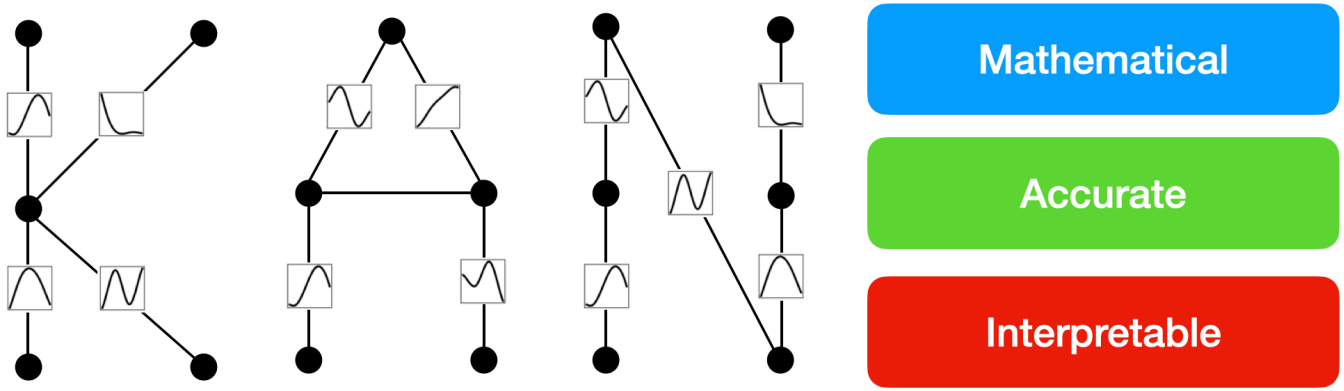
Kolmogorov-Arnold Networks

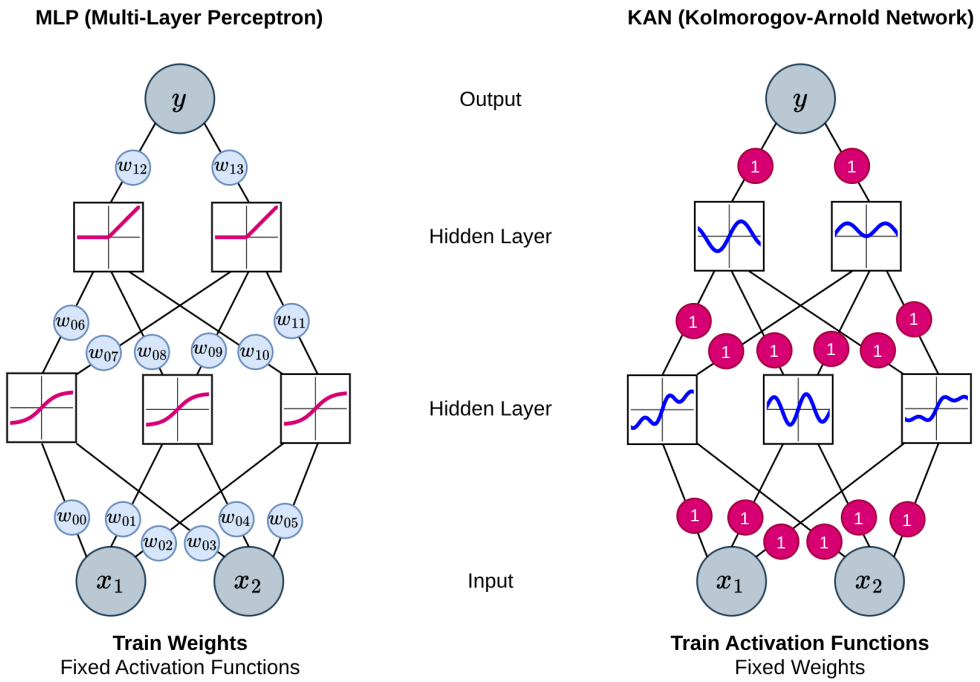


Jan Duchscherer, Felix Schladt

Advanced Deep Learning Deep Dive

› Munich University of Applied Sciences

Introduction to Kolmogorov-Arnold Networks



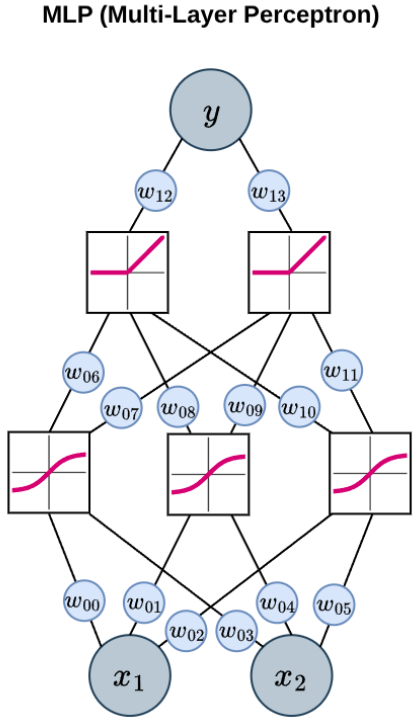
MLP vs KAN overview. [Liu+25]

MLPs: strengths and structural limits

MLPs are universal approximators — but not structure-aware.

Structural limitations

- **Interpretability**: Knowledge is contained in distributed parametric pattern; individual components are not meaningful by design.
- **Entangled degrees of freedom**: the same parameters encode both structure and precision.



