RULEKAN: ADDRESSING THE SYMBOLIC EXPRESSIVITY GAP IN KANS FOR SYMBOLIC REGRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Symbolic regression tries to turn data into analytic equations that carry useful meaning. Kolmogorov–Arnold Networks (KANs) are attractive for this purpose because their learned one-dimensional edge functions can be replaced by analytic formulas. The same flexibility creates a problem: each KAN edge can absorb structure that a scientist may want the equation to expose. For example, a single learned curve can approximate $e^x \sin x$, and a two-branch fuzzy-rule can be reproduced without displaying its gate and branches. We study which symbolic capacities are needed to recover those structures and more, introducing two related method families: *RuleKAN* trains a KAN to propose interacting variables and then searches explicit sums of products of analytic factors; *RuleSISP* searches the same language without the learned-support restriction. Bounded extensions of these methods add whole-expression powers, ratios, and one additional composition level.

Across 30 analytic tasks spanning products, fuzzy rules, powers and ratios, nested functions, canonical KAN targets, and physics equations, the RuleKAN family performs best overall (NRMSE 1.07×10^{-3}; mean rank 4.97) against baseline methods like PySR, SR-KAN, Operon, etc. It has significantly lower predictive error than Operon, SR-KAN, and the other six KAN-based baselines (FDR-adjusted $q < 0.001$), and wins on about 66% of tasks against PySR. Notably, on fuzzy rules, RuleSISP recovers the generating structure significantly better than PySR despite its competitive predictive error (mean expanded-rule F1 0.446 vs. 0.000; $q = 0.0084$). The results show that symbolic recovery is not one search problem, separating distinct bottlenecks in factorization, interaction coverage, whole-expression operations, symbolic depth, and recovery of the intended structure.

1 INTRODUCTION

Symbolic regression is the process of extracting from data an analytic equation that describes the relationships among its variables (Schmidt & Lipson, 2009; Cava et al., 2021; Cranmer, 2023). Unlike ordinary numerical regression, its output is intended to be read as mathematics: factors, interactions, and transformations are part of what must be recovered. A good numerical predictor may reproduce the same input–output behaviour without making this structure explicit.

This distinction matters because different symbolic expressions can approximate the same function while exposing very different structures. For example, $e^x \sin x$ may be represented by a single univariate function $h(x)$ or by a series expansion, neither of which reveals its factorization into an exponential and a sine. Symbolic recovery therefore requires not only an accurate approximation, but also a representation that exposes the structure of interest.

Kolmogorov–Arnold Networks (KANs) are attractive for this task because they decompose a fitted numerical model into one-dimensional functions that can be inspected and converted individually into analytic expressions (Liu et al., 2025b). This provides an intermediate representation between numerical fitting and symbolic recovery: rather than searching for an entire equation directly from the data, one can first use the KAN to identify simpler functional components and then attempt to express them symbolically. Specifically, a KAN processes data through a sequence of stages made up of numerical variables. Each variable is connected to variables in the next stage, thus forming a network (of functions). And each *connection*, or *edge*, of such network applies a learned univariate function

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.

to the value it receives. After training, these learned edge functions are independently replaced by analytic expressions such as sine, exponential, or polynomial functions.

As shown in Figure 1, this edge-wise extraction can again miss symbolic structure when a learned edge function cannot be represented by a single analytic function from the symbolic vocabulary. For example, an edge may learn $h(x) \approx e^x \sin x$, a product of two functions, e^x and $\sin x$. This can occur when edge functions are implemented using universal function approximators, as in KANs. Interactions such as xy pose a different challenge at the edge level: no single univariate edge acting on one scalar input represents xy directly. Compact analytic functions may also appear through truncated series rather than as recognizable primitives. These cases require symbolic search beyond independent edge replacement.

The issue is particularly important for fuzzy if–then–else rules. A two-branch rule can be written as $(1-g)u + gv$, where u and v are branch responses and $g \in [0, 1]$ is a gate. When $g = 0$, the output follows u; when $g = 1$, it follows v. Recovering such a rule therefore requires more than finding a numerically accurate expression: the symbolic representation must expose the gate, the two branches, and their particular multiplicative structure.

These observations motivate our two proposed methods, **RuleKAN** and **RuleSISP**, which extend symbolic extraction beyond the individual KAN edge (as shown in Figure 2), with fuzzy if–then–else rules as a motivating case. Rather than replacing each learned edge independently with a single analytic function, both methods search over sums of products of analytic factors, using trainable mixtures of candidate functions followed by discrete selection and refitting. RuleKAN first uses a KAN-derived numerical model to learn candidate variable interactions that constrain this search, whereas RuleSISP (Structure-Independent Symbolic Pursuit) enumerates variable multisets directly, at greater combinatorial cost. For structures beyond this sum–product language, we further introduce bounded extensions for powers, reciprocals, and analytic composition.

To assess these capacities, we compare against 14 external baselines spanning KAN-based extraction and general symbolic regression, including Symbolic-KAN, PySR, Operon, RILS-ROLS, ParFam, EQL, and SINDy; Section 5 gives the full set. We evaluate them on 30 analytic tasks covering products, fuzzy rules, powers, ratios, nested expressions, and canonical KAN targets.

This evaluation is organized around answering four research questions. **(RQ1)** Does a sum–product symbolic representation improve recovery when multiple analytic factors or fuzzy-rule components must be exposed explicitly? **(RQ2)** What is gained and lost by restricting symbolic search to variable interactions identified by a numerical KAN? **(RQ3)** When are whole-expression operators or additional composition depth needed beyond flat sums of products? **(RQ4)** When the generating structure is known, how closely does predictive accuracy reflect its symbolic recovery?

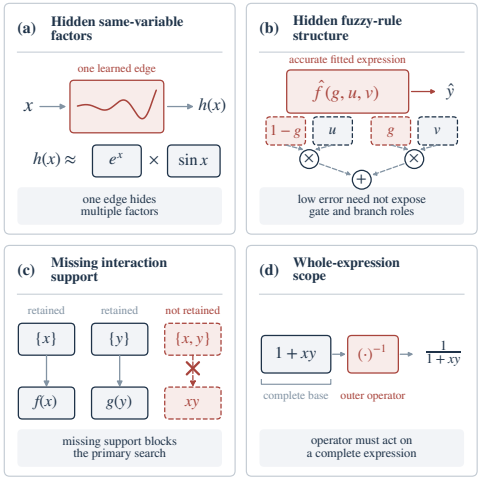

Figure 1: Per-edge symbolic extraction can hide factors, interactions, and conditions.

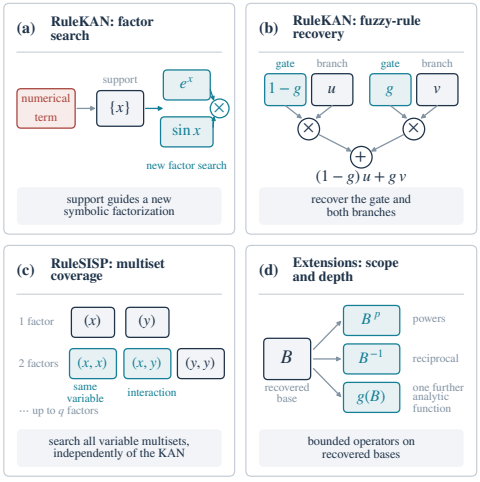

Figure 2: RuleKAN conditions structured search on learned interactions; RuleSISP does not.

2 RULEKAN AND RULESISP

Preliminaries: KAN pruning and symbolic extraction, GSR, and GMP. A Kolmogorov–Arnold Network (KAN) is a layered numerical model in which each directed connection, or *edge*, carries a trainable univariate function. If $z^{(\ell)}$ is the input to layer ℓ, an output coordinate has the form $z_i^{(\ell+1)} = b_i^{(\ell)} + \sum_j \phi_{ij}^{(\ell)}(z_j^{(\ell)})$, so one layer adds edge-function outputs and interactions can arise after layers are composed (Liu et al., 2025b). In the standard KAN interpretability workflow, sparsity regularization during training and subsequent pruning simplify the network, after which surviving learned edge functions $\phi_{ij}^{(\ell)}$ are replaced by simple analytic functions such as $\sin$, $\exp$, or x^2 (Liu et al., 2025b). We call the latter step *symbolic extraction*.

GSR and GMP change how those analytic replacements are selected (Sovrano et al., 2026). GSR tests discrete edge replacements greedily using the error of the complete network rather than the fit to one edge in isolation. GMP instead optimizes a differentiable mixture over candidate analytic functions and then selects a discrete function. Both improve the context in which an edge is symbolized, but the symbolic decision is still made at the level of an individual edge. This distinction motivates RuleKAN: the numerical model is used to determine which inputs interact, while the factorization of those inputs is decided separately in an explicit symbolic language.

Algorithms Overview. Let $x = (x_1, \ldots, x_d)$ denote the inputs, q the maximum number of symbolic factors in one product term, W the number of candidate numerical terms, and K the number of primitives in the analytic vocabulary. RuleKAN separates three stages.

Stage 1 trains and prunes a KAN-derived numerical interaction model to identify which inputs occur together. *Stage 2* resolves those groups into sums of products of analytic functions; fuzzy rules are a special case. *Stage 3* optionally adds integer powers, ratios, or one analytic composition around complete Stage 2 bases. RuleSISP uses the same Stage 2 and Stage 3 languages but enumerates variable groups independently of Stage 1.

A running example illustrates the separation. Suppose the target contains $e^{-0.7x}\sin(2.4x)$. Stage 1 only needs to establish that x is relevant to the term. A flexible numerical edge can represent the entire product as one univariate function, so its internal shape does not determine the symbolic factorization. Stage 2 therefore searches explicit analytic factors and may recover the exponential and sine as two separate factors of the same input.

Stage 1: which variables interact. Stage 1 adapts the pruning step to the structure needed by RuleKAN. Standard KAN pruning simplifies individual network nodes and edges before per-edge symbolic extraction (Liu et al., 2025b); RuleKAN instead prunes explicit candidate numerical terms and factor slots so the surviving terms identify multivariate variable supports. To make those interactions explicit, the numerical KAN is written as a sum of product terms. After the trainable choices are discretized, the numerical interaction model is

$$f_{\mathrm{num}}(x) = b + \sum_{r=1}^{W} \rho_r a_r \prod_{s=1}^{q} \left[1 - m_{rs} + m_{rs} h_{rs}(x_{j_{rs}})\right]. \tag{1}$$

Here h_{rs} is a univariate spline or radial-basis approximator applied to $x_{j_{rs}}$, and a_r is the term amplitude. During relaxed fitting, each factor slot softly weights the d input-specific edge outputs; discretization selects j_{rs}. The binary variables ρ_r and m_{rs} are RuleKAN selection gates, not components of a standard KAN: ρ_r retains or removes a complete term and m_{rs} retains or removes one factor slot, with $m_{rs} = 0$ replacing that factor by one. During training these choices are relaxed and sparse Hard-Concrete gates encourage compact terms (Louizos et al., 2018). Stage 1 then makes the sparsity discrete by iterative pruning: gates rounded off are removed, the surviving numerical model is refit, and the prune-refit step is repeated. Pruning acts on numerical terms and factor slots rather than on symbolic factors. Equation (1) is therefore a numerical interaction model, not the final symbolic expression.

