thus, active linear edges specify a candidate-term selection and Equation equation 11 gives coefficients in original units. Appendix A.3 derives the conversion and treats constant features. Multi-knot edges instead describe univariate numerical functions of their candidates. The choice of representation therefore sets the balance between explicit term recovery and flexible nonlinear modeling while leaving the node-local estimation mechanism intact.

4 Experiments

We compare LoKAN with SINDy-SR3, Spline Group Lasso, BP-KAN, HKAN, and MatrixKAN. SINDy-SR3 represents sparse equation discovery over a candidate library, whereas Spline Group Lasso iteratively fits spline edge functions with group sparsity. BP-KAN provides a backpropagation-trained KAN baseline; HKAN and MatrixKAN represent a hierarchical backpropagation-free KAN and a matrix-based spline KAN, respectively.

Our experiments cover four settings: noisy dynamical equation discovery and real-system prediction; equation fitting with correlated, weak, or missing candidate terms; nonlinear function fitting and representation limits; and an ablation of joint estimation. All tables report test normalized root mean squared error (NRMSE), defined as RMSE divided by the standard deviation of the test targets; lower is better. For every model and task, we report the lowest test NRMSE in the recorded runs. One-step and recursive results are selected separately. Values requiring more than five decimal places use compact scientific notation.

4.1 Dynamical Equation Discovery and Real-System Evaluation

We study Lotka-Volterra, Duffing, Van der Pol, Lorenz-63, FitzHugh-Nagumo, and Rössler, splitting each system by complete trajectory into training, validation, and test sets. In the *direct* setting, the training derivatives are corrupted by noise at 5% of their standard deviation. In the *state* setting, the measured states are corrupted by 1% noise before the training derivatives are estimated with Savitzky-Golay smoothing (Savitzky & Golay, 1964). We additionally evaluate acceleration prediction on an eight-node N-body system. For measured-data evaluation, Silverbox and Cascaded Tanks use, respectively, 2/3 and 6/6 output/input lags to predict the next output; these are established nonlinear system-identification benchmarks (Wigren & Schoukens, 2013; Schoukens et al., 2016). Recursive evaluation feeds predictions back after an initialization window for up to 2,000 steps.

As shown in Table 1, LoKAN obtains the lowest NRMSE on all six noisy-derivative tasks and four of the six noisy-state tasks. On noisy-state Rössler, its error is 0.31975, compared with 0.43225 for the next-best model. It also achieves the lowest N-body error (0.03840). On Cascaded Tanks, LoKAN attains the lowest one-step (0.02444) and recursive (0.22209) errors, demonstrating accurate prediction on measured sequences as well as simulated dynamics. Figure 2 shows the early and late portions of the same recursive test trajectory for LoKAN and all five main baselines, providing a trajectory-level view of the full-rollout results in Table 1.

Table 1: Test NRMSE on noisy dynamical systems and measured-data benchmarks (lower is better). LV, VdP, and FHN abbreviate Lotka-Volterra, Van der Pol, and FitzHugh-Nagumo; Tanks denotes Cascaded Tanks. Direct and state refer to noisy derivatives and noisy states; 1-step and rollout refer to one-step and recursive prediction. Bold marks the best result per row.

