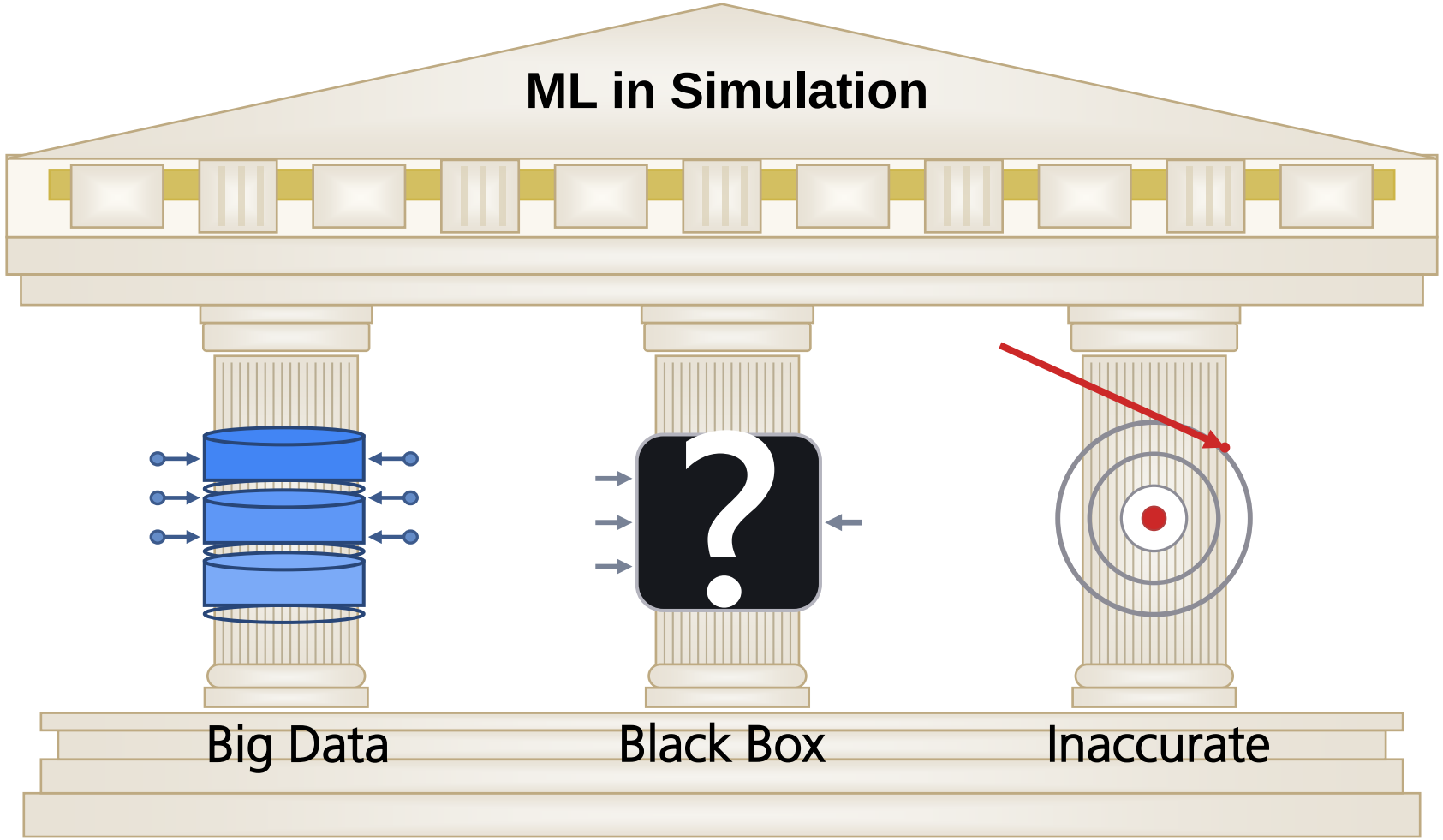
Scientific Machine Learning (SciML) – How the fusion of AI and physics is giving rise to promising simulation methodologies

SISPAD 2025, Grenoble
Andreas Rosskopf, and the Modeling- and AI-Department

Fraunhofer IISB

Fraunhofer-Institut für Integrierte Systeme und Bauelemente-technologie IISB

Three myths about ML in simulation …



Three techniques to combine …

… to fulfill future engineering needs

Measurement & Experiment

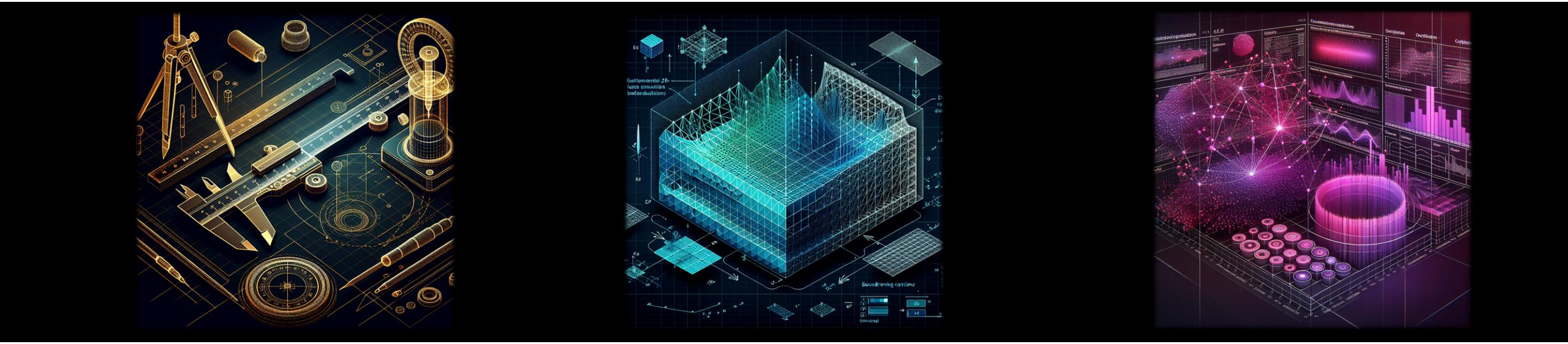
- Provides highest-fidelity ground-truth data from reality
- Uncovers unknown effects and model gaps; foundation for calibration
- Essential for verification/validation; defines boundary conditions and material parameters.