A small MLP (structure and precision are intertwined)

MLPs: strengths and structural limits

1. Can we separate what is computed from how precisely it is computed?
2. Can we design a network that is equally expressive but more interpretable?

Features of KANs

Where KANs work well

- Built for smooth + compositional relationships
- Matches structure common in physics / biology
- Allocate capacity where needed (local splines)

Efficient in science tasks

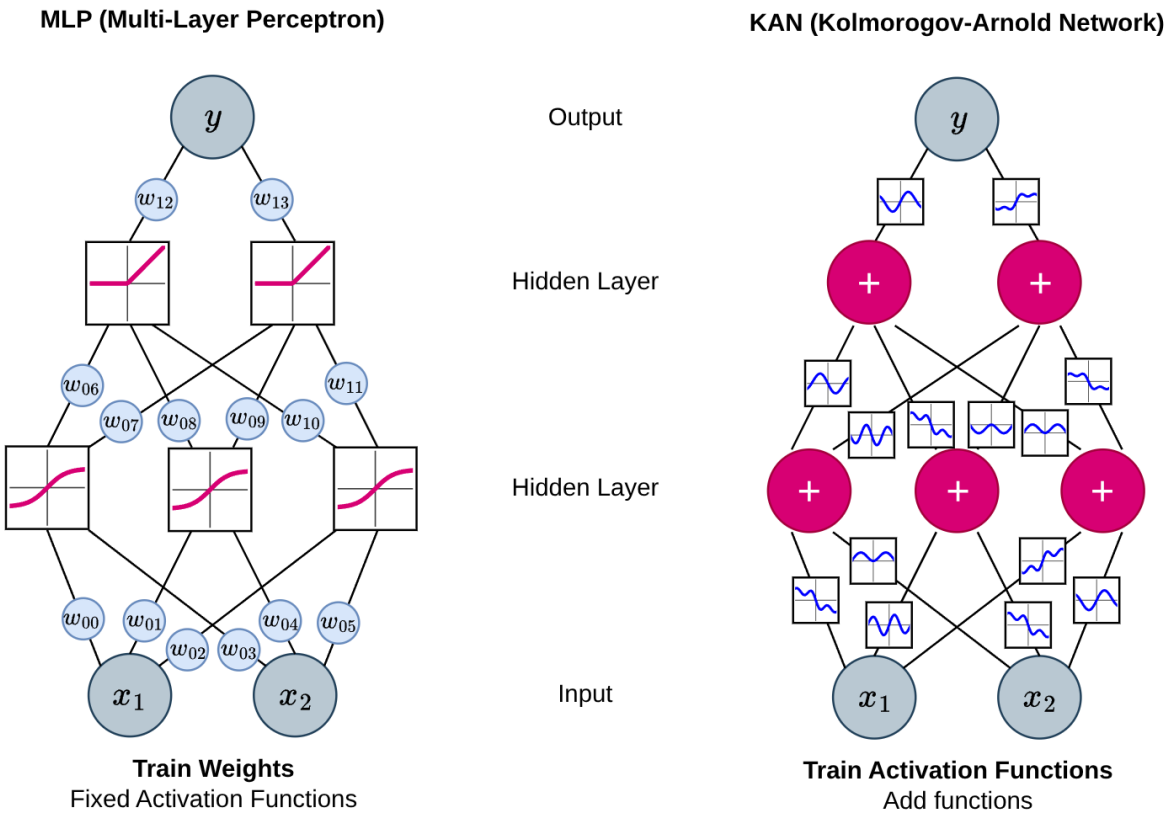
- Often similar error with fewer parameters than MLP baselines
- Function fitting and differential-equation solving (common in science)
- Precision scales independently from structure