Task	LoKAN	SR3	Spline GL	BP-KAN	HKAN	MatrixKAN
LV, direct	**5.972e-4**	8.589e-4	9.824e-4	0.01224	0.15214	0.01128
Duffing, direct	**9.981e-4**	1.353e-3	2.662e-3	5.530e-3	0.05578	0.01874
VdP, direct	**4.199e-4**	9.121e-4	9.439e-4	4.398e-3	0.24766	0.01159
Lorenz-63, direct	**8.628e-4**	1.177e-3	1.820e-3	0.02029	0.13425	0.03681
FHN, direct	**5.736e-4**	5.874e-4	1.817e-3	3.637e-3	0.38084	0.02519
Rössler, direct	**9.477e-4**	1.752e-3	1.799e-3	0.01809	0.13240	0.03821
LV, state	**0.02827**	0.03080	0.02973	0.03460	0.17520	0.03370
Duffing, state	0.01645	0.02111	0.01754	0.02440	0.07026	**0.01625**
VdP, state	**0.13179**	0.13583	0.13634	0.13489	0.33384	0.14030
Lorenz-63, state	0.74917	0.74920	0.74931	**0.74873**	0.75834	0.75793
FHN, state	**0.08335**	0.08607	0.08766	0.08552	0.57302	0.12626
Rössler, state	**0.31975**	0.43932	0.43921	0.43225	0.44310	0.45700
N-body	**0.03840**	0.05223	0.04352	0.04385	0.06707	0.04416
Silverbox, 1-step	0.03705	0.02073	0.04141	0.04110	0.13321	**0.01529**
Silverbox, rollout	0.08571	0.16899	0.11298	0.10506	0.59451	**0.06972**
Tanks, 1-step	**0.02444**	0.02715	0.02479	0.06296	0.05690	0.08330
Tanks, rollout	**0.22209**	0.48356	0.26894	0.43136	1.16768	0.84653

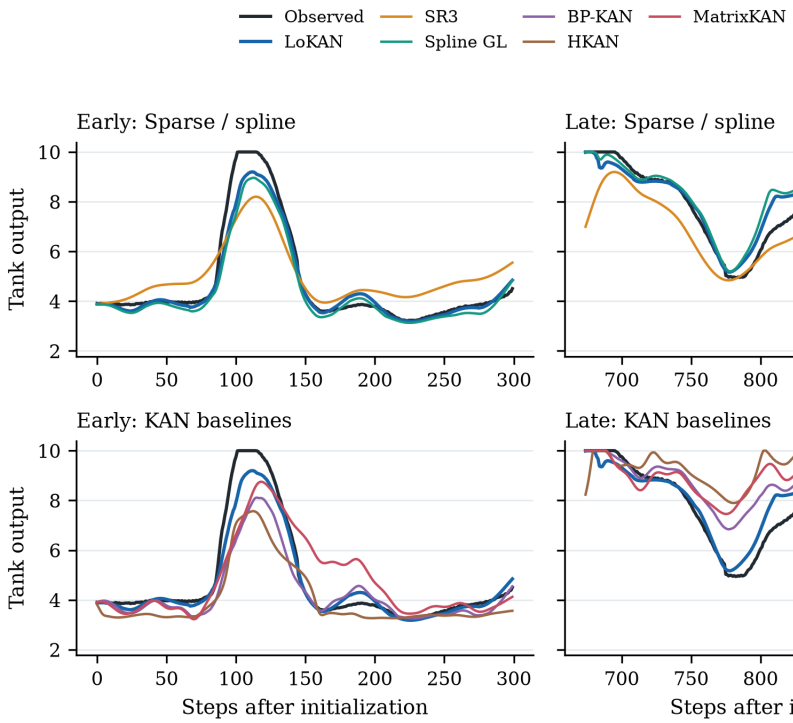


Figure 2: Recursive prediction on the Cascaded Tanks test sequence. The left and right columns show the first and last 300 predicted steps after initialization. The top row compares LoKAN with SR3 and Spline Group Lasso, while the bottom row compares it with BP-KAN, HKAN, and MatrixKAN. Each model recursively feeds its own predictions back during rollout.

Table 2: Test NRMSE with correlated, weak, and missing candidate terms. Bold indicates the lowest value in each row.