(Numerical) Simulation

- Physics-based solvers (e.g., FEM, FDTD); established and validated.
- Robust with limited data: supports parameter sweeps, inter-/extrapolation, and DoE with few measurements.
- Reproducible with built-in uncertainty quantification (e.g., sensitivities, MC).

ML (AI and Big Data)

- Automatic feature extraction from raw data even for complex, nonlinear physics
- Ultra-fast surrogate models for design-space exploration, optimization, and real-time control
- Highly parallelizable, adaptive via transfer and online learning; IP-protected



Three techniques to combine …

… to fulfill future engineering needs

Measurement & Experiment

- Provides highest-fidelity ground-truth data from reality
- Uncovers unknown effects and model gaps; foundation for calibration
- Essential for verification/validation; defines boundary conditions and material parameters.

(Numerical) Simulation

- Physics-based solvers (e.g., FEM, FDTD); established and validated.
- Robust with limited data: supports parameter sweeps, inter-/extrapolation, and DoE with few measurements.
- Reproducible with built-in uncertainty quantification (e.g., sensitivities, MC).

ML (AI and Big Data)

- Automatic feature extraction from raw data even for complex, nonlinear physics
- Ultra-fast surrogate models for design-space exploration, optimization, and real-time control
- Highly parallelizable, adaptive via transfer and online learning; IP-protected

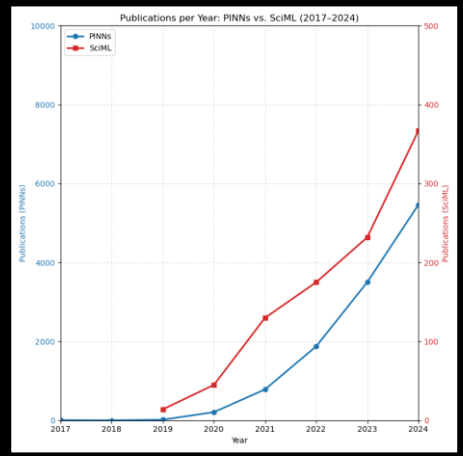
Scientific Machine Learning (SciML)

PINN: Physics-Informed Neural Network
PIML: Physics-Informed Machine Learning
FNO: Fourier Neural Operator
DeepONet: Deep Operator Network
Neural ODE: Neural Ordinary Differential Equations model
…

Scientific relevance

TITLE	CITED BY	YEAR
Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations M Raissi, P Perdikaris, GE Karniadakis Journal of Computational physics 378, 686-707	16550	2019

arXiv
https://arxiv.org › cs · Diese Seite übersetzen
[2404.19756] KAN: Kolmogorov-Arnold Networks
von Z Liu · 2024 · Zitiert von: 2025 — View a PDF of the paper titled **KAN: Kolmogorov-Arnold Networks**, by **Ziming Liu** and 6 other authors. View PDF HTML (experimental). Abstract …

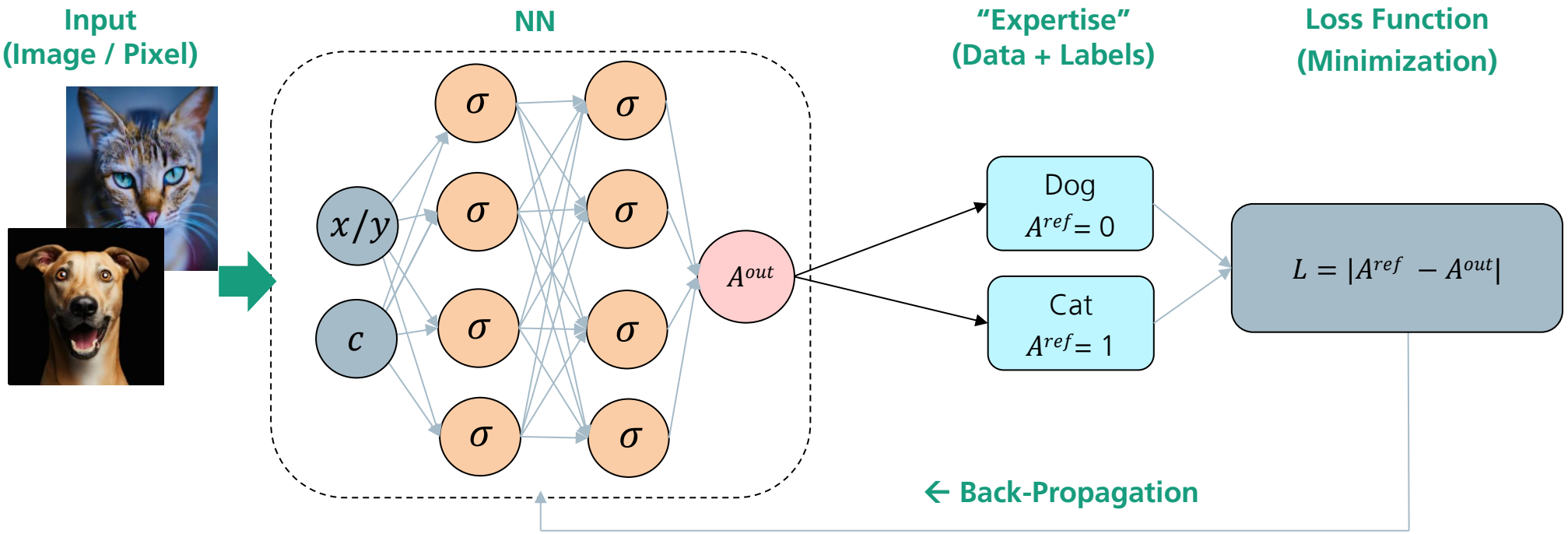


Fraunhofer IISB

SciML Basis

Physics-Informed Neural Network – Supervised Learning

Problem: Image recognition "dog or cat"
Solution: Supervised Learning



Fraunhofer IISB

SciML Basis

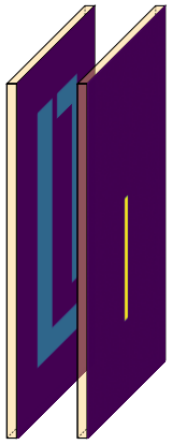
Physics-Informed Neural Network – Supervised Learning