Interpretability

- Individual components / connections are meaningful.
- Can be inspected, pruned, simplified into compact relations
- Extract a symbolic equation from the trained model

[Liu+25]

MLP vs KAN: Visual comparison



MLP vs. KAN visualization (learnable parts in blue, fixed in pink) [Ser24]

From MLPs to KANs

MLP

- Scalar weights $w_{j,i}$ on edges.
- Fixed activation functions σ on nodes.

$$\boldsymbol{x}_{l+1} = \sigma(\boldsymbol{W}_l \boldsymbol{x}_l + \boldsymbol{b}_l)$$

Learnable weights + fixed activations

KAN

- Each edge is a **learnable 1D function** $\varphi_{j,i}(x)$
- Nodes add input $\longrightarrow$ interpret learned functions.

$$x_{l+1,j} = \sum_{i=1}^{n_l} \varphi_{l,j,i}(x_{l,i})$$

Nodes add, function on edges are learned

Same dense bipartite wiring.
Activations become learnable & move from nodes to edges.

[Liu+25]

The Kolmogorov-Arnold theorem

- Any continuous $f(x_1, ..., x_n)$ on $[0, 1]^n$ can be represented using **1D functions + addition**.
- The only true multivariate operation is **sum**; everything else can be composed from univariate transforms + additions.
- Splines are well-suited to approximate 1D functions efficiently.

[Liu+25]

$$f(\boldsymbol{x}) = f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \varphi_{q,p}(x_p) \right)$$

Kolmogorov-Arnold Theorem[Liu+25]

$$xy = -\frac{x^2}{2} - \frac{y^2}{2} + \frac{(x+y)^2}{2}$$

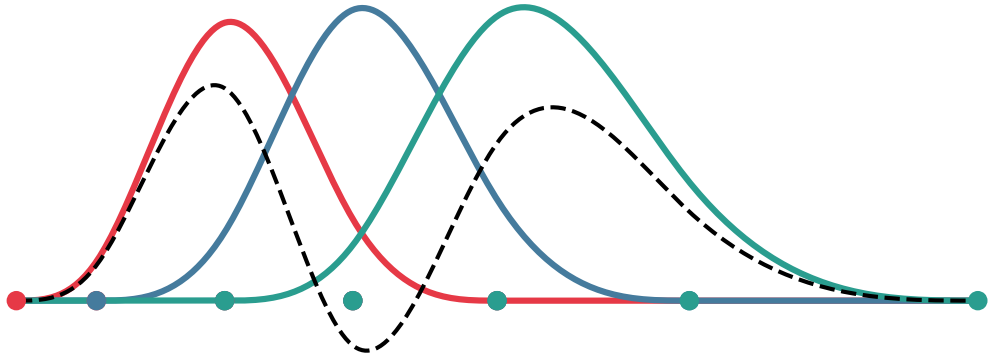
KAT representation example for xy
($\frac{(x+y)^2}{2}$ requires 1D summary of $t = x + y$)

Splines

- **Smooth**, **piecewise-polynomial** & **locally bounded** functions of one variable x.
- Approximate 1D functions well with few parameters.
- Controlled by knots $B_m(x)$ + coefficients c_m.
- Each Spline is a combination of many cubic Basis Splines (B-Splines)

$$\varphi_{j,i}(x) = \sum_{m=0}^{G+k-1} c_m B_m(x)$$

[Liu+25, Ser24]



Edge function (dashed): linear combination of $G = 3$ B-Splines of degree $k = 3$