For each retained numerical term, Stage 1 records only its *support*, the set of distinct input indices used by that term, $S_r = \{j_{rs} : m_{rs} = 1\}$. The distinct retained supports form the primary RuleKAN support bank $\mathcal{S}$. A support disappears from this bank only when no retained numerical term realizes that set of variables. The learned h_{rs} are not passed to Stage 2 as symbolic factors. In the running example, a numerical edge may learn $h(x) \approx e^{-0.7x}\sin(2.4x)$, but Stage 1 passes only the support $\{x\}$. This prevents the arbitrary numerical decomposition from fixing how many analytic factors of x must appear in the equation. The split is needed because h_{rs} is a universal function approximator:

multiple edge functions of the same input can be absorbed into another function, so extra numerical edges do not identify extra symbolic factors unless the edge family is restricted, as in GMP.

Stage 2: disambiguating the factor structure. Stage 2 rebuilds each term found by Stage 1 as a product of analytic functions drawn from a deliberately small *vocabulary* $\mathcal{V} = \{\psi_1, \ldots, \psi_K\}$ of *primitives*: identity, square, reciprocal, inverse square, square root, $\log$, $\exp$, $\sin$, $\cos$, and $\tanh$ ($K = 10$; Appendix A). Since $\mathcal{V}$ contains $\exp$ and $\sin$ but not $e^{-0.7x}$ or $\sin(2.4x)$, each factor applies its primitive to $\beta x + \gamma$ with a learned scale β and shift γ: $e^{-0.7x}$ is exp with $\beta = -0.7$, $\gamma = 0$. The expression Stage 2 searches for is

$$f_{\mathrm{SP}}(x) = b + \textstyle\sum_{r=1}^{R} a_r \prod_{s=1}^{m_r} \psi_{k_{rs}}(\beta_{rs} x_{j_{rs}} + \gamma_{rs}), \qquad m_r \le q. \tag{2}$$

As in Eq. (1), term r has amplitude a_r and factor s acts on input variable $x_{j_{rs}}$; now, however, factor s is a primitive with its own β_{rs}, γ_{rs}, the same variable may appear in several factors of one term, and there are no switches ρ_r, m_{rs}: the numbers of terms R and factors m_r are decided by symbolic search, not by Stage 1 pruning. Because a primitive has a fixed shape, a product can no longer be absorbed into one factor unless the product reduces to another member of the vocabulary. Thus, the search has two levels: the *structure* of a term, that is, which input variables it multiplies and how many times each (level 1), and then the *primitive and constants* of each factor (level 2).

A structure is a *variable multiset* z, the input variables of the term listed with repetition. For example, $z = (1, 1, 3)$ denotes a product with two analytic factors of x_1 and one of x_3. Its support $\mathrm{supp}(z)$ is the set of distinct indices appearing in the list, so $\mathrm{supp}(1, 1, 3) = \{1, 3\}$. Given a Stage 1 support such as $\{1, 3\}$, Stage 2 enumerates every admissible variable multiset with that support up to order q, such as $(1, 3)$, $(1, 1, 3)$, and $(1, 3, 3)$ when $q = 3$. Several additive terms may use the same variable multiset. In other words, RuleKAN generates candidate structures from the Stage 1 supports in two steps. We call the first step *repeated-variable expansion*: for a fixed support, Stage 2 tries every allowed repetition pattern up to order q. For example, support $\{x, y\}$ with $q = 3$ gives (x, y), (x, x, y), and (x, y, y). This lets the symbolic model recover several factors of the same input even if Stage 1 represented them with one numerical edge. The second step is *additive-term expansion*: the same variable multiset may appear in several additive terms of Eq. (2), subject to the bound $R_{\max}$ specified in Appendix D. For example, $[e^{-0.7x} \sin(2.4x) + \cos x]\, y$ is one numerical term but requires two symbolic terms with support $\{x, y\}$. Likewise, if a numerical model approximates $\sin x$ by $x - x^3/3! + x^5/5!$, Stage 1 still passes only support $\{x\}$; Stage 2 can then select $\sin$ directly or use several polynomial terms when their orders are within q. The primary candidate structures of RuleKAN are thus $\mathcal{G}_R = \bigcup_u \{z : \mathrm{supp}(z) = S_u,\ |z| \le q\}$, where $S_1, \ldots, S_U$ ($U \le W$) are the distinct supports found by Stage 1.

The second step, choosing the primitive of each factor, is a discrete decision, handled like the switches ρ_r, m_{rs} of Stage 1: relax, train, round. The *relaxed* phase replaces each factor by a weighted mixture $\sum_k w_k \psi_k(\beta x + \gamma)$ of all K primitives and trains the weights w_k jointly with β, γ and the amplitudes, as in GMP, applied here to factors of a product rather than to KAN edges. The *discrete* phase keeps the L best primitives per factor and refits every combination of them with one primitive per factor using a matching pursuit strategy, just like in GSR (Sovrano et al., 2026).

Fuzzy rules as Stage 2 factorizations. A two-branch fuzzy rule with gate $g \in [0, 1]$ and branch responses u and v can be written $(1 - g)u + gv$. This is already a sum of products in Eq. (2) (Takagi & Sugeno, 1985). RuleKAN can therefore search the two branch terms jointly and tie the complementary affine gate parameters: if $g(x_j) = \beta x_j + \gamma$, then $1 - g(x_j) = -\beta x_j + (1 - \gamma)$. The explicit gate and branch roles are retained even though distributivity also gives the equivalent form $u - gu + gv$. Finite nested fuzzy trees likewise expand into sums of products of path memberships and leaf responses; Appendix B gives the expansion.

RuleKAN vs. RuleSISP. RuleKAN and RuleSISP use Eq. (2), the same analytic vocabulary, and the same factor-fitting procedure. They differ in how candidate variable multisets are constructed. RuleKAN's primary search uses the pruned Stage 1 support bank $\mathcal{S} = \{S_1, \ldots, S_U\}$ and admits $\mathcal{G}_R = \bigcup_{u=1}^{U} \{z : \mathrm{supp}(z) = S_u,\ 1 \le |z| \le q\}$. Repeated factors and additive terms are first considered for these supports. Specialized validation-gated rescue may then derive nonempty support subsets or gate-aware unions from Stage 1 evidence, so the effective bank can contain supports absent from $\mathcal{S}$ without enumerating arbitrary supports (Appendix F). RuleSISP instead considers every variable multiset through order q, $\mathcal{G}_S = \{z : 1 \le |z| \le q\}$.

Stage 3: operators on complete expressions. Stages 1 and 2 recover an explicit symbolic model by identifying the relevant variable structure and resolving it into analytic factors and product terms. This already captures multivariate interactions such as xy, $e^x \sin y$, and sums of such terms. Some targets, however, require an additional operation applied to an expression that Stage 2 has already constructed. For example, $(1 + xy)^{-1}$ applies a reciprocal to the complete base $1 + xy$, while $\sin(1 + xy)$ applies an analytic function to that same base. More generally, a numerical KAN may encode a nested mapping $h(x) \approx f(g(x))$, where Stage 2 can recover the inner expression $g(x)$ but not necessarily the outer transformation f. Stage 3 handles these cases by extending the recovered Stage 2 expressions with integer powers, including negative powers for reciprocals and ratios, and with analytic composition.

Stage 3 has two extensions: *PowerRuleKAN* for integer powers, including reciprocals through negative powers, and the **-Comp* variants for one analytic composition level.

The first extension, PowerRuleKAN, searches powers of Stage 2-language bases B_{rt} assembled in Stage 2: $f_{\mathrm{PSP}}(x) = b + \sum_{r=1}^{R_o} a_r \prod_{t=1}^{b_r} B_{rt}(x)^{p_{rt}}, p_{rt} \in D \subset \mathbb{Z} \setminus \{0\}$, where R_o is the number of outer terms, b_r is the number of Stage 2 bases multiplied in outer term r, and D is a finite set of nonzero integer powers. Negative powers introduce reciprocals and ratios, such as $(1 + xy)^{-1}$. Positive integer powers do not enlarge the unbounded finite sum-product closure because $(b+\sum_r t_r)^p$ has a finite product-term expansion. Under fixed factor-order and term-count bounds, however, an explicit power can enlarge the exactly representable bounded family and can make the expression much shorter. Appendix B–C gives the proof and term counts.

The second extension, used by RuleKAN-Comp and RuleSISP-Comp, searches one analytic composition around selected bounded Stage 2 base topologies: a unary factor, a two-variable product, or a sum of two unary factors. It considers expressions of the form $C(x) = a\, g(\beta B(x) + \gamma) + d$, where $B(x)$ has one of these topologies and $g \in \mathcal{V}$ is an analytic primitive. This allows the symbolic model to recover nested structure such as $\sin(1 + xy)$ and $\exp(xy)$ without unrestricted recursive tree search. PowerRuleKAN-Comp combines both extensions by permitting composed bases inside the powered and ratio search. The Stage 2 expression remains available throughout validation-based selection. Appendix I specifies the bounded search families and selection procedure.

Variants and hyperparameters. We evaluate nine family variants. Standard RuleKAN uses GSR-style Stage 2 term selection (Sovrano et al., 2026): it fits each candidate term to the current residual, commits the validation-best term, and jointly refits selected terms. RuleKAN + OMP-Full changes only this term-selection procedure. Its fitted Stage 2 terms form an orthogonal matching pursuit (OMP) dictionary: each step selects the term with largest absolute residual correlation and jointly least-squares refits the selected amplitudes (Pati et al., 1993). The Stage 1 support bank, factor-level GMP screening (Sovrano et al., 2026), vocabulary, and set of allowed variable patterns are unchanged. RuleKAN-RBF replaces Stage 1 splines with Gaussian radial-basis edges (Li, 2024, FastKAN).