Problem: Calculation of electromagnetic fields
Solution: Physics-Informed Learning

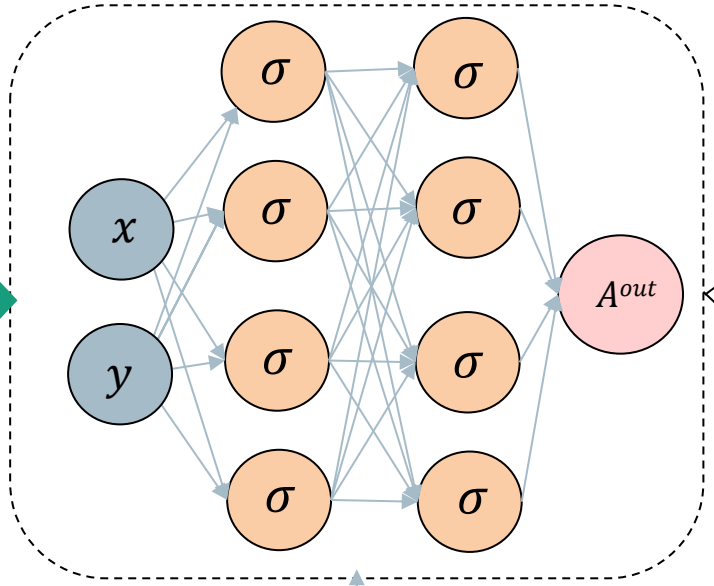
$$\frac{1}{\mu}\left(\frac{\partial^2 A(x,y)}{\partial x^2} - \frac{\partial^2 A(x,y)}{\partial y^2}\right) = J(x,y)$$

$$A(x,y) = 1 \qquad (x,y) \in \partial\Omega$$

Input (Geometry, Time, etc.)



NN



"Expertise" (Physics)

$$L_{pde} = \frac{1}{\mu}\left(\frac{\partial^2 A(x,y)}{\partial x^2} - \frac{\partial^2 A(x,y)}{\partial y^2}\right) - J(x,y)$$

$$L_{bc} = \begin{cases} A(x,y) - 1, & (x,y) \in \partial\Omega \\ 0, & (x,y) \notin \partial\Omega \end{cases}$$

Loss Function (Minimization)

$$L = |L_{PDE}| + |L_{bc}| + \\ + \ \ldots$$

← Back-Propagation

1D Ni-SiC Silicidation Model

Formulation

SEMICONDUCTOR STRUCTURES, INTERFACES, AND SURFACES

Simulation of Interaction Between Nickel and Silicon Carbide during the Formation of Ohmic Contacts

O. V. Aleksandrov[a]^ and V. V. Kozlovski[b]^^
[a]St. Petersburg State Electrotechnical University "LÉTI," St. Petersburg, 197376 Russia
^e-mail: Aleksandr_ov@mail.ru
[b]St. Petersburg State Polytechnical University, St. Petersburg, 195251 Russia
^^e-mail: Kozlovski@tuexph.stu.neva.ru
Submitted November 6, 2008; accepted for publication November 17, 2008

Reaction-diffusion model for concentrations C_{Ni}, C_{SiC}, C_{C}, C_{NiSi}, $C_{\mathrm{NiSi_2}}$ depending on time $t \in [0, T]$, space $x \in [0, L]$.

$$\partial_t C_{\mathrm{Ni}} = \partial_x (D^* \partial_x C_{\mathrm{Ni}}) - k_1 C_{\mathrm{Ni}} C_{\mathrm{SiC}},$$
$$\partial_t C_{\mathrm{SiC}} = \partial_x (D^* \partial_x C_{\mathrm{SiC}}) - k_1 C_{\mathrm{Ni}} C_{\mathrm{SiC}} - k_2 C_{\mathrm{NiSi}} C_{\mathrm{SiC}},$$
$$\partial_t C_{\mathrm{C}} = \partial_x (D^* \partial_x C_{\mathrm{C}}) + k_1 C_{\mathrm{Ni}} C_{\mathrm{SiC}} + k_2 C_{\mathrm{NiSi}} C_{\mathrm{SiC}},$$
$$\partial_t C_{\mathrm{NiSi}} = k_1 C_{\mathrm{Ni}} C_{\mathrm{SiC}} + k_2 C_{\mathrm{NiSi}} C_{\mathrm{SiC}},$$
$$\partial_t C_{\mathrm{NiSi_2}} = k_2 C_{\mathrm{NiSi}} C_{\mathrm{SiC}},$$
$$D^* = D_{\mathrm{Ni}} \frac{C_{\mathrm{SiC}} + C_{\mathrm{C}} + C_{\mathrm{NiSi}} + C_{\mathrm{NiSi_2}}}{C_{\mathrm{Ni}} + C_{\mathrm{SiC}} + C_{\mathrm{C}} + C_{\mathrm{NiSi}} + C_{\mathrm{NiSi_2}}},$$

$\partial_x C(x,t) = 0$ for $x = 0$ and $x = L$ for all components,

$C_{\mathrm{Ni}}(x, t=0) = C_{0,\mathrm{Ni}}$ for $0 \le x \le h$, $C_{\mathrm{Ni}}(x, t=0) = 0$ for $h \le x \le L$,

$C_{\mathrm{SiC}}(x, t=0) = 0$ for $0 \le x \le h$, $C_{\mathrm{SiC}}(x, t=0) = C_{0,\mathrm{Si}}$ for $h \le x \le L$,

$C_{\mathrm{C}}(x, t=0) = C_{\mathrm{NiSi}}(x, t=0) = C_{\mathrm{NiSi_2}}(x, t=0) = 0.$

Reaction constants:
$k_1 = k_2 = 6 \cdot 10^{-5} \frac{\mathrm{nm}^3}{\mathrm{s}}$.

Metal diffusivity:
$D_{\mathrm{Ni}} = 6 \cdot 10^{-14} \frac{\mathrm{cm}^2}{\mathrm{s}}$.