> In KANs, each edge learns its own 1D spline $\varphi_{i,j}(x)$. [Liu+25]

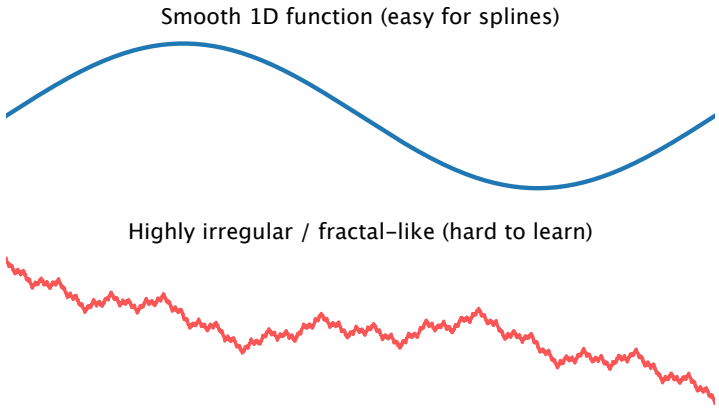
Can KAT represent any high-dimensional function?

Classical KAT is elegant, but resulting 1D inner functions can be non-smooth or fractal

- Hard to learn in the rigid depth-2 KART form in practice
- Earlier research described it as "theoretically sound but practically useless"[GP89, PBL19]

Mitigation

- Don't stick to the rigid depth-2, width $(2n + 1)$ form
  - deeper/wider KANs for smoother representations
- In real tasks we often expect smooth + compositionally sparse structure
  - most typical cases allows smooth KA-like representations



Smooth vs Fractal Curves visualization

Curse of Dimensionality

> More input dimensions → combinations explode → exponential growth of parameters

MLPs

- Universal Approximation Theorem: 2-layer MLP can approximate any continuous f
- But no efficiency guarantee
  - width can grow exponentially with dimension (CoD in practice)

KANs

- Stack layers to learn compositional structure (feature learning)
- Replace weights with learnable 1D functions
- No high-D spline grid:
  - many 1D splines + sums, can beat CoD when the target is smooth + compositional

KAN Layer

Layer = matrix of 1D functions

- Each edge learns a univariate mapping

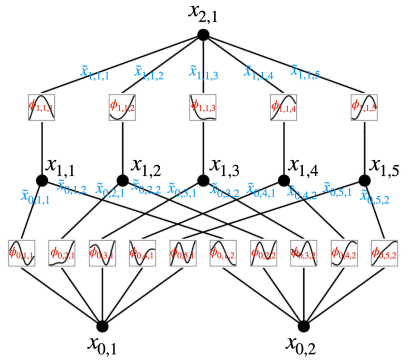
$$\varphi_{l,j,i} : \mathbb{R} \to \mathbb{R}.$$

- Activation of the j-th neuron in layer $l+1$:

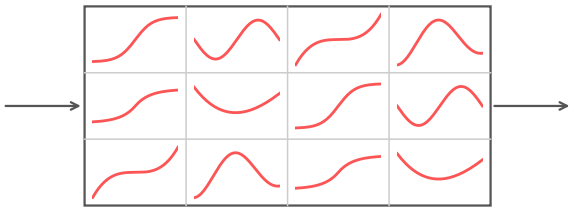
$$x_{l+1,j} = \sum_{i=1}^{n_l} \varphi_{l,j,i}(x_{l,i})$$

- A single KAN layer can be written in **matrix form**:

$$\boldsymbol{x}_{l+1} = \underbrace{\begin{pmatrix} \varphi_{l,1,1}(\cdot) & \varphi_{l,1,2}(\cdot) & \cdots & \varphi_{l,1,n_l}(\cdot) \\ \varphi_{l,2,1}(\cdot) & \varphi_{l,2,2}(\cdot) & \cdots & \varphi_{l,2,n_l}(\cdot) \\ \vdots & \vdots & & \vdots \\ \varphi_{l,n_{l+1},1}(\cdot) & \varphi_{l,n_{l+1},2}(\cdot) & \cdots & \varphi_{l,n_{l+1},n_l}(\cdot) \end{pmatrix}}_{\boldsymbol{\Phi}_l \in (\mathbb{R}\to\mathbb{R})^{n_{l+1}\times n_l}} \boldsymbol{x}_l$$