RuleKAN Adaptive may replace the primary bank $\mathcal{S}$ by selected supports from the pre-pruning bank $\mathcal{S}^+ \supseteq \mathcal{S}$ when a validation residual remains and validation error improves; the same bounded support augmentation then applies. RuleSISP instead enumerates all variable multisets through order q. RuleKAN-Comp adds composition, PowerRuleKAN adds powers and ratios, PowerRuleKAN-Comp combines both Stage 3 extensions, and RuleSISP-Comp adds composition to RuleSISP. Appendix D gives the upper bounds on search size and the validation-based selection rules.

3 Related Work

KAN-based symbolic methods. Several approaches modify either the numerical representation of a KAN or the way symbolic expressions are extracted from it. MultKAN introduces explicit multiplication nodes (Liu et al., 2025a), while FastKAN replaces spline edge functions with Gaussian radial basis functions (Li, 2024). SR-KAN uses separability and symmetry tests to decompose a trained KAN into simpler symbolic subproblems (Bühler & Guillén-Gosálbez, 2026); Symbolic-KAN instead embeds analytic functions and learned projection directions directly in the trainable network (Faroughi et al., 2026).

Structured symbolic representations. The sum–product formula used in Stage 2 is related to *separated representations*, which approximate multivariate functions by sums of products of univariate factors and have been developed for numerical approximation and regression (Beylkin et al., 2009).

Table 1: Fixed-budget costs for all proposed variants and selected baseline families. Per-edge KAN rows exclude baseline numerical training.

Method	Cost
RuleKAN, RuleKAN-RBF, RuleKAN + OMP-Full	$O(Nd + N_s K A_R)$
RuleKAN Adaptive	$O(Nd + N_s K A_A)$, $A_R \leq A_A \leq A_S$
RuleKAN-Comp	$O(Nd + N_s K A_R + J_c N_s + N)$
RuleSISP	$O(N_s K d^q)$
RuleSISP-Comp	$O(N_s K d^q + J_c N_s + N)$
PowerRuleKAN	$O(Nd + N_s K A_R + HNR_o^3 + HR_o^4)$
PowerRuleKAN-Comp	$O(Nd + N_s K A_R + J_c N_s + HNR_o^3 + HR_o^4 + N)$
GSR / GMP	$O(N_s K E)$ per-edge extraction
SR-KAN	$O(NK)$ under fixed architecture/grid budgets
Symbolic-KAN	$O(NKE)$ per gradient step
PySR / Operon	$O(NPS)$ per generation
SINDy (STLSQ)	$O(NM^2 + M^3)$ per threshold step
ParFam	$O(N)$ per objective evaluation under fixed family

Our setting differs in that the factors are selected from an analytic vocabulary, repeated occurrences of the same variable are represented explicitly, and the objective is symbolic recovery rather than numerical approximation.

General symbolic regression includes PySR and Operon (Cranmer, 2023; Burlacu et al., 2020), RILS-ROLS (Kartelj & Djukanović, 2023), ParFam (Scholl et al., 2025), EQL (Sahoo et al., 2018), and sparse-library identification via SINDy (Brunton et al., 2016). There are also methods specialized in fuzzy-rule modelling and extraction. Takagi–Sugeno models and ANFIS represent input–output relations through explicit fuzzy if–then rules (Takagi & Sugeno, 1985; Jang, 1993). Their interpretability follows from a predefined rule architecture, whereas our objective is to recover such a factorization within a general symbolic language that also represents products, powers, ratios, and nested expressions.

4 Computational complexity

We use the notation of Section 2: d input variables, q maximum factors per term, W numerical terms, K primitives, L primitives kept per factor, R_o powered terms, and E KAN edges subjected to per-edge symbolic extraction. Let N be the number of training observations, $N_s \leq N$ the subsample used by the Stage 2 relaxed phase, T_1, T_2 the numbers of gradient steps in Stages 1 and 2, and J_c the number of bounded composition families screened when composition is triggered. The bounds below hold the optimization budgets T_1, T_2 and per-primitive evaluation cost fixed while retaining K (vocabulary-size) and L; Appendix C gives exact counts.

Stage 1 evaluates W terms of at most q edge functions, each a soft weighting over d inputs, per observation and step: $O(T_1 NWqd)$, or $O(Nd)$ under fixed budgets.

Stage 2 costs are instead proportional to the number of candidate factor slots, meaning the total number of factor occurrences across all candidate structures. RuleSISP has $A_S = d\binom{d+q}{q-1}$ such slots, so its relaxed phase costs $O(T_2 N_s K A_S)$, that is $O(N_s K d^q)$ for fixed q: linear in the subsample, linear in vocabulary size, and polynomial of degree q in the number of input variables. RuleKAN instantiates only the $A_R \leq A_S$ factor slots of $\mathcal{G}_R$, giving $O(N_s K A_R)$. RuleKAN Adaptive may enlarge this to A_A factor slots with $A_R \leq A_A \leq A_S$ when pre-pruning supports are admitted; with five two-variable supports in $d = 10$, $q = 3$, ordinary RuleKAN gives a $19.5\times$ reduction (Table 4). The discrete phase then refits at most L^{m_r} primitive combinations per candidate term, exponential in the factor count if q and L are allowed to grow, which is why both are kept small.

Stage 3 may rerun Stage 2 on simple transformations of the target used to initialize reciprocal, square-root, or ratio candidates. It then selects among H candidate bases by refitting the powered expression after each addition, for $O(HNR_o^3 + HR_o^4)$ with dense normal equations (Appendix C). Composition screens J_c bounded depth-two families in $O(J_c N_s)$, followed by refinement of a fixed number of finalists, linear in N. Appendix I gives the dependence of J_c on outer and inner vocabulary sizes and admissible variable assignments.

Table 1 summarizes fixed-budget search and extraction costs, with P the evolutionary population size, S the maximum tree size, and M the SINDy library size. Per-edge extraction scales with KE. PySR and Operon evaluate a fixed population in $O(NPS)$ per generation; operator-vocabulary size does not multiply this fixed-population bound, but enlarges the recursive grammar's branching factor and combinatorial search space (Cranmer, 2023; Burlacu et al., 2020). SR-KAN permits its final-expression function library to be restricted under fixed architecture and grid budgets (Bühler & Guillén-Gosálbez, 2026).

5 Experimental Design and Benchmarks

Benchmarks, datasets, and provenance. Our evaluation uses 30 noise-free symbolic regression tasks over three seeds, chosen to isolate the targeted expressivity gaps and test transfer to KAN and physics-based problems. The first four suites are synthetic tests of the capacities in Section 2: 6 flat product tasks cover same- and cross-input factorizations and multi-term mixtures; 7 fuzzy tasks use Takagi–Sugeno-style rules with single, shared, nested, and product-valued branches (Takagi & Sugeno, 1985); 4 power/ratio tasks apply squares or reciprocals to multivariate bases; and 5 nested tasks add one analytic function around a unary or multivariate base. The fuzzy tasks retain their generating gate-and-branch structure to score prediction and structural recovery separately. Three additional tasks use canonical KAN examples: exponential-of-sum, logarithmic composition, and radial square-root targets (Liu et al., 2025b). And five use physics-style kinetic-energy, inverse-square, Gaussian, mixed-product, and resonance forms from the AI Feynman benchmark family (Udrescu & Tegmark, 2020). All target equations and sampling domains are given in Appendix A.

Methods. We compare 23 symbolic methods: 9 RuleKAN/RuleSISP variants and 14 external baselines spanning KAN-based extraction and general symbolic regression. We fix $q = 3$, the largest required interaction order. RuleKAN/RuleSISP and matched KAN controls use the common elementary vocabulary where supported; other methods retain the native parameterizations in Appendix A. To prevent a fixed overcomplete library from trading formula length directly for error, the main SINDy baseline is capped at 12 active non-bias terms (otherwise it would produce thousands since it tends to produce the same series expansions we want to avoid). Stage 1 and conventional KAN/MultKAN methods use total hidden width 12; SR-KAN has no directly comparable width control (Bühler & Guillén-Gosálbez, 2026).

Data splits and metrics. For each seed and task, inputs are sampled independently and uniformly over the bounded domain of the equation, yielding $1600/400/500$ train/validation/test observations. Training and model selection use only the training and validation sets, followed by one held-out test evaluation. We use two types of metrics: predictive error and structural recovery. For predictive accuracy, we report NRMSE (normalized root mean squared error) defined as test RMSE divided by $\mathrm{sd}(y_{\mathrm{test}})$, so errors are comparable across tasks with different output scales; within-task ablations use RMSE because no cross-task normalization is needed. Each task is summarized by the median over three seeds. For the seven fuzzy tasks, we also report expanded-rule F1 and exact structural recovery, since low predictive error need not imply recovery of the intended rule. F1 is computed after one-to-one matching of predicted and annotated gate-and-branch rules; exact recovery additionally requires equal rule counts and a complete match under the stated equivalences. Exact recovery is counted over the $21 = 7 \times 3$ runs. Appendix A gives matching tolerances and more details.

Statistical comparisons and failures. Pairwise predictive comparisons use two-sided Wilcoxon signed-rank tests on $\log_{10}(\mathrm{NRMSE}_A/\mathrm{NRMSE}_B)$ across the 30 failure-aware task medians. Benjamini–Hochberg controls the false discovery rate (FDR) across the 14 external baselines; q denotes the adjusted p-value. A win means lower task-median NRMSE, and mean rank averages the per-task ranks. Each job has a 2 400 second limit; failures receive the worst-error penalty in Appendix A. Fuzzy failures contribute F1 $= 0$ and exact structural recovery $= 0$; exact structural-recovery indicators use paired task-seed runs in exact McNemar tests with the same FDR correction.

Component ablation. To understand which ingredients of the proposed methods are most important, we perform a component ablation analysis. A 2^3 factorial over 14 tasks and three seeds (336 runs) independently toggles the Stage 2 sum-product grammar, Stage 1 iterative pruning, and Stage 2 GMP screening. For each component, the main-effect RMSE multiplier is geometric-mean RMSE when enabled divided by that when disabled, balanced over the four combinations of the other two components; values below one favor enabling it. Auxiliary variants are reported in Appendix H.