Heterodiffusion coefficient D^*.

Reflecting boundaries.

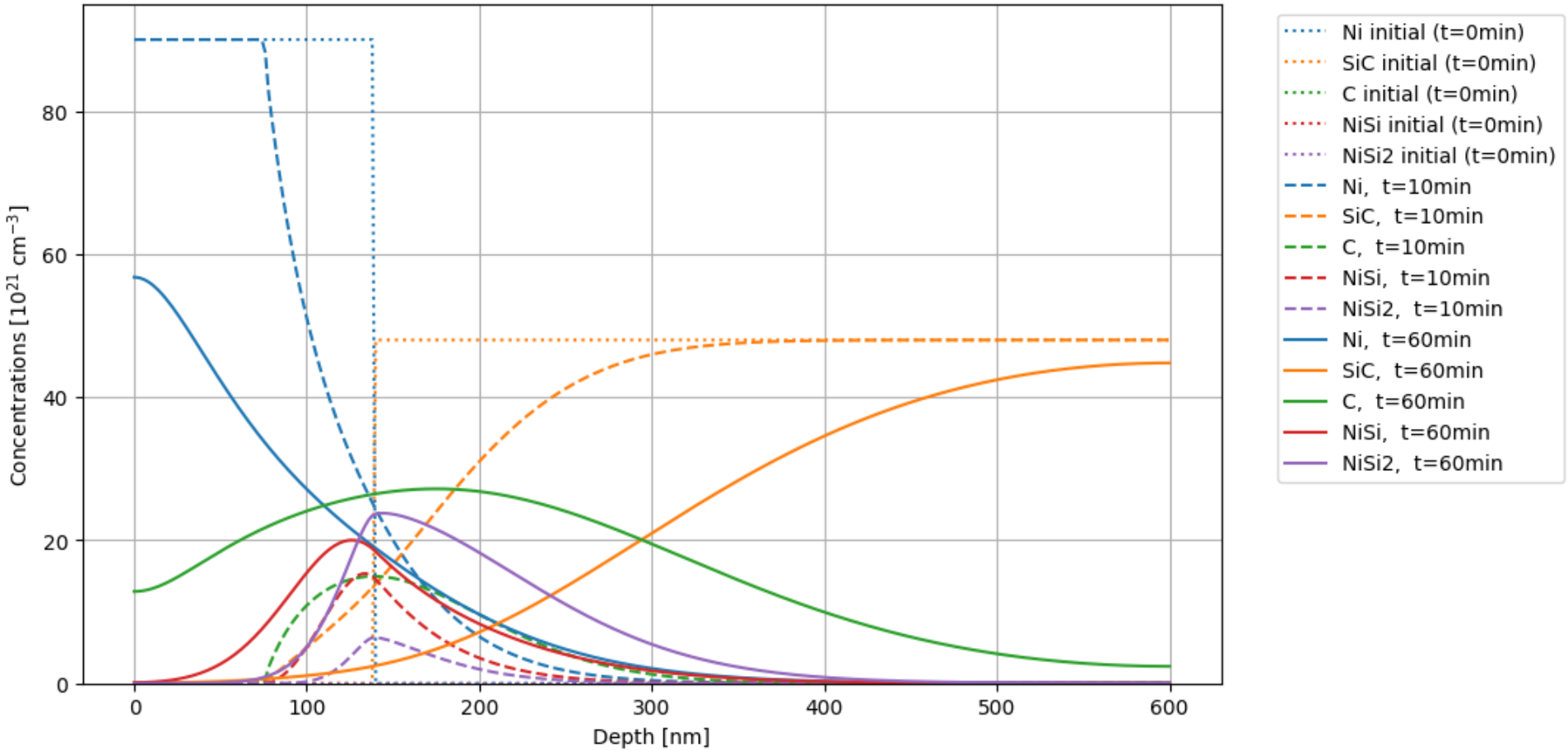
Initial metal thickness: $h = 140$ nm,
and intrinsic concentrations: $C_{0,\mathrm{Ni}} = 9 \cdot 10^{22} \mathrm{cm}^{-3}$,
$C_{0,\mathrm{Si}} = 4.8 \cdot 10^{22} \mathrm{cm}^{-3}$.

Total time: T = 1 h.

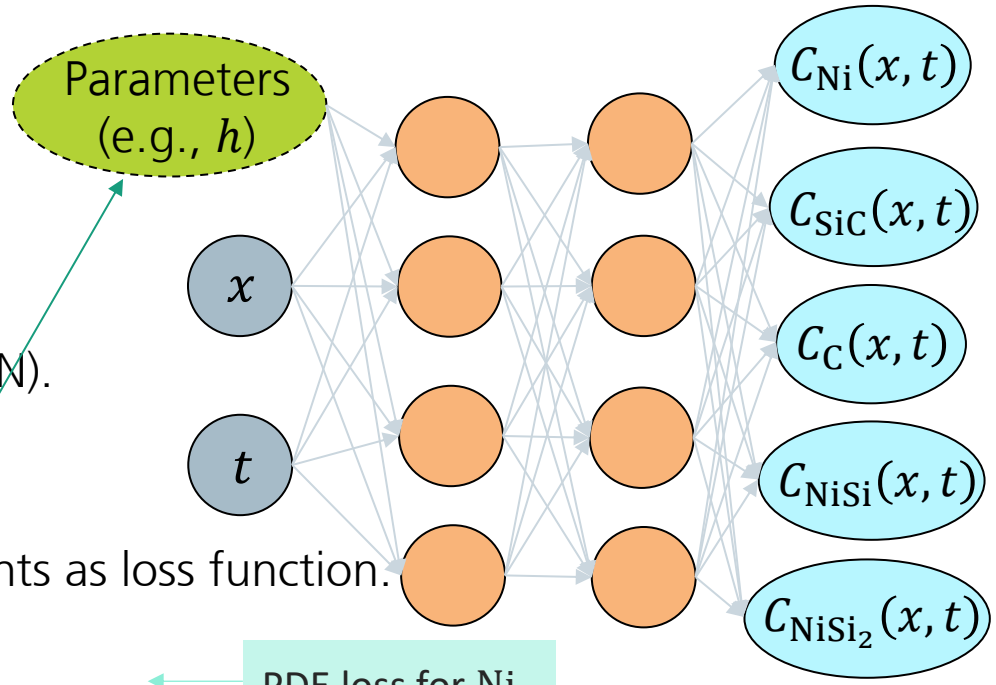
Total depth: $L = 600$ nm.