Task	LoKAN	SR3	Spline GL	BP-KAN	HKAN	MatrixKAN
Correlated decoy	**0.01892**	0.01986	0.01986	0.01996	0.12217	0.04130
Dense decoys	**0.01874**	0.02058	0.02058	0.02048	0.12050	0.04551
Weak true term	**0.02121**	0.02154	0.02160	0.02160	0.07206	0.02655
Missing $z_1 z_2$	**0.2232**	0.25697	0.2570	0.2557	0.2799	0.2411

Table 3: Joint-estimation ablation on representative tasks. Independent and Backfit denote independent-edge fitting and edge-wise backfitting. Direct and state refer to noisy derivatives and noisy states; Tanks denotes Cascaded Tanks. All entries are test NRMSE (lower is better).

Task	Independent	Backfit	LoKAN
Dense decoys	0.9254	0.02058	**0.01874**
Duffing, direct	2.06782	0.01619	**9.981e-4**
Duffing, state	2.08162	0.02525	**0.01645**
Tanks, rollout	2.1802	1.07798	**0.2221**

4.2 Equation Fitting with Correlated and Missing Candidates

All four tasks are generated from $y = z_1 - 0.8z_2 + 0.6z_1z_2 + \epsilon$, with 1,500 samples per task and noise standard deviation 0.03. The correlated-decoy task adds a candidate highly correlated with the true interaction. The dense-decoy task introduces additional correlated and irrelevant candidates. The weak-term task reduces the true interaction coefficient to 0.08. Finally, the missing-term task removes z_1z_2 from every model's input candidate set while leaving the target equation unchanged.

LoKAN obtains the lowest NRMSE in all four conditions (Table 2). Its errors on the correlated, dense-decoy, and weak-term tasks are 0.01892, 0.01874, and 0.02121. With the true interaction removed from the candidate set, LoKAN reaches 0.2232, compared with 0.2411-0.2799 for the other methods. Joint fitting thus remains effective as candidate correlation and representation difficulty increase.

4.3 Ablation of Joint Estimation

We compare LoKAN's joint estimation with one-pass independent-edge fitting and edge-wise backfitting while retaining the single-layer functional-edge representation. Table 3 selects representative candidate-interference, noisy-dynamics, and measured-system tasks. Cascaded Tanks is evaluated by recursive NRMSE; the other rows use direct test NRMSE.

LoKAN achieves the lowest NRMSE in all four selected ablations. On dense decoys, joint estimation reaches 0.01874, compared with 0.02058 for backfitting and 0.9254 for independent edges. On noisy Duffing derivatives and states, it reaches 9.981×10^{-4} and 0.01645, respectively. Its Cascaded Tanks rollout error is 0.22209, compared with 1.07798 for backfitting and 2.1802 for independent edges. These gains across candidate interference, dynamical systems, and measured sequences demonstrate the value of jointly estimating incoming functional edges.

5 Conclusion

This paper proposes LoKAN, a backpropagation-free framework for equation discovery that combines fixed spline edge functions, node-wise joint ridge regression, and contribution-based pruning. In our evaluation, LoKAN achieves the lowest test error on six noisy derivative tasks and four candidate-term equation-fitting tasks, while remaining competitive across a range of function-fitting and system-prediction tasks. Its expressive power depends on the design of candidate features: using only the original coordinates restricts the model to additive nonlinearities, whereas interaction terms must be explicitly included as candidate features. This framework provides a foundation for further research on backpropagation-free equation discovery methods that integrate function-on-edge representations, joint parameter estimation, and structural selection.

AI USE STATEMENT

We used GPT5.6sol as a tool to support writing and language polishing, as well as literature retrieval and discovery, including identifying potentially relevant related work. The authors are responsible for the accuracy and integrity of the paper, including all claims, citations, analyses, and conclusions.

REFERENCES

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73:243–272, 2008. doi: 10.1007/s10994-007-5040-8.

Gert-Jan Both, Subham Choudhury, Pierre Sens, and Remy Kusters. DeepMoD: Deep learning for model discovery in noisy data. *Journal of Computational Physics*, 428:109985, 2021. doi: 10.1016/j.jcp.2020.109985.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016a. doi: 10.1073/pnas.1517384113.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Sparse identification of nonlinear dynamics with control (sindyc), 2016b.