B-spline parametrization and grid refinement. [Liu+25]



KAN layer in matrix form

General KAN with L layers

$$\mathrm{KAN}(\boldsymbol{x}) = (\boldsymbol{\Phi}_{L-1} \circ \boldsymbol{\Phi}_{L-2} \circ \ldots \circ \boldsymbol{\Phi}_0)(\boldsymbol{x})$$

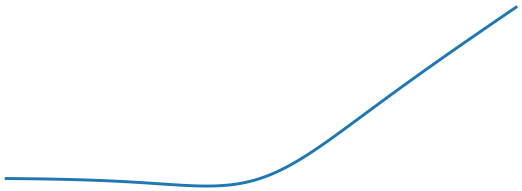
KART ⊷ KAN of shape $[n \to 2n+1 \to 1]$

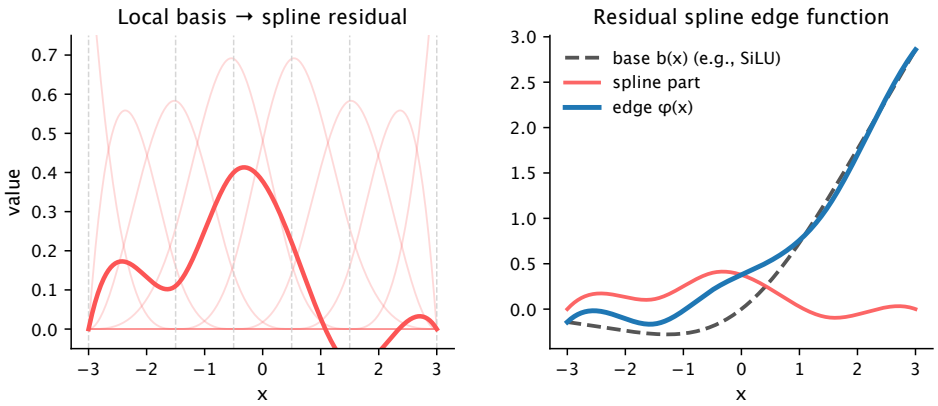
Edge Functions - Residual Splines

Edge - Residual Spline

$$\varphi(x) = w_b b(x) + w_s \sum_i c_i B_i(x)$$

- Trainable (per edge, backprop): c_i, w_b, w_s
- $B_i(x)$: B-spline basis functions, fixed given the current grid.
- $b(x)$: fixed **global** non-linearity (i.e. SiLU).
  1. Ensure φ is well-defined on $\mathbb{R}$
  2. Residual path eases optimization – learn deviation from $b(x)$ rather than full function





Residual spline edge function: local basis ⟶ spline ⟶ $\varphi(x)$.

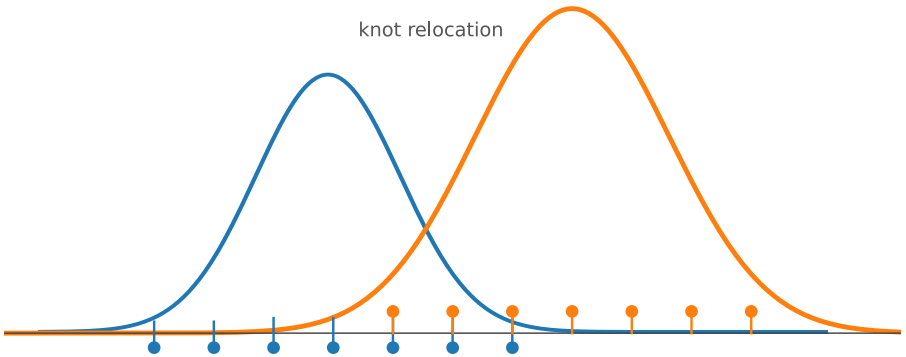
Why B-Splines?

- **local**, **translation-invariant** basis
  - local capacity allocation
  - continual learning ↑, catastrophic forgetting ↓
- Allows for other orthogonal bases (Fourier, Chebyshev).
- Locality ↔ global efficiency.