1D Ni-SiC Silicidation Model

Solution behaviour



Physics-informed neural networks (PINNs)



Solve this system by training a „physics-informed neural network“ (PINN).

- Represent solution as a neural network.
  Train it by using the problem's residuals evaluated at collocation points as loss function.

$$\mathcal{L} := \frac{1}{N_{\mathrm{PDE}}} \sum_{j=1}^{N_{\mathrm{PDE}}} \left(-\partial_t C_{\mathrm{Ni}} + \partial_x \left(D^* \partial_x C_{\mathrm{Ni}} \right)^2 - k_1 C_{\mathrm{Ni}} C_{\mathrm{SiC}} \right)^2 \Big|_{(x_j^{\mathrm{PDE}}, t_j^{\mathrm{PDE}})}$$

PDE loss for Ni.

$$+ \frac{1}{N_{\mathrm{ic}}} \sum_{j=1}^{N_{\mathrm{ic}}} \left(C_{\mathrm{Ni}}(x_j^{\mathrm{ic}}, 0) - C_{0,\mathrm{Ni}} \cdot \mathbb{1}_{[0,h]}(x_j^{\mathrm{ic}}) \right)^2$$

Initial condition for Ni.

$$+ \frac{1}{N_{\mathrm{bc}}} \sum_{j=1}^{N_{\mathrm{bc}}} \left[\left(\partial_x C_{\mathrm{Ni}}(0, t_j^{\mathrm{bc}}) \right)^2 + \left(\partial_x C_{\mathrm{Ni}}(L, t_j^{\mathrm{bc}}) \right)^2 \right] + \dots$$

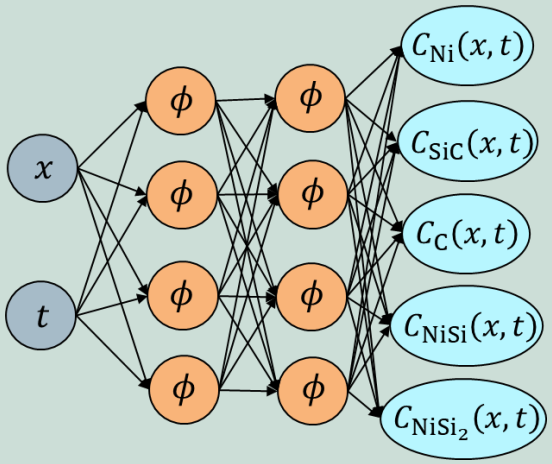
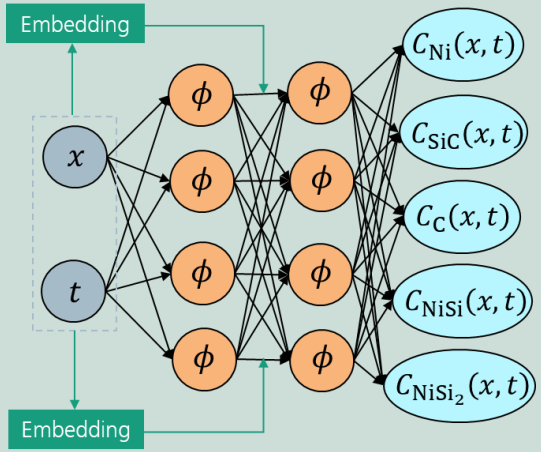
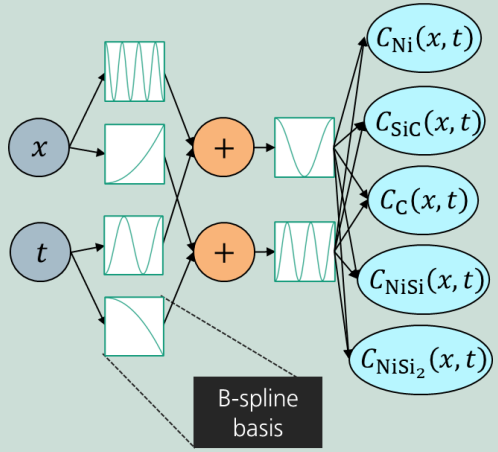
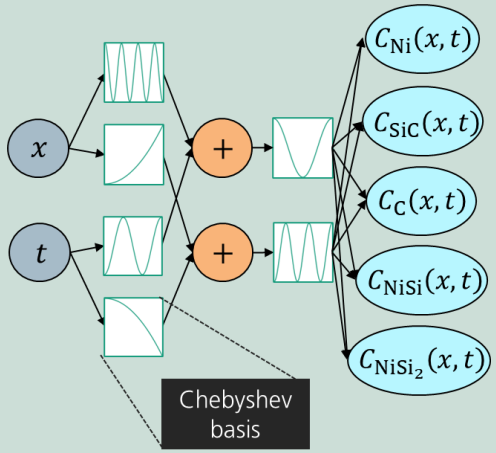
Boundary condition for Ni.

- Extendable to a „physics-informed neural operator“ (PINO).
  → Make parameters (initial condition, model parameters like temperature, …) part of neural network's input.
  → Arrive at fast to evaluate surrogate model that can, e.g., be used for optimization tasks.