Andreas Buja, Trevor Hastie, and Robert Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, 17(2):453–510, 1989. doi: 10.1214/aos/1176347115.

Kathleen Champion, Peng Zheng, Aleksandr Y. Aravkin, Steven L. Brunton, and J. Nathan Kutz. A unified sparse optimization framework to learn parsimonious physics-informed models from data, 2020.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.2307/1271436.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. doi: 10.1016/0893-6080(89)90020-8.

Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1–3):489–501, 2006. doi: 10.1016/j.neucom.2005.12.126.

Kadierdan Kaheman, J. Nathan Kutz, and Steven L. Brunton. SINDy-PI: A robust algorithm for parallel implicit sparse identification of nonlinear dynamics. *Proceedings of the Royal Society A*, 476(2242):20200279, 2020. doi: 10.1098/rspa.2020.0279.

Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993. doi: 10.1016/S0893-6080(05)80131-5.

Ziming Liu, Max Tegmark, Pingchuan Ma, Wojciech Matusik, and Yixuan Wang. Kolmogorov-arnold networks meet science. *Phys. Rev. X*, 15:041051, Dec 2025a. doi: 10.1103/4t7t-v19l. URL https://link.aps.org/doi/10.1103/4t7t-v19l.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Hou, and Max Tegmark. Kan: Kolmogorov–arnold networks. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 70367–70413, 2025b.

Zichao Long, Yiping Lu, and Bin Dong. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 399:108925, 2019. doi: 10.1016/j.jcp.2019.108925.

Daniel A. Messenger and David M. Bortz. Weak SINDy for partial differential equations. *Journal of Computational Physics*, 443:110525, 2021a. doi: 10.1016/j.jcp.2021.110525.

Daniel A. Messenger and David M. Bortz. Weak SINDy: Galerkin-based data-driven model selection. *Multiscale Modeling & Simulation*, 19(3):1474–1497, 2021b. doi: 10.1137/20M1343166.

Shirin Panahi, Mohammadamin Moradi, Erik M. Bollt, and Ying-Cheng Lai. Data-driven model discovery with kolmogorov-arnold networks. *Phys. Rev. Res.*, 7:023037, Apr 2025. doi: 10.1103/PhysRevResearch.7.023037. URL https://link.aps.org/doi/10.1103/PhysRevResearch.7.023037.

Maziar Raissi, Paris Perdikaris, and George E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

Pradeep Ravikumar, John Lafferty, Han Liu, and Larry Wasserman. Sparse additive models. *Journal of the Royal Statistical Society: Series B*, 71(5):1009–1030, 2009. doi: 10.1111/j.1467-9868.2009.00718.x.

Eric Reinhardt, Dinesh Ramakrishnan, and Sergei Gleyzer. SineKAN: Kolmogorov-arnold networks using sinusoidal activation functions. *Frontiers in Artificial Intelligence*, 7:1462952, 2025. doi: 10.3389/frai.2024.1462952.

Samuel H. Rudy, Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017. doi: 10.48550/arXiv.1609.06401.

Subham S. Sahoo, Christoph H. Lampert, and Georg Martius. Learning equations for extrapolation and control. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4442–4450, 2018.

Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, Rafal Bogacz, and Zhenghua Xu. Reverse differentiation via predictive coding, 2023. URL https://arxiv.org/abs/2103.04689.

Abraham Savitzky and Marcel J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964. doi: 10.1021/ac60214a047.

Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11:24, 2017. doi: 10.3389/fncom.2017.00024.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009. doi: 10.1126/science.1165893.

Maarten Schoukens, Per Mattsson, Torbjorn Wigren, and Jean-Philippe Noël. Cascaded tanks benchmark combining soft and hard nonlinearities, 2016. URL https://www.nonlinearbenchmark.org/benchmarks/cascaded-tanks.