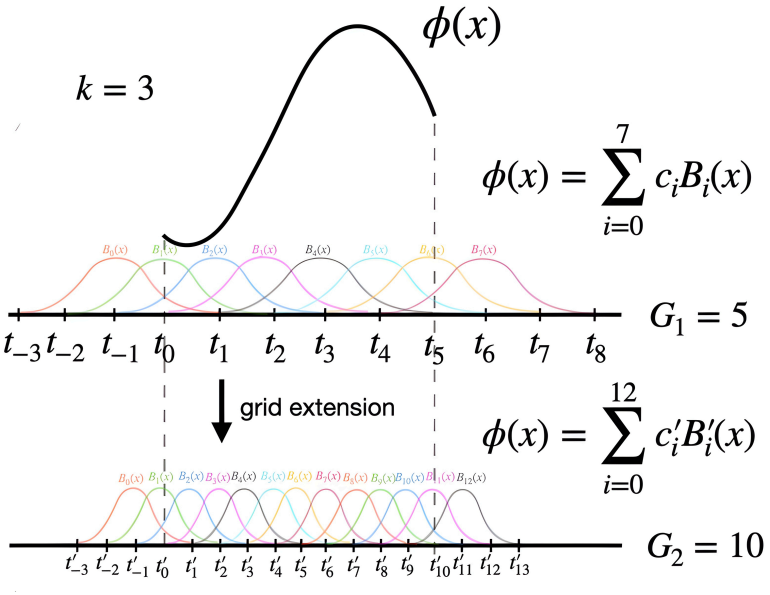
Grid Update - Knot Relocation

Keep knots where the data lives

- Non-stationary activations in training, but splines live on bounded grid
- **Grid update:** periodically estimate activation distributions; move knots to maintain coverage.
- non-differentiable reparameterization



Non-stationarity motivates knot relocation.



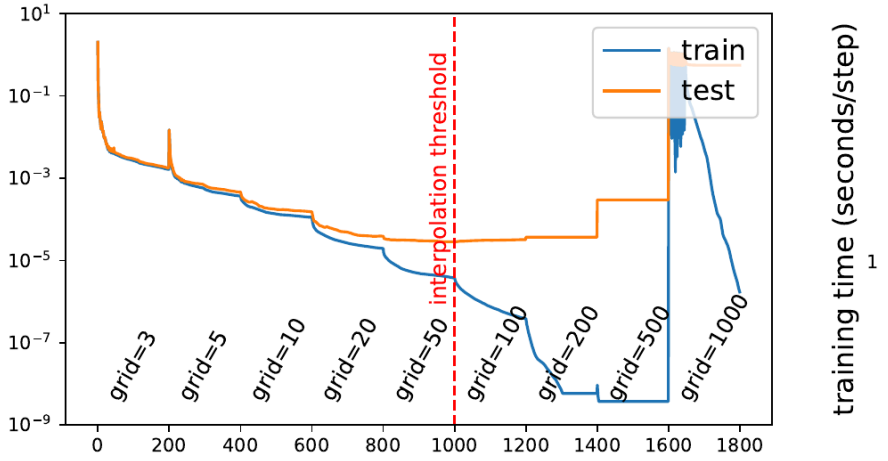
[Liu+25]

Grid updates reallocate representational capacity at **fixed number of knots** (contrast: grid extension adds knots).

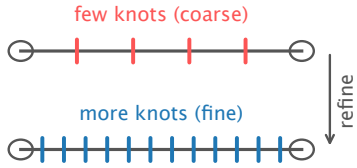
Grid Extension: Accuracy Scaling

Key idea

- **Grid extension**: add knots ($G \longrightarrow G'$) $\longrightarrow$ higher spline resolution.
- Curriculum-style schedule:
  1. Start with coarse spatial resolution – fewer knots, global structure, simpler optimization.
  2. Gradually increase resolution, initialize finer splines via least-squares fit to coarse spline.
- Monitor validation error to stop grid extension once improvement ceases.
- RMSE $\propto G^{-4}$ (on test split)



Staircase-like loss drops after each grid refinement. [Liu+25]



$$\{c'_j\} = \underset{\{c'_j\}}{\operatorname{argmin}} \mathbb{E}_{x\sim p(x)} \left(\sum_{j=0}^{G_2+k-1} c'_j B'_j(x) - \sum_{i=0}^{G_1+k-1} c_i B_i(x) \right)^2$$

External vs Internal DoF: Structure vs Precision

Two kinds of degrees of freedom (DoF)

- **External DoF**: width/depth/connectivity $\longrightarrow$ **which interactions** among variables.
- **Internal DoF**: spline grid G + coefficients c_i $\longrightarrow$ **how precisely** each interaction is represented.
- **Key shift**: KANs separate structure from precision, so each can scale independently.