PINN Architectures

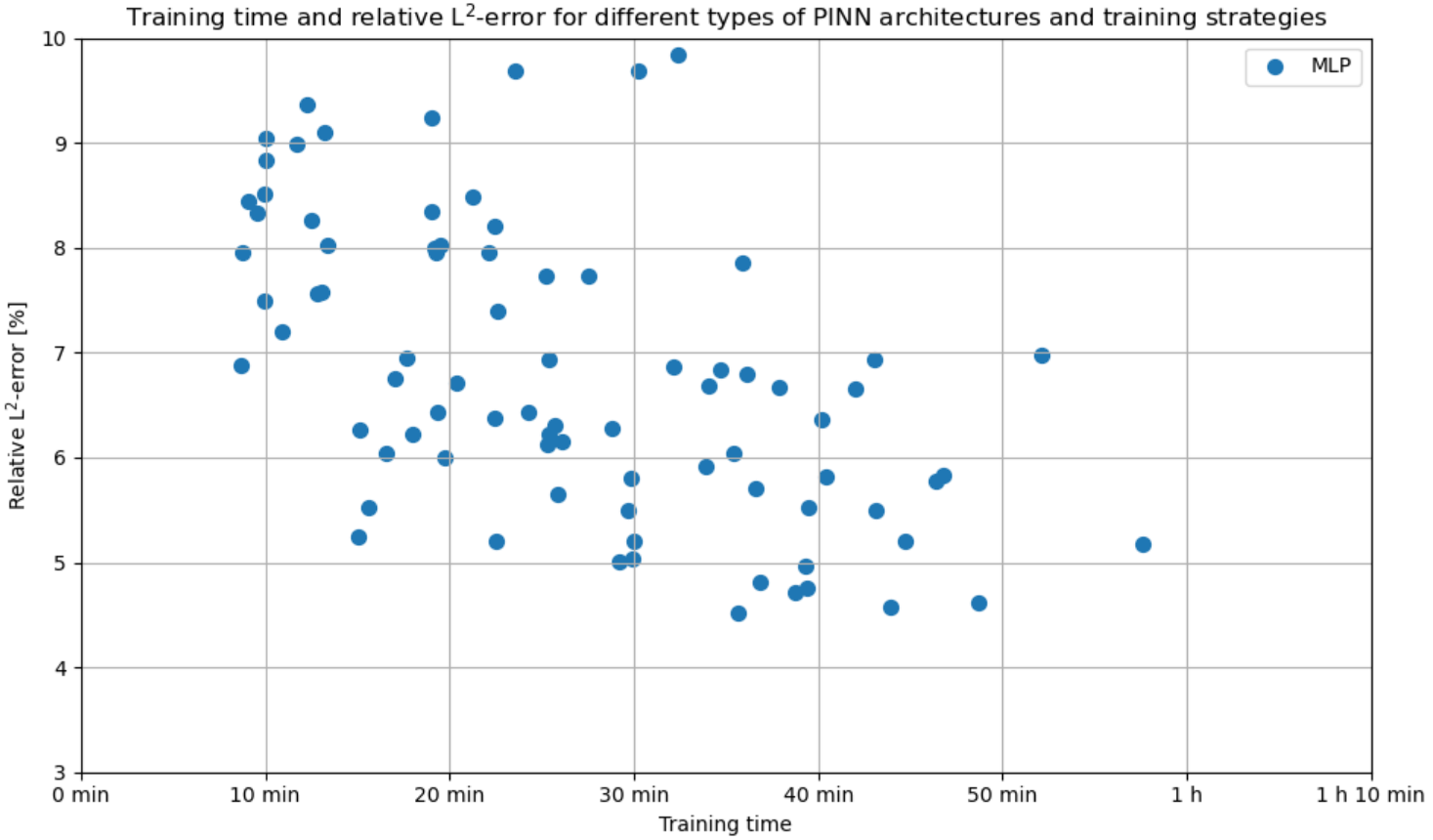
Overview

Four common architectures for the NN part inside the PINN:

MLP Multi-Layer Perceptron	mMLP Modified Multi-Layer Perceptron	KAN Kolmogorov-Arnold Network	cKAN Chebyshev-based Kolmogorov-Arnold network
			
Source: arXiv:2407.19421v1	Source: arXiv:2407.19421v1	Source: arXiv:2406.02917v1	Sources: arXiv:2406.02917v1, arXiv:2405.07200v3