Yuhang Song, Thomas Lukasiewicz, Zhenghua Xu, and Rafal Bogacz. Can the brain do backpropagation? exact implementation of backpropagation in predictive coding networks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22566–22579, 2020.

Charles J. Stone. Additive regression and other nonparametric models. *The Annals of Statistics*, 13 (2):689–705, 1985. doi: 10.1214/aos/1176349548.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996. doi: 10.1111/j.2517-6161.1996.tb02080.x.

Silviu-Marian Udrescu and Max Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020. doi: 10.1126/sciadv.aay2631.

James C. R. Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural Computation*, 29(5):1229–1262, 2017. doi: 10.1162/NECO_a_00949.

Torbjörn Wigren and Johan Schoukens. Three free data sets for development and benchmarking in nonlinear system identification. In *2013 European Control Conference*, pp. 2933–2938, 2013. doi: 10.23919/ECC.2013.6669201.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B*, 68(1):49–67, 2006. doi: 10.1111/j.1467-9868.2005.00532.x.

Sheng Zhang and Guang Lin. Robust data-driven discovery of governing physical laws with error bars. *Proceedings of the Royal Society A*, 474(2217):20180305, 2018. doi: 10.1098/rspa.2018.0305.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67(2):301–320, 2005. doi: 10.1111/j.1467-9868.2005.00503.x.

A METHOD DETAILS AND PROOFS

The results below concern single-layer LoKAN with a fixed candidate map and fixed spline bases. Statements about exact coefficient solves additionally hold the output-node support and ridge scale fixed.

A.1 BASIS CONSTRUCTION, TRAINING PROCEDURE, AND SELECTION

Let S_i denote the active edges of output i and K the knot count. For $K > 2$, use knot positions $\kappa_k = -3 + 6(k-1)/(K-1)$ and spacing $h = 6/(K-1)$, and define

$$B_{K,k}(u) = \max\left\{1 - \frac{|u - \kappa_k|}{h}, 0\right\}, \qquad k = 1, \ldots, K. \tag{12}$$

The anchored basis is $a_K(z) = B_K(\mathrm{clip}(z, -3, 3)) - B_K(0)$. For $K = 2$, use $a_2(z) = (-z/2, z/2)^\top$, which extends linearly over the real line. Both constructions satisfy $a_K(0) = 0$. Although the two columns of the $K = 2$ basis are linearly dependent, a positive ridge scale makes the fixed-support augmented system positive definite.

Algorithm A.1 (node-local LoKAN training). Inputs are fitting data, candidate map θ, knot count K, base ridge parameter $\lambda > 0$, contribution threshold τ, and pruning budget R.

1. Compute fitting-set means and scales for all θ_j and y_i, and cache the anchored basis blocks A_j.
2. For each output i, initialize $S_i^{(0)} = \{1, \ldots, p\}$. At round ℓ, set $q_i^{(\ell)} = 1 + K|S_i^{(\ell)}|$ and
$$\rho_i^{(\ell)} = \lambda \left[1 + \frac{\mathrm{tr}(\bar{A}_i^{(\ell)\top} \bar{A}_i^{(\ell)})}{q_i^{(\ell)}}\right].$$
Solve $\left(\bar{A}_i^{(\ell)\top} \bar{A}_i^{(\ell)} + \rho_i^{(\ell)} D_i^{(\ell)}\right) w_i^{(\ell)} = \bar{A}_i^{(\ell)\top} t_i$.
3. Evaluate the contribution score in Equation equation 10 for every $j \in S_i^{(\ell)}$. Retain edges whose scores are at least τ; if none qualifies, retain the highest-scoring edge. Call the result $S_i^{(\ell+1)}$.
4. Stop if $S_i^{(\ell+1)} = S_i^{(\ell)}$ or R scoring rounds have been completed; otherwise return to Step 2. Removed edges do not re-enter later supports.
5. Hold the final S_i fixed, recompute its ridge scale, and solve its joint system once more. Save the support, coefficients, and fitting-set standardization statistics.