MLPs

- Structure + precision are **entangled**.
- More accuracy $\longrightarrow$ scale width/depth (external DoF).
- No dedicated control over internal DoF.
- $\mathcal{O}(N^2 L)$

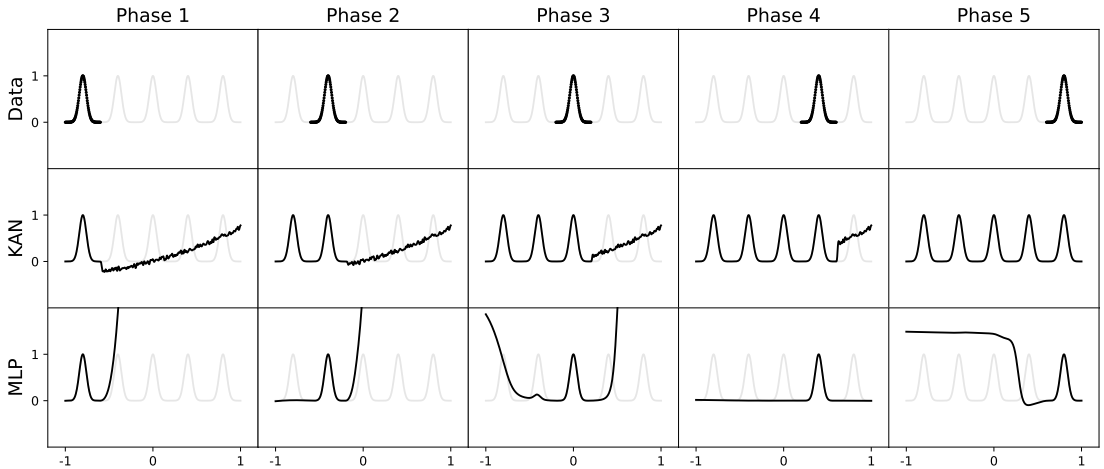
KANs

- External DoF $\leftrightarrow$ **compositional structure**.
- Internal DoF $\leftrightarrow$ **local precision** on edges.
- Grid extension increases internal DoF while keeping structure fixed.
- $\mathcal{O}(N^2 L(G+k))$