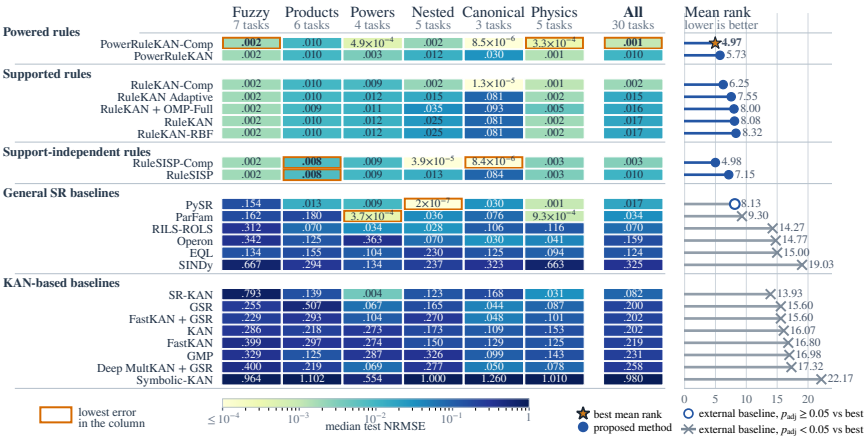

Figure 3: Predictive test NRMSE. Suite cells are medians across task-level seed medians. Rank markers identify FDR-controlled comparisons of the best proposed method with external baselines.

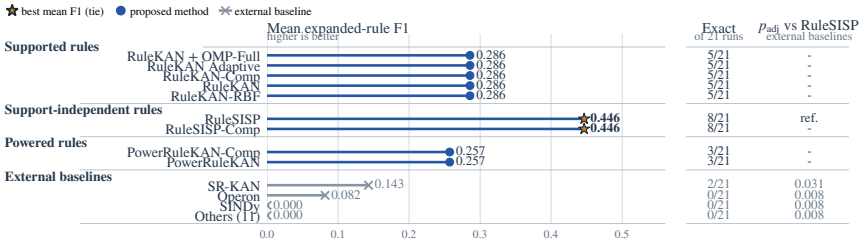

Figure 4: Fuzzy structural recovery. Bars show mean expanded-rule F1; rightmost values are FDR-adjusted exact McNemar tests for exact structural recovery against RuleSISP.

6 Results

Figure 3 reports predictive error and all-task rank; Figure 4 reports fuzzy rule structure. Across all 30 analytic tasks, PowerRuleKAN-Comp has the best mean rank (4.97) and the lowest median NRMSE (1.07×10^{-3}). It has significantly lower error than 13 of 14 external baselines after FDR correction; PySR remains non-significant ($q = 0.0879$), with lower task-median NRMSE on 20 of 30 tasks. Appendix Table 7 reports symbolic coverage and timeout counts; missing outputs are retained through the failure-aware evaluation rule.

In the component ablation, the median factorial RMSE multiplier is 0.0907 for the sum-product grammar, 0.995 for GMP, and 1.000 for iterative pruning (Appendix Figure 7 and Table 6). Moreover, compared with RuleKAN, RuleKAN-RBF has median RMSE ratio 1.00 and 4.9% lower runtime.

RuleSISP improves on RuleKAN for synthetic products (0.00772 versus 0.00961 median NRMSE) and nested targets (0.0128 versus 0.0251), while their fuzzy-suite predictive medians tie at 0.00189. RuleSISP-Comp reaches 3.89×10^{-5} on nested targets versus 0.00217 for RuleKAN-Comp, isolating the effect of Stage 1-conditioned support construction under a shared vocabulary and factor order. On the power suite, median NRMSE falls from 0.0118 for RuleKAN to 0.00336 for PowerRuleKAN and 4.85×10^{-4} for PowerRuleKAN-Comp. On nested targets, PowerRuleKAN-Comp reaches 0.00216, whereas PySR reaches 2.02×10^{-7}. The elementary vocabulary makes these gains depend on explicit scope and composition rather than predefined compound primitives.

Finally, both RuleSISP variants achieve exact structural recovery on $8/21$ task-seed runs, five Stage 1-conditioned RuleKAN variants on $5/21$, both PowerRuleKAN variants on $3/21$, SR-KAN on $2/21$, and the remaining baselines on none. RuleSISP's mean expanded-rule F1 is 0.446; versus PySR, exact recovery is $8/21$ versus $0/21$ with FDR-adjusted exact McNemar $q = 0.0084$ ($q \leq 0.031$ versus every external baseline). On the same-variable fuzzy rule, SR-KAN has median test NRMSE 5.09×10^{-4} but zero exact structural recovery.

7 DISCUSSION

The ablations suggest that the main bottleneck is representational rather than procedural. Because a KAN edge can absorb several analytic factors of one input (Appendix B), better selection alone cannot expose factors hidden inside that edge. The much larger benefit of the sum-product grammar than of GMP or iterative pruning therefore points to representation: explicit products expose multiplicative factors and fuzzy gate-and-branch structure that flexible univariate components can absorb (RQ1). RuleSISP reveals a separate tradeoff. Searching beyond learned Stage 1 supports protects against omitted interactions, while RuleKAN reduces search cost but can miss an interaction removed during pruning (RQ2).

Once factorization and interaction coverage are sufficient, two grammar limits remain: *scope*, when an operator must act on a complete multivariate expression, and *depth*, when one recovered expression becomes the input to another function. Powers address scope; composition addresses depth. Because the vocabulary contains only elementary unary primitives, these gains cannot come from predefined shortcuts such as e^{-x^2} or $\sqrt{1+x^2}$. The resulting hierarchy is concrete: factors define the symbolic atoms, Stage 1-conditioned support construction controls which variables may interact, whole-expression operators control scope, and composition controls nesting (RQ3).

Expression compactness is also part of symbolic recovery. Evolutionary methods like SINDy can fit targets through long additive expansions rather than compact factorized formulas. For example, with the same order-3 library, unconstrained (i.e., uncapped active non-bias terms) SINDy lowers median task NRMSE from 0.325 to 0.00167 but expands the median expression from 12 to 672 active terms, with some targets reaching thousands of terms. We therefore use SINDy in the main comparison and analyze the unconstrained expansion in Appendix H.

The fuzzy-rule experiments separate predictive accuracy from structural recovery. RuleSISP and RuleKAN variants recover annotated gate-and-branch structure that many baselines miss even when predictive errors are small. For example, on the *same-variable* rule, SR-KAN reaches median test NRMSE 5.09×10^{-4}, but its trigonometric surrogate contains neither the complementary gate nor the tanh branch. Factor order also affects RuleKAN: across four fuzzy tasks evaluated with both $q = 2$ and $q = 3$, mean task-median expanded-rule F1 decreases from 0.629 to 0.250, and exact recovery from $6/12$ to $3/12$, because the larger factor-order bound can introduce redundant factors (Appendix H). NRMSE therefore measures functional agreement, while the structural metrics measure recovery of the specified generating rule (RQ4).

8 LIMITATIONS

Structural scores are defined relative to the annotated factorization, vocabulary, factor-order, and symbolic-term bounds, so prediction-equivalent formulas can score differently. We therefore report structure only on the seven annotated fuzzy tasks and keep NRMSE separate. PySR, Operon, and the other evolutionary baselines permit restricted operator sets, but their evaluated recursive tree grammars contain additional unary operators; this can favor them when a target subexpression is directly available. Appendix A lists these operators and the broader-vocabulary sensitivity for the KAN-based methods. Statistical resolution is also limited by three seeds per task, which motivates task-level medians with FDR control. Finally, the evaluation suites are noise-free and use modest dimension and interaction order. They establish the targeted expressivity and support-selection effects under controlled conditions, but do not by themselves characterize recovery under noise, large d, large q, or unrestricted expression trees.

9 CONCLUSION

KAN symbolic extraction can fail when an edge absorbs several factors, the required interaction is absent from the learned support bank, or the target requires whole-expression operators or greater composition depth. Treating these as separate bounded capacities yields the best mean rank among 23 methods across all 30 analytic tasks, while RuleSISP recovers $8/21$ annotated fuzzy rules and PySR recovers none. The central implication is that predictive accuracy does not certify recovery of the generating structure. Future work could characterize the minimum rank, factor order, and composition depth required by broader function classes and by noisy, higher-dimensional data.

AI USE STATEMENT

A large language model was used for proofreading, consistency checks, and suggestions about exposition and literature coverage. It did not run the reported benchmark experiments. The authors reviewed the AI-generated text and take responsibility for the submission.

REPRODUCIBILITY STATEMENT

Additional implementation details, hyperparameters, and exact experimental counts are provided in the appendix. Code, benchmark definitions, data-processing scripts, and instructions for reproducing the reported results are available at https://anonymous.4open.science/r/RuleKAN-267B

REFERENCES

Gregory Beylkin, Jochen Garcke, and Martin J Mohlenkamp. Multivariate regression and machine learning with sums of separable functions. *SIAM Journal on Scientific Computing*, 31(3):1840–1857, 2009.

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, 2016.

Marco Andrea Bühler and Gonzalo Guillén-Gosálbez. Sr-kan: A kolmogorov-arnold network guided symbolic regression framework. *Computers & Chemical Engineering*, pp. 109721, 2026.

Bogdan Burlacu, Gabriel Kronberger, and Michael Kommenda. Operon C++: An efficient genetic programming framework for symbolic regression. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pp. 1562–1570. Association for Computing Machinery, 2020. doi: 10.1145/3377929.3398099. URL https://doi.org/10.1145/3377929.3398099.

William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabrício Olivetti de França, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason H. Moore. Contemporary symbolic regression methods and their relative performance. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/c0c7c76d30bd3dcaefc96f40275bdc0a-Abstract-round1.html.

Miles Cranmer. Interpretable machine learning for science with PySR and SymbolicRegression.jl. *arXiv preprint arXiv:2305.01582*, 2023. doi: 10.48550/arXiv.2305.01582. URL https://arxiv.org/abs/2305.01582.

Salah A Faroughi, Farinaz Mostajeran, Amirhossein Arzani, and Shirko Faroughi. Symbolic–kan: Kolmogorov-arnold networks with discrete symbolic structure for interpretable learning. *Journal of Computational Physics*, pp. 115223, 2026.

Jyh-Shing Roger Jang. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):665–685, 1993. doi: 10.1109/21.256541.

Aleksandar Kartelj and Marko Djukanović. Rils-rols: Robust symbolic regression via iterated local search and ordinary least squares. *Journal of Big Data*, 10(71), 2023. doi: 10.1186/s40537-023-00743-2.

Ziyao Li. Kolmogorov–arnold networks are radial basis function networks. *arXiv preprint arXiv:2405.06721*, 2024. doi: 10.48550/arXiv.2405.06721. URL https://arxiv.org/abs/2405.06721.

Ziming Liu, Max Tegmark, Pingchuan Ma, Wojciech Matusik, and Yixuan Wang. Kolmogorov–arnold networks meet science. *Physical Review X*, 15(4):041051, 2025a. doi: 10.1103/4t7t-v19l. URL https://doi.org/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. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/afaed89642ea100935e39d39a4da602c-Paper-Conference.pdf.

Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through l_0 regularization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018. URL https://openreview.net/forum?id=H1Y8hhg0b.

Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar conference on signals, systems and computers*, pp. 40–44. IEEE, 1993.

Subham Sahoo, Christoph Lampert, and Georg Martius. Learning equations for extrapolation and control. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4442–4450. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/sahoo18a.html.

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

Philipp Scholl, Katharina Bieker, Hillary Hauger, and Gitta Kutyniok. Parfam – (neural guided) symbolic regression via continuous global optimization. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 81527–81565, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ca98452d4e9ecbc18c40da2aa0da8b98-Paper-Conference.pdf.

Francesco Sovrano, Lidia Losavio, Giulia Vilone, and Marc Langheinrich. In-context symbolic regression for robustness-improved kolmogorov–arnold networks. *arXiv preprint arXiv:2603.15250*, 2026. doi: 10.48550/arXiv.2603.15250. URL https://arxiv.org/abs/2603.15250.

Tomohiro Takagi and Michio Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(1):116–132, 1985. doi: 10.1109/TSMC.1985.6313399.

Silviu-Marian Udrescu and Max Tegmark. Ai feynman: A physics-inspired method for symbolic regression. *Science advances*, 6(16):eaay2631, 2020.

APPENDIX ROADMAP

Section	Contents
A	Benchmark catalogue, evaluation protocol, vocabulary controls, broader-vocabulary sensitivity, and structural metrics.
B	Representation proofs for fuzzy-tree expansion, reciprocal bilinear targets, positive powers, and same-variable factorization.
C	Exact grammar counts and computational accounting for Stage 1-conditioned and support-independent search.
D	Structural scoring conventions and hyperparameters.
E	Analytic-factorization assumptions, canonical parameterizations, symmetries, complements, and real-domain checks.
F	Interaction proposals, fuzzy-rule refactoring, joint fitting, and cleanup.
H	Detailed predictive and structural results, factorial ablation, composition analysis, task-level results, and statistical tests.
I	Composition-search families, Stage 1-conditioned support construction, powered compositions, computational cost, and validation.

A BENCHMARK CATALOGUE AND EVALUATION PROTOCOL

The 30 symbolic regression tasks of the evaluation setup are grouped into six benchmark families. The flat product family covers same- and cross-input factorizations, including repeated variables and multi-term mixtures; the fuzzy family uses Takagi–Sugeno-style rules with single, shared, nested, and product-valued branches; the power/ratio family applies squares or reciprocals to multivariate bases; and the nested family adds one analytic function around a unary or multivariate base. The canonical KAN family uses standard KAN function-fitting examples, while the physics family uses compact equations from the AI Feynman benchmark family. Every analytic task is noise-free and uses three independent seeds with $1600/400/500$ train/validation/test observations. Table 2 gives every target and sampling domain.

Table 2: Complete analytic benchmark catalogue. Intervals are sampled independently and uniformly for each input.

Suite	Benchmark	Target $f(x)$	Domain
Products	Same-variable exponential-sine product	$0.8e^{-0.7x_0}\sin(2.4x_0)$	$x_0 \in [-1.5, 1.5]$
Products	Cross-variable log-cosine product	$0.75\log(1+x_0^2)\cos(1.7x_1)$	$x_0 \in [-2,2]$, $x_1 \in [-\pi,\pi]$
Products	Four-mechanism mixture	$1.3e^{-0.9x_0}\sin(2.2x_1)+0.75\log(1+x_2^2)\cos(1.6x_3)+0.25\tanh(2x_4)+0.4e^{-0.65x_4}\sin(2.35x_4)$	$x_0 \in [0,2]$, $x_{1,3} \in [-\pi,\pi]$, $x_2 \in [-2,2]$, $x_4 \in [-1.5,1.5]$
Products	Three-way analytic product	$0.9\sin(2x_0)e^{0.55x_1}\sqrt{1+(1.2x_2)^2}$	$[-1.5,1.5]^3$
Products	Two same-variable mechanisms	$0.3\tanh(2x_0)+0.55e^{-0.6x_0}\sin(2.5x_0)$	$x_0 \in [-1.5,1.5]$
Products	Rational and radical mixture	$0.9\cos(2x_0)/(1+1.5x_1^2)+0.55\sqrt{1+x_2^2}\tanh(1.7x_3)+0.2\sin(3x_4)$	$[-1.5,1.5]^5$
Fuzzy	Cross-variable branches	$(1-x_0)0.70\sin(2.1x_1)+x_0\,1.10e^{-0.75x_2}$	$x_0 \in [0,1]$, $x_{1,2} \in [-1.5,1.5]$
Fuzzy	Same-variable branches	$(1-x_0)0.65\sin(2.4x_0)+x_0\,0.85\tanh(2x_0)$	$x_0 \in [0,1]$
Fuzzy	Shared branch variable	$(1-x_0)0.80\cos(1.8x_1)+x_0\,0.55e^{-0.60x_1}$	$x_0 \in [0,1]$, $x_1 \in [-1.5,1.5]$
Fuzzy	Nested fuzzy tree	$(1-x_0)[(1-x_1)0.60\cos(1.7x_2)+x_1\,0.90e^{-0.80x_3}]+x_0\,0.50\tanh(1.9x_4)$	$x_{0,1} \in [0,1]$, $x_{2:4} \in [-1.5,1.5]$
Fuzzy	Product-valued branches	$(1-x_0)0.80e^{-0.45x_1}\sin(2.2x_2)+x_0\,0.60\sqrt{1+(1.15x_3)^2}\cos(1.55x_4)$	$x_0 \in [0,1]$, $x_{1:4} \in [-1.5,1.5]$
Fuzzy	Same-gate product branches	$(1-x_0)0.70e^{-0.55x_0}\sin(2x_1)+x_0\,0.65\tanh(2.1x_0)\cos(1.45x_2)$	$x_0 \in [0,1]$, $x_{1,2} \in [-1.5,1.5]$
Fuzzy	Two independent fuzzy rules	$0.70[(1-x_0)\sin(2x_2)+x_0e^{-0.70x_3}]+0.40[(1-x_1)\cos(1.5x_4)+x_1\tanh(1.8x_5)]$	$x_{0,1} \in [0,1]$, $x_{2:5} \in [-1.5,1.5]$
Powers/ratios	Reciprocal bilinear expression	$(1+0.8x_0x_1)^{-1}$	$[-0.8,0.8]^2$
Powers/ratios	Reciprocal quadratic sum	$(1+0.6x_0^2+0.4x_1^2)^{-1}$	$[-1,1]^2$

Suite	Benchmark	Target $f(x)$	Domain
Powers/ ratios	Squared affine sum	$(0.8 + 0.4x_0 + 0.3x_1)^2$	$[-1, 1]^2$
Powers/ ratios	Product-over-bilinear ratio	$(0.7 + 0.4x_0)(1 + 0.6x_0x_1)^{-1}$	$[-0.8, 0.8]^2$
Nested	Sine of exponential	$\sin(e^{0.7x_0})$	$x_0 \in [-1.5, 1.5]$
Nested	Tanh of sine	$\tanh(1.6\sin(2.2x_0))$	$x_0 \in [-1.5, 1.5]$
Nested	Sine of cross-variable product	$\sin(1.3x_0x_1) + 0.2x_2^2$	$[-1.5, 1.5]^3$
Nested	Mixed nested products	$0.7e^{-0.5x_0}\sin(2.1x_1) + 0.4\tanh(1.4\sin(2x_2))/(1 + x_3^2) + 0.35e^{-0.7x_4}\sin(2.4x_4)$	$[-1.5, 1.5]^5$
Nested	Cross-variable oscillator	$\sin(2.5\sin(1.6x_0) + 0.7\cos(1.3x_1))$	$[-\pi, \pi]^2$
Canonical KAN	Exponential-sine-square	$\exp[\sin(\pi x_0) + x_1^2]$	$[-1, 1]^2$
Canonical KAN	Logarithmic composition	$\sin[2(\log x_0 + \log x_1)]$	$[0.2, 5]^2$
Canonical KAN	Radial square root	$\sqrt{x_0^2 + x_1^2}$	$[-1, 1]^2$
Physics	Kinetic-energy form	$0.5x_0x_1^2$	$x_0 \in [0.2, 3]$, $x_1 \in [-2, 2]$
Physics	Inverse-square form	x_0x_1/x_2^2	$[0.5, 2]^3$
Physics	Gaussian density	$e^{-x_0^2/2}/\sqrt{2\pi}$	$x_0 \in [-3, 3]$
Physics	Lorentz-style mixed products	$x_0x_1 + x_0x_2x_3$	$[-2, 2]^4$
Physics	Resonance denominator	$[(x_1^2 - x_0^2)^2 + (x_2x_0)^2]^{-1}$	$x_0 \in [0.2, 2]$, $x_1 \in [0.5, 2]$, $x_2 \in [0.1, 1]$

Elementary vocabulary and comparison controls. The common KAN vocabulary contains exactly ten elementary univariate primitives: identity, square, reciprocal, inverse square, square root, logarithm, exponential, sine, cosine, and hyperbolic tangent. Each primitive acts on an affine argument, and real-valued domain restrictions remain part of its definition. SR-KAN receives the same ten forms, using its native aliases for identity, square, reciprocal, and inverse square. Compound shortcuts are not primitives. Consequently, the Gaussian density requires exponential-of-square composition; the cross-variable log-cosine and four-mechanism mixture targets require $\log(1 + x^2)$; the rational and radical mixture requires both $(1+x^2)^{-1}$ and $\sqrt{1 + x^2}$; the three-way product requires $\sqrt{1 + x^2}$; and the mixed nested-products target requires $(1 + x^2)^{-1}$.

PySR and Operon search recursive expression trees with their specified arithmetic and analytic operators, so composition is intrinsic to their grammars. RILS-ROLS uses its native grammar; ParFam uses polynomial/rational families with sine, cosine, exponential, logarithm, square root, and hyperbolic tangent; EQL uses the matched unary units plus learned multiplication. SINDy uses the ten elementary forms in a fixed feature library with interactions through order 3 and at most 12 active non-bias terms; the identical uncapped library is reported only as a capacity sensitivity below. SymbolicKAN uses its closest native bank $\{x, x^2, 1/x, \sqrt{|x| + \epsilon}, \log(|x| + \epsilon), \exp, \sin, \cos, \tanh\}$, which has no direct inverse-square primitive. SR-KAN retains its native decomposition and simplification while its final-expression library is restricted to the ten shared forms. The conventional KAN/MultKAN controls use total hidden width 12; SR-KAN has no directly comparable hidden-width control.