PINN Results for Silicidation Problem

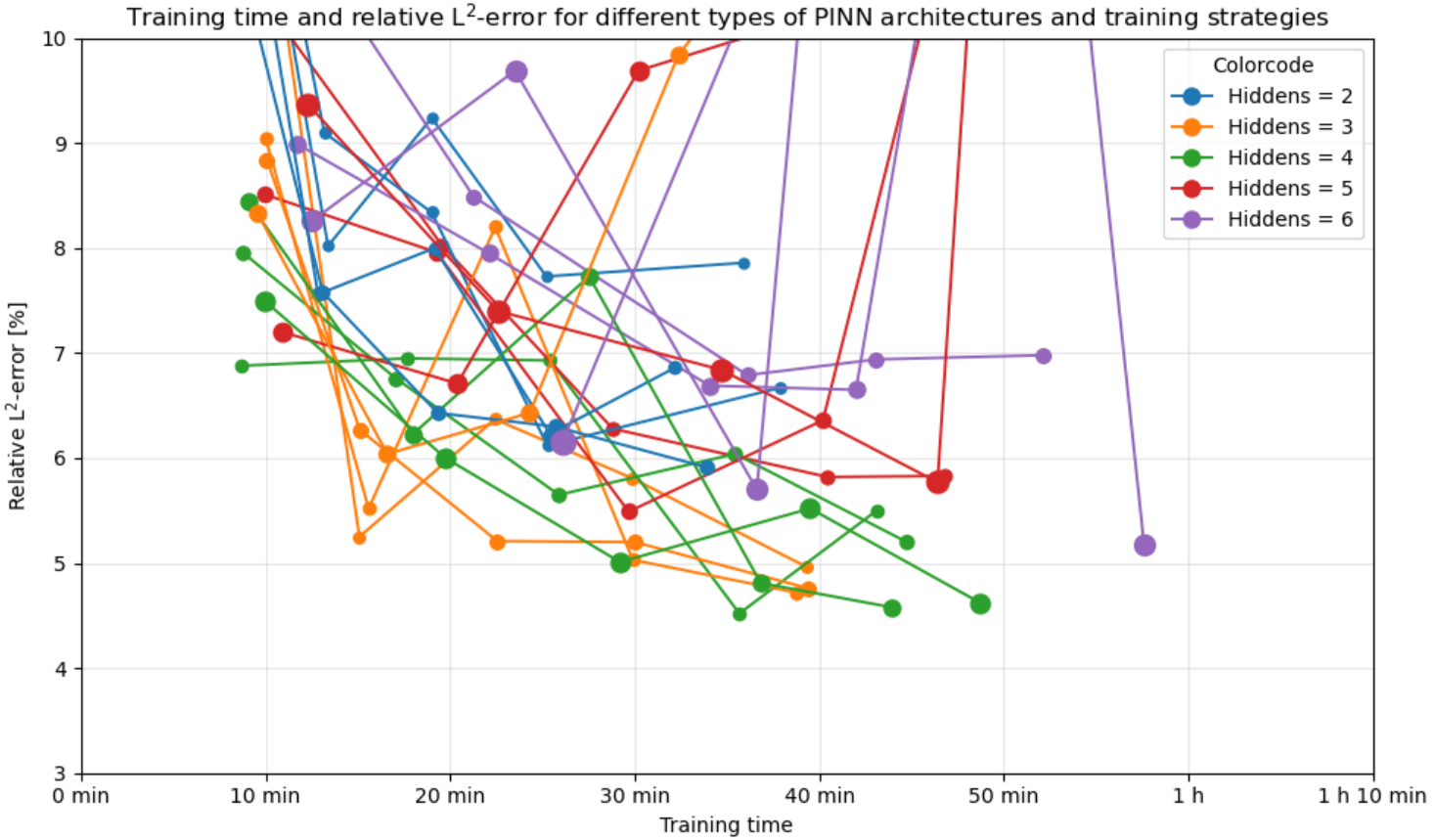
MLPs



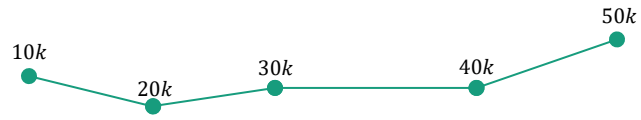
- 100 MLP trainings varying in the
  - amount of layers $\{2,3,4,5,6\}$
  - amount of neurons per layer $\{48,64,96,128\}$
  - amount of iterations $\{10k, 20k, 30k, 40k, 50k\}$ (Adam optimizer incl. LR decay)

PINN Results for Silicidation Problem

MLPs

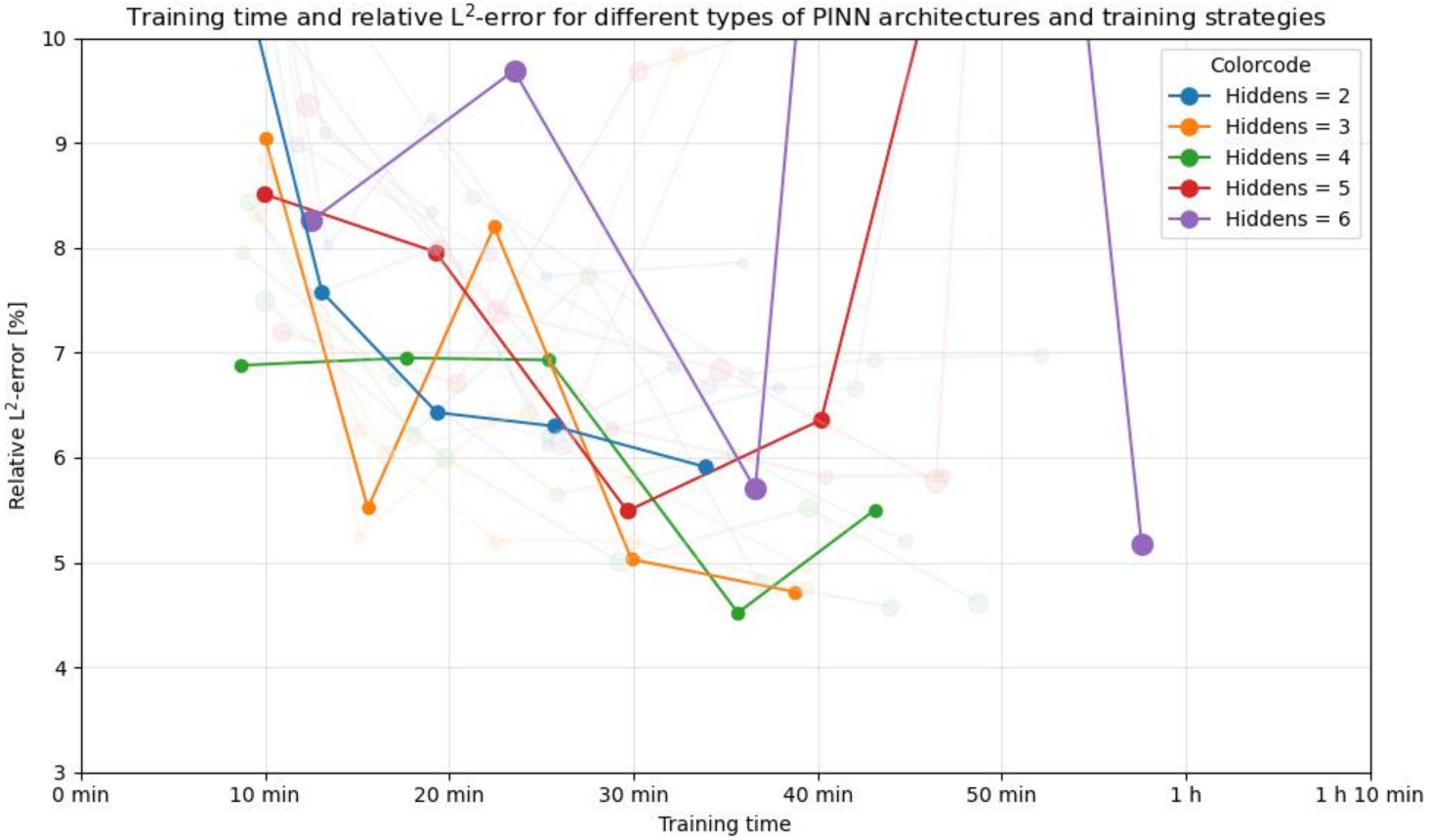


- 100 MLP trainings varying in the
  - amount of $\mathbf{layers}$ $\{2,3,4,5,6\}$ layers
  - amount of neurons per layer $\{48,64,96,128\}$
  - amount of iterations $\{10k, 20k, 30k, 40k, 50k\}$ (Adam optimizer incl. LR decay)
  - Clustering and colorcoding