Continual Learning & Locality

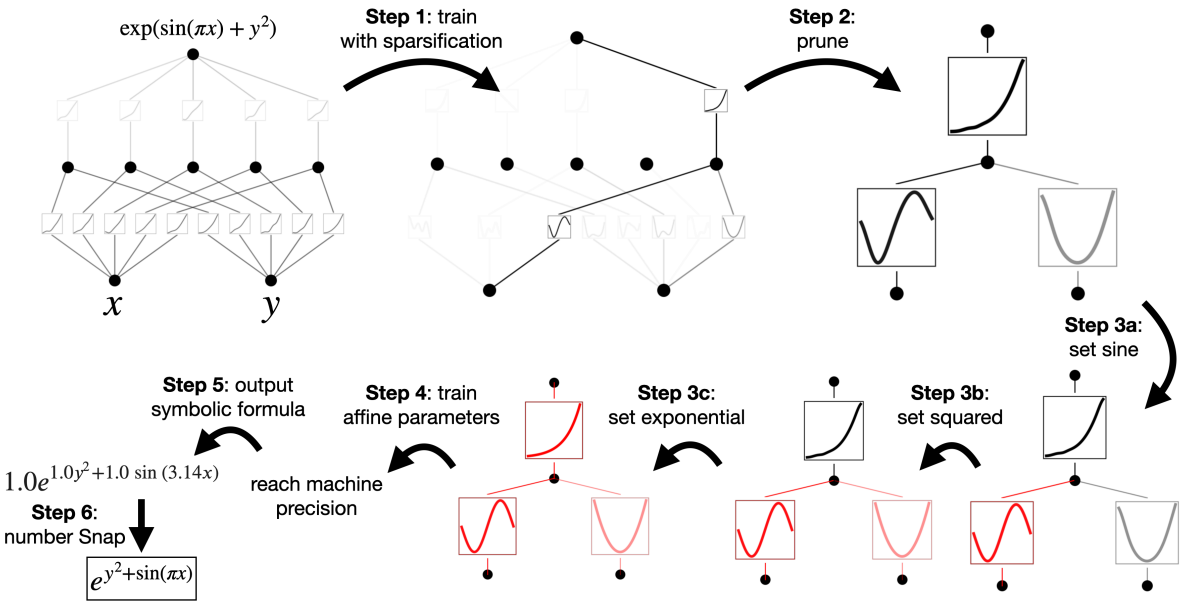
Why KANs forget less

- **Local plasticity:** Each spline basis is local:
  → only a few basis functions are non-zero per sample
  → gradients touch only a few nearby coefficients.
  → updates are localized.
- **Contrast (MLPs):** Any local update has global effects:
  → learning a new task may overwrite previous knowledge
  → catastrophic forgetting.
- Generalization to higher dimensions is unclear [YYW24].



Sequential learning (toy 1D): KAN preserves earlier regions; MLP forgets. [Liu+25]

Reductionist Symbolification



Symbolic regression workflow with KANs. [Liu+25]

KANs are connectionist throughout training & symbolic after reductionist sparsification & compression into symbolic forms.

Structure → equation

1. **Sparsify**: L1 on activations + entropy regularization
   → few dominant edges.
2. **Prune** low-importance nodes
   → minimal graph
   → external DoF ↓.
3. **Symbolify**: replace splines with analytic forms + affine wrapper
   $$y \approx c f(ax + b) + d$$
   → internal DoF ↓.
4. **Further training** of the (affine) parameters.

Summary: Takeaways & Limitations

Takeaways

- KANs are most suited for **structured, compositional, small-scale scientific tasks**.
- **Decoupled internal and external DoF** improve fine-tuning, **continual learning** & scaling.
- **Symbolic interpretability** and discovery of symbolic formulas (**white-box ML**).
- Rapid **follow-up ecosystem**: efficiency variants, hybrids, theory, and broad applications.

Limitations

- Baseline KANs are up to 10x **slower** (batching, spline updates, HW); faster follow-ups exist [Li24].
- Likely **not an MLP replacement** in large-scale tasks [YYW24].
- Sensitivity to **additional hyperparameters** (grid size, update schedule).

Summary: Takeaways & Limitations

Trade-off: interpretability & controllable precision vs training efficiency.

References

[Liu+25] Z. Liu et al., "KAN: Kolmogorov-Arnold Networks." Accessed: Dec. 21, 2025. [Online]. Available: http://arxiv.org/abs/2404.19756

[Ser24] Serrano.Academy, "Kolmogorov-Arnold Networks (KANs) - What are they and how do they work?." Accessed: Dec. 21, 2025. [Online]. Available: https://www.youtube.com/watch?v=myFtp5zMv8U

[GP89] F. Girosi and T. Poggio, "Representation Properties of Networks: Kolmogorov's Theorem Is Irrelevant," Neural Computation, vol. 1, no. 4, pp. 465–469, Dec. 1989, doi: 10.1162/neco.1989.1.4.465.

[PBL19] T. Poggio, A. Banburski, and Q. Liao, "Theoretical Issues in Deep Networks: Approximation, Optimization and Generalization." Accessed: Jan. 10, 2026. [Online]. Available: http://arxiv.org/abs/1908.09375

[YYW24] R. Yu, W. Yu, and X. Wang, "KAN or MLP: A Fairer Comparison." Accessed: Jan. 12, 2026. [Online]. Available: http://arxiv.org/abs/2407.16674

[Li24] Z. Li, "Kolmogorov-Arnold Networks are Radial Basis Function Networks." Accessed: Jan. 12, 2026. [Online]. Available: http://arxiv.org/abs/2405.06721