Failure-aware predictive comparison. Failed runs, including timeouts and non-finite symbolic outputs, are never omitted from the analytic comparison. Because NRMSE is lower-is-better, zero success credit is implemented as a worst-error penalty rather than numerical NRMSE zero: for each task, a failed run receives NRMSE equal to ten times the largest finite run-level NRMSE observed on that task. The median is then taken over the three seeds, supplying the 30 task medians used for mean ranks and pairwise tests in Section 5. This preserves ordinary NRMSE values for completed runs while ensuring that failure cannot improve a score or remove a difficult task.

Broader-vocabulary sensitivity. The main benchmark uses only the ten elementary primitives above, so compound forms must be constructed by composition. To test how strongly this restriction affects the KAN-based methods, a separate sensitivity comparison adds e^{-x^2}, $\log(1 + x^2)$, $\sqrt{1 + x^2}$, and $(1 + x^2)^{-1}$ as single primitives and restores SR-KAN's full native function library. These functions directly match recurring subexpressions in the benchmark catalogue and therefore reduce the amount of composition required during symbolic recovery. Table 3 reports the corresponding

all-task median NRMSE under the same failure-aware protocol as the main benchmark. SR-KAN benefits substantially, while PowerRuleKAN-Comp remains lower in median NRMSE for both vocabulary choices.

Table 3: Broader-vocabulary sensitivity on all 30 analytic tasks for the two affected methods. The elementary row is the main benchmark. The broader row adds the four compound primitives listed above to the KAN vocabulary and restores SR-KAN's full native function library. Lower median NRMSE is better.

Vocabulary	SR-KAN	PowerRuleKAN-Comp
10 elementary primitives	0.0820	1.07×10^{-3}
Broader vocabulary	0.0134	7.73×10^{-4}

Structural metrics. Multiset-aware structural precision and recall use one-to-one matches between predicted and reference rule multisets, with their harmonic mean defining structural F1. A rule with the right support but the wrong repeated-variable count is a different multiset. Fuzzy matching additionally checks factor roles and membership orientation: orientation $+1$ denotes g and orientation -1 denotes $1 - g$. Native RuleKAN-family models are scored from their discrete factors. Branch factors must match variable and primitive family, with sine and cosine treated as phase-equivalent; a gate must be an identity factor on the correct variable and, after optimal scalar rescaling, have held-out gate NRMSE at most 0.08 against g or $1 - g$. Powered outer terms with absolute coefficient at most 5×10^{-7} are treated as numerical residue. Exact structural recovery requires equal predicted and reference rule counts and a complete one-to-one rule matching.

External symbolic formulas are converted to raw input coordinates, expanded into bounded disjunctive normal form, and canonicalized without refitting to target values. An affine gate receives orientation only when its scale-invariant NRMSE on a fixed grid over the task domain is at most 0.12; a coefficient is canonicalized to a product of unit-interval memberships only when its relative RMS residual is at most 0.03. Unsupported atoms remain unmatched. Failed structural evaluations receive F1 $= 0$ and exact structural recovery $= 0$. The 0.08 native-gate and 0.12 external-canonicalization tolerances apply to different checks: the former scores a held-out fitted gate, whereas the latter determines whether an arbitrary external formula can be assigned a gate orientation during canonicalization.

B Representation results

Fuzzy-tree expansion. At each internal node, a fuzzy tree combines its children as $(1 - g_v)f_{v,0} + g_v f_{v,1}$. Repeated substitution gives

$$f(x) = \sum_{\ell} \left[\prod_{v \in P_\ell} g_v(x_{j_v})^{b_{v\ell}} (1 - g_v(x_{j_v}))^{1 - b_{v\ell}} \right] f_\ell(x),$$

where P_ℓ is the path to leaf ℓ and $b_{v\ell}$ indicates the selected branch. If each leaf contains k_ℓ admissible factors and each membership or complement is one admissible factor, leaf ℓ requires at most $|P_\ell| + k_\ell$ factors. Thus exact expansion is available when the corresponding supports are searched and $q \geq \max_\ell(|P_\ell| + k_\ell)$.

Reciprocal bilinear expressions. The target $(1 + cxy)^{-1}$ has no finite representation $\sum_{r=1}^{R} u_r(x)v_r(y)$ on a nondegenerate product domain. Choose $R + 1$ distinct nonzero values x_i. If

$$\sum_i \frac{\alpha_i}{1 + cx_i y} = 0,$$

multiplication by $\prod_i (1 + cx_i y)$ gives a polynomial identity. Evaluating it at $y = -1/(cx_j)$ forces $\alpha_j = 0$ for every j, so the $R + 1$ slices are linearly independent, contradicting membership in an R-dimensional span. Hence a finite flat sum-product model cannot represent this target exactly, whereas PowerRuleKAN can represent it by applying exponent -1 to the base $1 + cxy$.

Positive powers. For $B = b + \sum_{r=1}^{R} t_r$ and positive integer p, the multinomial theorem expands B^p into finitely many product terms. If each t_r contains at most q factors, each expanded term contains

at most pq factors. Positive powers therefore improve compactness under bounded search budgets but do not enlarge the unbounded finite sum-product closure.

Same-variable factorization. If a numerical edge represents $h(x) = \prod_j g_j(x)$, observations of the unrestricted edge identify only h, not its individual factors. Recovering the separate factors therefore requires additional restrictions on the symbolic candidate family. This motivates the repeated-variable expansion used in Stage 2.

C GRAMMAR IDENTITIES AND COMPUTATIONAL ACCOUNTING

For all variable multisets of order m, stars and bars gives $\binom{d+m-1}{m}$ structures. Their cumulative factor count is $A_S = \sum_{m=1}^{q} m\binom{d+m-1}{m} = d\sum_{m=1}^{q}\binom{d+m-1}{m-1} = d\binom{d+q}{q-1}$. For a fixed nonempty support containing s distinct variables, an order-m term must use every supported variable at least once. The m factor slots can therefore be distributed among the s variables in $\binom{m-1}{s-1}$ ways. Therefore $\sum_{m=s}^{q}\binom{m-1}{s-1} = \binom{q}{s}$, and $\sum_{m=s}^{q} m\binom{m-1}{s-1} = s\sum_{m=s}^{q}\binom{m}{s} = s\binom{q+1}{s+1}$. Summing the second expression over the primary learned supports yields A_R; bounded rescue additions are accounted for separately in Appendix F. These identities count canonical variable multisets. Independent additive components with identical multisets are additional model instances, not additional distinct structures.

Table 4 gives a concrete count example independent of empirical performance. With $d = 10$ and $q = 3$, unrestricted enumeration yields 285 variable patterns containing 780 factor slots in total. If Stage 1 retains only five distinct two-variable supports, the corresponding RuleKAN search has 15 variable patterns and 40 factor slots. Thus one complete primitive-screening pass evaluates 19.5 times fewer factor slots when the per-primitive evaluation cost is the same.

Table 4: Exact counts of candidate variable patterns for ten inputs and maximum order three. The RuleKAN example assumes that Stage 1 retained five two-variable supports.

Grammar	Variable patterns	Factor slots
All multisets	285	780
Five learned pair supports	15	40

Cost of selecting powered bases. At round r, each prospective powered base extends the current selection to r columns. Rebuilding the dense normal equations costs $O(Nr^2)$; their direct solution costs $O(r^3)$; validation adds $O(N_v r)$. With at most H trials per round, total work is $O\Big(H\sum_{r=1}^{R_o}(Nr^2 + r^3 + N_v r)\Big) = O(HNR_o^3 + HR_o^4 + HN_vR_o^2)$. The main-text bound takes training and validation sizes of the same order. Caching all base predictions requires $O((N + N_v)H)$ space. Shared Gram matrices or incremental factorizations can reduce repeated work, but the direct bound matches the trial-by-trial dense fitting procedure. Base training additionally stores the parameters and activations described in Section 4.

For a fixed symbolic expression with F factor evaluations, bounded-cost analytic functions give $O(NF)$ forward time. Reverse-mode differentiation has the same arithmetic order for a fixed computation graph. If exponents are not bounded, exponentiation by squaring adds $O(\log|p|)$ multiplications per powered base, plus division for negative powers. Vocabulary size enters twice: relaxed KAN screening evaluates K primitives per factor position, while an order-m discrete variable pattern admits up to K^m primitive assignments before shortlisting. For recursive tree grammars, the operator vocabulary analogously increases the branching factor even when the per-tree evaluation cost is unchanged.

D ADDITIONAL EVALUATION DETAILS

Final powered expressions and fuzzy scores. Structural scoring first sets powered outer coefficients $|a| \le 5 \times 10^{-7}$ to zero. A single base at power one expands into its product terms; other powered terms, reciprocals, composed bases, and products of bases count as unmatched fuzzy rules. This metric distinguishes an explicit fuzzy-rule decomposition from a prediction-equivalent powered

formula. Ordinary-formula checks also compare domain-sensitive analytic functions with their protected numerical evaluations on the evaluation inputs.

Practical hyperparameter selection. Table 5 summarizes the structural controls introduced by the proposed families. They are upper bounds on search size rather than unconstrained accuracy parameters. The product order q is especially important because RuleSISP's grammar grows as $\Theta(d^q)$ for fixed q. The numerical width W should be large enough to preserve interaction recall. The final symbolic model may still contain more additive terms than Stage 1 retains numerical terms, because Stage 1 passes variable supports rather than fixing the number of symbolic terms used for each support. Vocabulary size K should reflect plausible analytic functions; adding distractors increases both relaxed screening and discrete candidate search. The conventional KAN/MultKAN controls use total hidden width 12. SR-KAN has no directly comparable hidden-width control in its public regressor; its intermediate single-layer KAN search is specified through sum/multiplication combination types and spline grid counts.

Table 5: Practical structural hyperparameters and selection rules.

Control	Role	Selection rule
W	Numerical term width	Increase until validation fit and the pre-pruning support set both stabilize across width increases and restarts; the final number of symbolic terms is selected later.
Pruning	Stage 1 compression	Iteratively remove numerical terms or factor slots whose gates round off, then refit the survivors.
Adaptive support extension	Pre-pruning support search	If the ordinary symbolic fit leaves a validation residual, test supports from the pre-pruning bank $\mathcal{S}^+$ without enumerating all supports.
q	Maximum factors per rule	Use the smallest plausible interaction order; increase only when residual interactions remain.
K	Analytic vocabulary size	Include scientifically plausible analytic functions and a small number of distractors; avoid broad vocabularies without evidence.
L	GMP shortlist per factor	Keep small enough that the L^q discrete primitive combinations remain tractable; enlarge when operator recall is poor.
D	Integer power set	Include 1 and only needed signed powers. The benchmark default is $\{-3, -2, -1, 1, 2, 3\}$.
R_o	Maximum powered outer terms	Start from one or two bases and increase only when validation residuals remain structured.
Composition gate	When to add depth	Trigger only after the flat model remains above a validation-error threshold and require a positive validation improvement.