Validation data select K, λ, τ, and other task configurations; test data are evaluated only after selection is frozen. Candidate-library definitions, validation splits, and search grids for each task belong in the experimental setup. The experimental pruning budget is $R = 4$, and the final fixed-support solve is performed even when pruning stabilizes early.

A.2 WHY JOINT ESTIMATION IS NEEDED AND PROOF OF THE GRAM-COMPONENT RESULT

First consider two perfectly correlated edges. After removing the intercept, suppose each edge has one effective, identical, centered design column $u \neq 0$, with $\mathbf{1}^\top u = 0$. Let $g = u^\top u > 0$, $h = u^\top M t$, and $M = I - \mathbf{1}\mathbf{1}^\top/N$. For a common fixed ridge scale $\rho > 0$, the joint system is

$$\begin{bmatrix} g + \rho & g \\ g & g + \rho \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} h \\ h \end{bmatrix}, \qquad c_1 = c_2 = \frac{h}{2g + \rho}. \tag{13}$$

Fitting the two edges separately gives $h/(g + \rho)$ for each. When $h \neq 0$, these estimates differ from the joint solution and duplicate the same signal across the two edges. Parameter separation across KAN edges therefore does not imply separability of their estimation at a shared target node.

We now prove Proposition 1. Fix an output support S and ridge scale $\rho > 0$, and write $A = [A_j]_{j\in S}$, $G = A^\top MA$, and $H = G + \rho I$. Since $G \succeq 0$, H is positive definite. For a partition $\mathcal{P} = \{P_1, \ldots, P_L\}$ of the active edges, separate solves use

$$\widetilde{H} = \mathrm{blkdiag}_{P\in\mathcal{P}}(G_{PP} + \rho I_P). \tag{14}$$

For an arbitrary right-hand side $r \in \mathbb{R}^{K|S|}$, the joint and partitioned solutions are $H^{-1}r$ and $\widetilde{H}^{-1}r$. They agree for every r if and only if $H^{-1} = \widetilde{H}^{-1}$. Both matrices are invertible, so this is equivalent to $H = \widetilde{H}$, or

$$G_{jk} = 0 \quad \text{whenever edges } j \text{ and } k \text{ belong to different partition blocks.} \tag{15}$$

In the graph with edges j–k exactly when $G_{jk} \neq 0$, condition equation 15 requires the endpoints of every graph edge to remain in the same partition block. Every exact partition block must therefore be a union of complete connected components. The components form the unique finest exact partition. Splitting a component leaves a nonzero cross-partition Gram block, so $H \neq \widetilde{H}$ and at least one right-hand side yields different solutions. Edge-wise estimation is the singleton-block case: it is exact for every right-hand side if and only if every cross-edge Gram block vanishes. This statement concerns *all algebraic right-hand sides*; a particular target can still produce the same solution under a nonexact partition by accidental cancellation.

A.3 Conversion from two-knot edges to equation coefficients

When $K = 2$, $a_2(z) = (-z/2, z/2)^\top$ gives

$$\phi_{ij}(z) = \frac{c_{ij,2} - c_{ij,1}}{2}\, z = \gamma_{ij} z. \tag{16}$$

Substitute the standardization in Equation equation 1 into Equation equation 3, then transform the target back to its original units:

$$\begin{aligned} \widehat{y}_i(x) &= \mu_i^y + \sigma_i^y b_i + \sum_{j\in S_i} \sigma_i^y \gamma_{ij} \frac{\theta_j(x) - \mu_j^\theta}{\sigma_j^\theta} \\ &= \beta_{i0} + \sum_{j\in S_i} \beta_{ij}\theta_j(x). \end{aligned} \tag{17}$$