RuleKAN-family methods with Stage 1 use $W = 12$ numerical terms, and the matched KAN baselines use total hidden width 12; the remaining shared limits are grid 12 and $R_{\max} = 12$ final additive symbolic terms before allowing additional terms. Composition is attempted above validation NRMSE 0.003 and accepted only after at least 0.2% relative validation mean squared error (MSE) improvement. Reciprocal candidates must maintain a selection-domain denominator margin of at least 0.02.

E Canonicalization and domain checks

Equivalent parameterizations. Analytic factors can have functionally equivalent parameterizations: scale may move between a factor and its rule coefficient, sine and cosine differ by phase, odd and even functions admit sign symmetries, and exponential shifts may be absorbed into rule scale. We therefore use family-aware multistart fitting and canonicalization so equivalent parameterizations are not treated as distinct symbolic forms. Affine parameters are transformed consistently between standardized and original input coordinates; complementary gates share one affine parameterization so their sum remains exactly one.

Domain checks. During fitting, protected versions of domain-sensitive operators keep optimization finite, for example by clamping invalid square-root or logarithm arguments. A recovered expression is therefore accepted as an ordinary analytic formula only when its factors are valid over the data

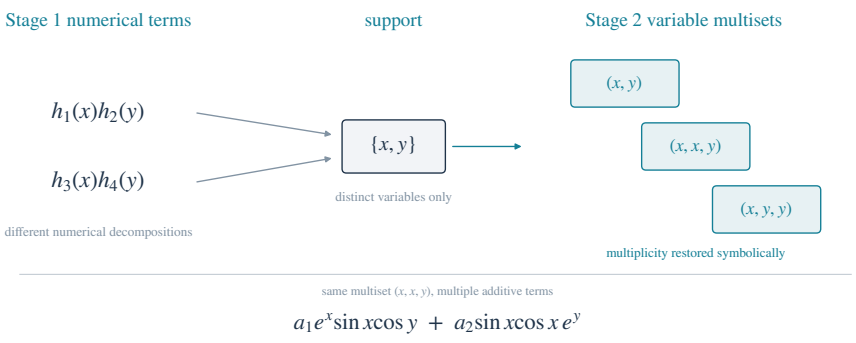

Figure 5: Stage 1 retains only the set of distinct variables in each interaction. Stage 2 then tries repeated occurrences of those variables and may use several additive terms with the same variable pattern.

domain and protected and ordinary evaluations agree there. Reciprocal bases additionally require a nonzero magnitude margin on both selection and test inputs.

F Interaction proposals and fuzzy-rule refactoring

Before fitting validation-gated rescue candidates, RuleKAN constructs an effective support bank from Stage 1 evidence. Downward closure adds nonempty subsets of retained supports, while gate-aware recombination joins support classes only when they share a non-gate branch variable, contain distinct membership-like gate variables, and remain within the maximum factor order q. This permits targeted recovery of supports lost during Stage 1 pruning without enumerating arbitrary missing supports. RuleSISP requires no such augmentation because it enumerates all variable multisets through order q.

For supported variable pairs, RuleKAN estimates a two-dimensional response surface and removes its row and column main effects. A nearly rank-one residual interaction suggests a separable product mechanism; its leading singular vectors provide candidate univariate factors, which are then matched to analytic families and evaluated end-to-end. Selected two-rule blocks are jointly refit so complementary mechanisms can be assessed together rather than independently.

Fuzzy-rule refactoring searches pairs of order-two structures sharing a gate variable. Candidates that are predictively near-tied on validation data are compared using cancellation and symbolic description cost, favoring compact complementary-rule representations over equivalent distributive forms.

Two bounded rescue mechanisms extend this construction. Product-partition rescue allows each branch of a complementary gate pair to contain a product of analytic factors drawn from the effective support bank. Recursive-partition rescue permits a depth-two partition with two gates and three unary branch functions. Both are attempted only when the incumbent validation error exceeds a fixed trigger and are accepted only when they improve validation performance.

After each accepted rule, backfitting re-estimates the selected amplitudes and factor parameters against the updated residual. Final cleanup removes unnecessary terms only after refitting the resulting expression.

G Supports, repeated variables, and search growth

Figure 5 separates what Stage 1 retains from what Stage 2 is allowed to reconstruct. A support records only the distinct variables, for example $\{x, y\}$. A variable's multiplicity is simply how many times it appears in a pattern, so (x, x, y) uses x twice and y once. Several additive terms may use the same pattern. The example depicts possible symbolic terms rather than a recovered benchmark equation.

Figure 6 illustrates the combinatorial reduction obtained by conditioning Stage 2 on a retained support. RuleSISP enumerates every multiset of up to q factor slots over all d input variables, yielding

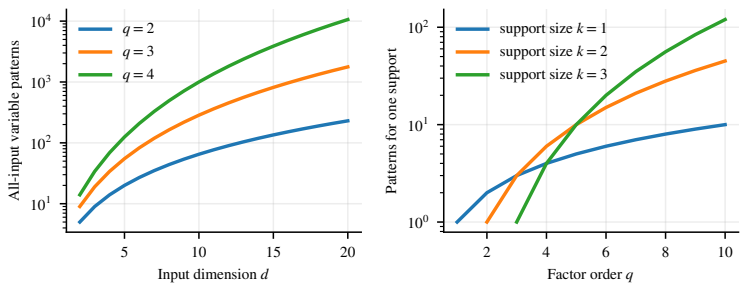

Figure 6: Support conditioning reduces the variable-pattern search space before analytic primitives are assigned. Left: without support conditioning, RuleSISP enumerates patterns over all d input variables, so the number of candidates grows rapidly with input dimension, particularly at larger maximum factor order q. Right: once RuleKAN fixes a support of k distinct variables, the search is restricted to repetition patterns within that support; the candidate count therefore depends on k and q, rather than on the full input dimension d. Both vertical axes use logarithmic scales.

$\binom{d+q}{q} - 1$ candidate variable patterns. For a RuleKAN support containing k distinct variables, the search is restricted to repetition patterns over those variables, giving only $\binom{q}{k}$ patterns through order q. After an order-m variable pattern is selected, assigning one of K analytic primitives to each factor introduces at most K^m primitive combinations before shortlisting. These counts characterize the nominal search space, not runtime.

H Detailed empirical comparisons

Fuzzy structural comparisons. The failure-aware structural analysis counts timeouts and failed structural evaluations as F1 $= 0$ and exact structural recovery $= 0$. The five Stage 1-conditioned RuleKAN variants each have mean expanded-rule F1 0.286 and exact recovery on $5/21$ task-seed runs. Both RuleSISP variants have mean F1 0.446 and recover $8/21$ exactly; both PowerRuleKAN variants recover $3/21$. SR-KAN has mean F1 0.143 with exact recovery on $2/21$, Operon has mean F1 0.082 with no exact recoveries, and the remaining external baselines have neither. Exact McNemar comparisons with RuleSISP give FDR-adjusted $q \leq 0.031$ for all 14 external baselines.

The same-variable task illustrates why this structural criterion is needed. Its generating rule is $(1 - x)\, 0.65 \sin(2.4x) + x\, 0.85 \tanh(2x)$. SR-KAN obtains test NRMSE 5.09×10^{-4}, 1.76×10^{-4}, and 5.28×10^{-4} over seeds 0, 1, and 2, respectively, but exact structural recovery is zero for all three. These are single-task values, not the seven-task fuzzy-suite aggregate in Figure 3. Its recovered formulas are trigonometric surrogates, including a sum of two sines for seed 1 and linear-plus-trigonometric forms for seeds 0 and 2, without the explicit complementary gate and tanh branch. The predictive score therefore records functional approximation, while the structural score records failure to recover the annotated rule.

Factor-order sensitivity. The main benchmark uses maximum factor order $q = 3$. For RuleKAN, four fuzzy tasks were also evaluated with $q = 2$ (the minimum for these tasks) to examine the effect of a smaller factor-order bound. Task-median expanded-rule F1 changes from 1.000 to 0.000 on the cross-variable rule, 1.000 to 1.000 on the same-variable rule, 0.182 to 0.000 on the shared-branch-variable rule, and 0.333 to 0.000 on the two-independent-rules task when moving from $q = 2$ to $q = 3$. Across these four tasks, mean task-median F1 decreases from 0.629 to 0.250, while exact recovery decreases from $6/12$ to $3/12$. On the cross-variable rule, the $q = 3$ runs retain gate recall $= 1$ and branch recall $= 1$ for all three seeds, but additional near-constant or redundant factors reduce strict expanded-rule F1 to zero. Thus, increasing the maximum factor order can over-factorize an otherwise correct rule.

Component ablation. The core 2^3 design defined in Section 5 contains 336 runs: 14 tasks, three seeds, and eight factorial cells. The ablation table also contains six auxiliary one-at-a-time variants over the same 42 task-seed cases, giving 588 runs in total; these auxiliary variants are not included in the factorial main effects. Figure 7 and Table 6 summarize the three factorial RMSE multipliers. A multiplier below one favors enabling the component.

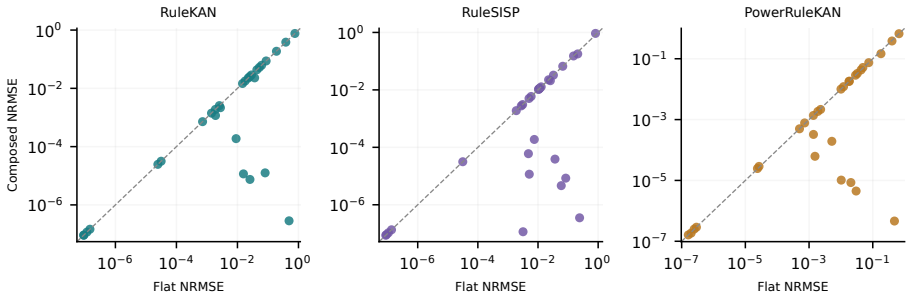

Figure 8: Flat versus composed task-median NRMSE; points below the diagonal favor composition.

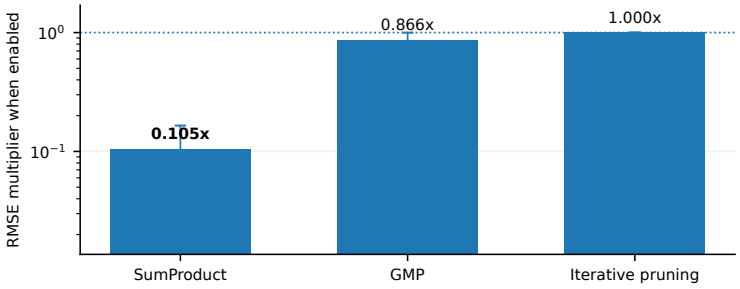

Figure 7: RuleKAN factorial ablation. For each component, the multiplier is geometric-mean RMSE when enabled divided by geometric-mean RMSE when disabled, averaged over the other two toggles. Values below one favor enabling the component; bars show interquartile ranges.

Table 6: RuleKAN factorial component ablation. Each RMSE multiplier compares enabled with disabled after balancing over the other two toggles; values below one favor enabling the component.

Component enabled	Median	First quartile	Third quartile
Sum-product grammar	**0.0907**	9.82×10^{-4}	**0.273**
GMP	0.995	0.736	1.010
Iterative pruning	1.000	1.000	1.000

Direct paired knockouts tell the same story from the opposite direction. Removing the sum-product grammar raises median symbolic RMSE by 15.97 times, while removing GMP leaves median RMSE at 1.00 times full RuleKAN. Removing iterative pruning, symbolic backfitting, or same-variable symbolic products also has a median RMSE ratio of 1.00 on 42 matched runs; the Gaussian-RBF Stage 1 model has the same median ratio over 90 matched runs. The ablation therefore identifies the product grammar as the primary source of the observed aggregate accuracy gain, while the other mechanisms have smaller median effects and can affect particular tasks.

Composition and compound targets. Figure 8 compares each composition method with its own flat counterpart. Points below the diagonal show lower error after composition. The strongest improvements lie on nested and canonical targets, whereas some tasks worsen on held-out data. The validation rule keeps the current flat model unless the composed alternative improves validation error; this does not guarantee lower test error.

Task-level errors. Figure 9 shows the individual failure-aware task medians underlying the suite-level summaries.

Coverage. The main comparison is defined over 23 methods, 30 tasks, and three seeds. Table 7 reports finite symbolic coverage and explicit timeout counts for each method; missing symbolic outputs remain in ranking and hypothesis tests through the failure-aware rule in Appendix A. There are 28 explicit timeouts: 17 for SR-KAN, 5 for Deep MultKAN+GSR, 4 for GMP, and 2 for GSR.

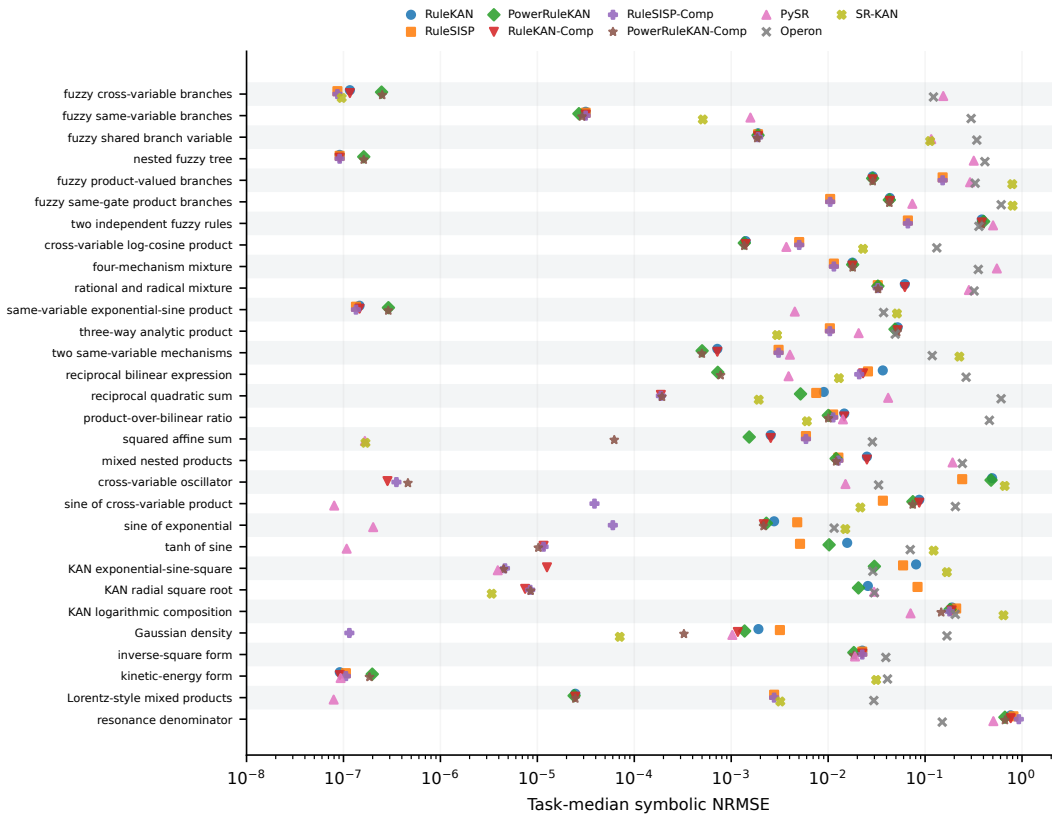

Figure 9: Task-level failure-aware symbolic NRMSE; Figure 3 reports the median within each suite, not individual task values. Leftmost methods have lower error; shapes distinguish methods within each family.

Table 7: All symbolic methods on the analytic benchmark. Bold marks the lowest median error.

Method	Symbolic runs	Timeouts	Median NRMSE
Powered rules			
PowerRuleKAN-Comp	90/90	0	**0.00107**
PowerRuleKAN	90/90	0	0.0101
Supported rules			
RuleKAN-Comp	90/90	0	0.00203
RuleKAN + OMP-Full	90/90	0	0.0160
RuleKAN Adaptive	90/90	0	0.0150
RuleKAN	89/90	0	0.0168
RuleKAN-RBF	90/90	0	0.0168
Support-independent rules			
RuleSISP-Comp	90/90	0	0.00293
RuleSISP	90/90	0	0.0105
General symbolic regression			
PySR	90/90	0	0.0170
ParFam	90/90	0	0.0338
RILS-ROLS	90/90	0	0.0703
SINDy (STLSQ)	90/90	0	0.325
Operon	90/90	0	0.159
EQL (reprod.)	90/90	0	0.124
KAN-based baselines			
SR-KAN	69/90	17	0.0820
Symbolic-KAN	67/90	0	0.980
FastKAN + GSR	90/90	0	0.202
GSR	88/90	2	0.200
GMP	86/90	4	0.231
Deep MultKAN + GSR	85/90	5	0.258
KAN	90/90	0	0.202
FastKAN	90/90	0	0.219

Complete symbolic comparison. Table 7 reports all 23 symbolic methods. "Symbolic runs" is the number of finite analytic symbolic predictions out of 90 attempted task-seed runs. "Timeouts" counts explicit hits of the 2400 s limit. NRMSE is the failure-aware median of task-level seed medians.

SINDy capacity sensitivity. SINDy limits the order-3 fixed library to 12 active non-bias terms. Without this cap, SINDy can reduce prediction error by retaining a long additive expansion: median task NRMSE falls from 0.325 to 0.00167, while the median expression grows from 12 to 672 active terms and from about 690 to 37 803 serialized characters (Table 8). Active-term count measures library usage directly, while character count summarizes the size of the serialized expression.

Table 8: SINDy capacity sensitivity on the same 30 analytic tasks. Both variants use the same order-3 library; SINDy is used in the main comparison.

Variant	Median NRMSE	Median terms	Median characters	Max terms
SINDy	0.325	12	690	12
Unconstrained SINDy	0.00167	672	37 803	13 748

The length increase is visible on individual targets. On the same-variable fuzzy rule, unconstrained SINDy reaches median NRMSE 7.02×10^{-7} using 164 active terms and about 8 886 characters; one seed begins $0.01596 + 0.00495x + 0.00216x^2 + 0.01221\sqrt{x} - 0.00703 \log |x| + \cdots$, a long surrogate rather than the annotated complementary-gate rule. On the two-independent-rules task it uses a median 3 113 terms and 177 547 characters for NRMSE 8.69×10^{-3}. The largest observed expression, on the four-mechanism mixture task, contains 13 748 active terms and 804 615 characters with NRMSE 1.08. These examples show how the unconstrained library can reduce error by expanding formula length, which motivates the 12-term cap used by SINDy in the main comparison.

Task-level statistical comparisons. Each comparison uses the 30 failure-aware task medians and the log-ratio Wilcoxon statistic defined in Section 5. PowerRuleKAN-Comp has lower error on $20/30$ tasks against PySR ($q = 0.0879$), the only non-significant external comparison. Against the other 13 baselines all comparisons are significant after Benjamini–Hochberg correction; the weakest is ParFam ($20/30$ tasks, $q = 0.0212$). SINDy loses all 30 task medians to PowerRuleKAN-Comp ($q = 6.52 \times 10^{-9}$). Proposed variants are not included in this 14-baseline multiplicity family.

I COMPOSITION SEARCH AND STRUCTURAL DEPTH

The flat Stage 2 model, $b + \sum_r a_r \prod_s g_{rs}(x_{j_{rs}})$, can add product terms but cannot place a new nonlinear operator around an already constructed multivariate expression. Composition adds this depth through bases of the form $a\, g(\beta B(x) + \gamma) + d$, where B is a unary factor, a two-variable product, or a sum of two unary factors. Continuous affine parameters and coefficients are optimized after the operator families are selected.

Support and search. RuleKAN-Comp restricts composed expressions to supports available in its effective support bank, including validation-gated support augmentation, whereas RuleSISP-Comp searches independently of Stage 1 support selection. The existing flat Stage 2 model remains a candidate, so composition is accepted only when it improves validation performance.

Composition families. The search considers unary compositions, composed products, and compositions of sums of unary factors. Candidates are first screened with fitted output coefficients, after which a bounded subset is refined by continuous optimization. A separate unary residual term may also be added when needed.

Powered compositions. PowerRuleKAN-Comp allows composed bases within the ordinary Stage 2 search, power-transformed targets, and numerator or denominator bases used by ratio search. Power and composition therefore extend different aspects of the symbolic language and may appear together in one expression, while RuleKAN variants continue to inherit their candidate supports from the effective support bank.

Validation. Composition is selected using validation performance and does not imply improved test error. The search is restricted to the depth-two families above rather than arbitrary expression trees; Figure 8 reports the resulting predictive performance.