Thus $\beta_{ij} = (\sigma_i^y/\sigma_j^\theta)\gamma_{ij}$ and $\beta_{i0} = \mu_i^y + \sigma_i^y b_i - \sum_{j\in S_i} \beta_{ij}\mu_j^\theta$, as stated in Equation equation 11. A constant candidate column becomes identically zero after standardization, and its constant contribution is represented by the node intercept. Equation equation 17 supports term-level coefficient reporting when the candidate features correspond to the intended equation terms and the edges are linear. The K coefficients of a multi-knot edge instead parameterize a univariate function, not a single equation-term coefficient.

A.4 Equivalence of sufficient statistics across data shards

Fix the candidate representation and support for output i, and partition its training rows into L disjoint shards. Let $\bar{A}_i^{(r)}$ and $t_i^{(r)}$ denote the augmented design and standardized target on shard r. Define

$$Q_r = \bar{A}_i^{(r)\top} \bar{A}_i^{(r)}, \qquad h_r = \bar{A}_i^{(r)\top} t_i^{(r)}. \tag{18}$$

Row concatenation gives $Q = \sum_{r=1}^L Q_r = \bar{A}_i^\top \bar{A}_i$ and $h = \sum_{r=1}^L h_r = \bar{A}_i^\top t_i$. With $q_i = 1 + K|S_i|$, compute the ridge scale after aggregation as $\rho_i = \lambda[1 + \mathrm{tr}(Q)/q_i]$. The resulting system

$$(Q + \rho_i D_i) w_i = h \tag{19}$$

is identical to the centralized Equation equation 6. All shards must use the same fitting-set standardization, candidate map, basis, and support. Independently standardizing shards or computing separate ridge scales changes the system. The centered Gram matrix can likewise be recovered from aggregated uncentered statistics: $A^\top MA = A^\top A - (A^\top \mathbf{1})(\mathbf{1}^\top A)/N$.

Table 4: Test NRMSE on nonlinear function-fitting tasks. Bold indicates the lowest value in each row.

Task	LoKAN	SR3	Spline GL	BP-KAN	HKAN	MatrixKAN
Pendulum	6.077e-3	0.02891	6.319e-3	6.363e-3	0.02701	**3.619e-3**
Rational	**0.01088**	0.08514	0.01095	0.01096	0.05544	0.01709
Saturation	**0.01601**	0.19795	0.01694	0.01694	0.13587	0.06814
Multifrequency	**0.01173**	0.64317	0.15284	0.15320	0.41658	0.15663
Raw interaction	0.99781	**0.06830**	1.00035	1.00211	0.91585	1.00138
Lifted interaction	**3.905e-3**	0.02929	3.955e-3	6.901e-3	0.12794	6.613e-3

A.5 Contribution scoring and support interpretation

The numerator of Equation equation 10 is the root-mean-square amplitude of a fitted edge over the training rows. If the same edge function is reparameterized in another basis while its samplewise contributions v_{rij} remain unchanged, its score remains unchanged; $\|c_{ij}\|_2$ generally does not have this property. Division by target standard deviation expresses the threshold relative to target variation.

Contribution magnitude alone cannot resolve highly correlated candidate terms. The identical columns in Appendix A.2 have identical fitted contributions, so their individual attribution cannot be determined from the observed target alone. Ridge regularization ensures a unique coefficient vector for a fixed support, while contribution pruning gives a computable structure-selection rule. Neither guarantees recovery of the true support for an arbitrary candidate library. Support F1 or exact-support rates should therefore be reported only where both candidate terms and ground-truth support have explicit definitions.

A.6 Expressivity boundary of raw-coordinate additive edges

If the candidates are only raw coordinates, a two-input single-layer model has the form $g(x_1, x_2) = b + g_1(x_1) + g_2(x_2)$. Every such function obeys the rectangular identity

$$g(1,1) + g(0,0) - g(1,0) - g(0,1) = 0. \tag{20}$$

For $f(x_1, x_2) = x_1 x_2$, the same expression equals one. If $\|f - g\|_\infty < 1/4$, the absolute sum of the four pointwise errors would be strictly less than one, contradicting this difference. Hence the uniform approximation error of any raw-coordinate additive model for $x_1 x_2$ on $[0,1]^2$ is at least $1/4$, regardless of knot count. Multivariate interactions therefore require corresponding combination features in the candidate representation or a mechanism beyond this single-layer additive structure.

B Supplementary Experiments

B.1 Nonlinear Function Fitting and Representation Limits

For each task, we sample 9,000 points from $(a, b) \in [-3, 3]^2$. The targets are a pendulum mapping $[b, -\sin(a) - 0.1b]$, a rational/hyperbolic-tangent mapping $[-a/(1+a^2) + 0.5b, -0.5a - \tanh(b)]$, a saturation mapping $\mathrm{clip}(1.7a, -1, 1) + 0.25b$, a multifrequency mapping $\sin(2a) + 0.2\sin(9a) + 0.3b$, and an interaction mapping $ab + 0.5\sin(a + b)$. For the interaction task, we compare the raw inputs (a, b) with lifted inputs augmented by ab and $a + b$.

LoKAN has the lowest NRMSE on four of the six function-fitting tasks (Table 4). Its multifrequency error is 0.01173, compared with 0.15284–0.64317 for the other methods. Adding interaction coordinates reduces LoKAN's error from 0.99781 to 0.003905, showing how input lifting expands the functions captured by a shallow functional-edge model.

B.2 Computational Efficiency and Gram Storage

We report a separate efficiency evaluation of the fixed-library LoKAN instantiation. Model fitting and support selection were timed on one NVIDIA RTX 4050 Laptop GPU with PyTorch 2.7.0 and

Table 5: Fixed-library recovery with 5% training-derivative noise. SR3, Spline GL, and STLSQ denote PySINDy-SR3, spline group lasso, and SINDy-STLSQ; BP-KAN and STLSQ use tuned configurations. Time is the median per fit, including support selection.

Metric	LoKAN	SR3	Spline GL	BP-KAN	STLSQ
NRMSE ↓	0.0505	0.0634	0.0555	0.0531	0.0501
Support F1 ↑	0.991	0.967	0.708	0.830	0.994
Exact support ↑	0.958	0.875	0.175	0.525	0.975
Coef. error ↓	0.0061	0.0306	0.1615	0.0408	0.0019
Time (ms) ↓	9.35	2.46	281	2223	1.13

Table 6: Exact Gram-component scaling. Relative error is measured against the full-node coefficient solution; speedup compares solve times. Gram reduction is the ratio of full-matrix to component-matrix element counts.

Edges	Coefficients	Relative error ↓	Solve speedup ↑	Gram reduction ↑
16	64	9.1e-16	1.64×	4×
32	128	1.8e-15	2.29×	8×
64	256	2.2e-15	2.74×	16×
128	512	2.1e-15	9.09×	32×
256	1,024	2.3e-15	33.75×	64×

CUDA 12.8; data generation and hyperparameter search were excluded. This protocol is distinct from the A–D accuracy benchmarks in the main text. In the 5% training-derivative-noise setting, LoKAN fits in 9.35 ms, compared with 2,223 ms for the forward-matched BP-KAN and 281 ms for spline group lasso (Table 5).

We also test exact Gram-component decomposition separately from end-to-end training. Each component contains four edges with four knots per edge; within-component correlation is 0.8 and cross-component Gram blocks vanish. The component solves reproduce the full-node solution to numerical precision. At 1,024 coefficients, they reduce solve time by $33.75\times$ and Gram-matrix storage by $64\times$ (Table 6). The storage ratio counts matrix elements, rather than peak process or GPU memory.