PINN Results for Silicidation Problem

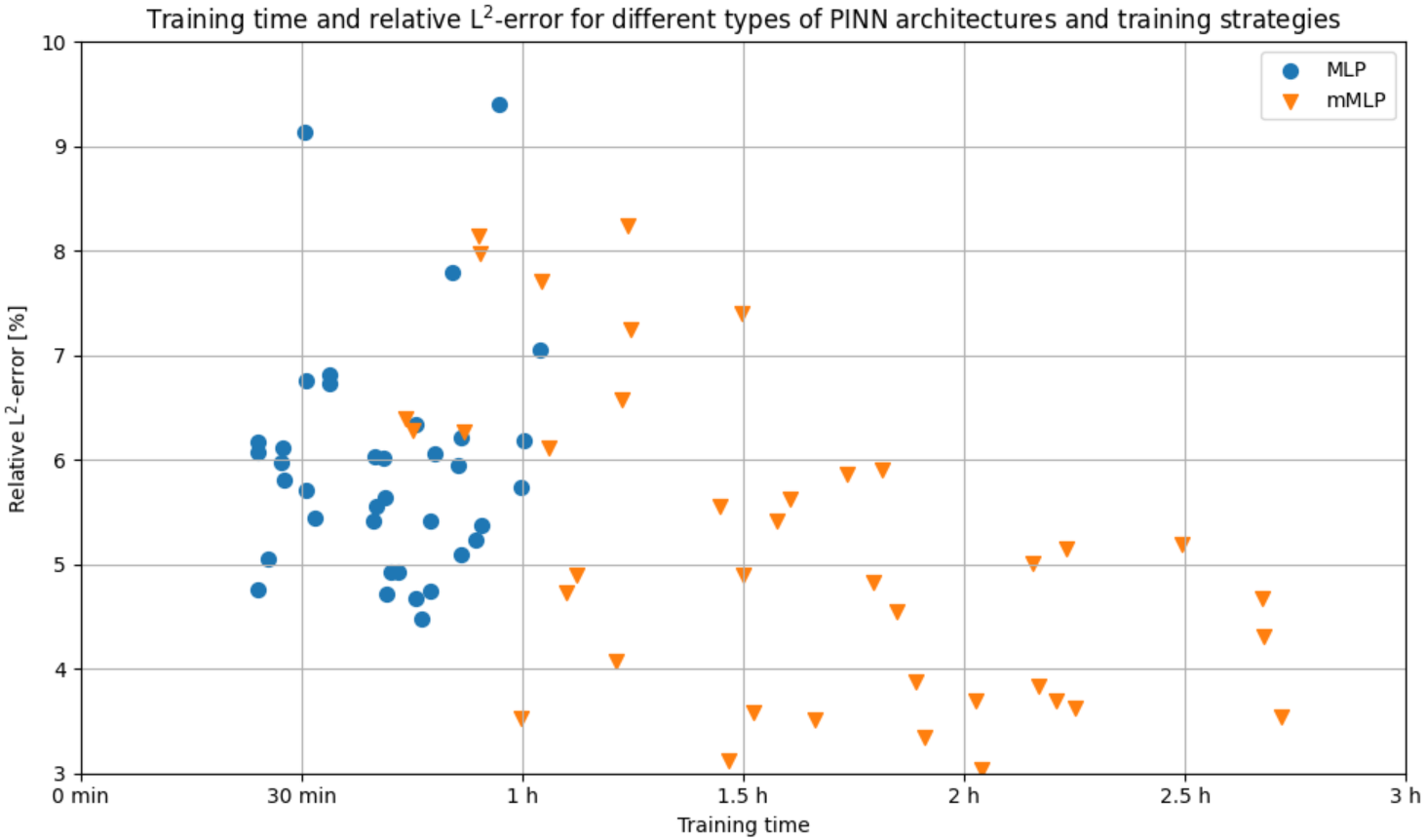
MLPs



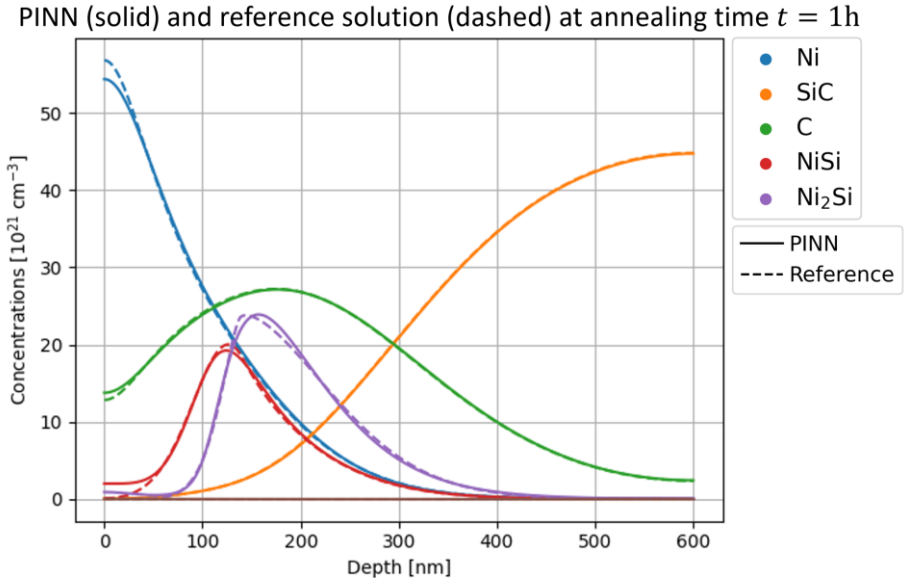
- 100 MLP trainings show …
  - best setups with errors of 4%-5%
  - (unsteady) convergence over training (10k – 50k steps) for minor amount of hidden layers
  - overfitting for high(er) amount of hidden layers

PINN Results for Silicidation Problem

mMLPs



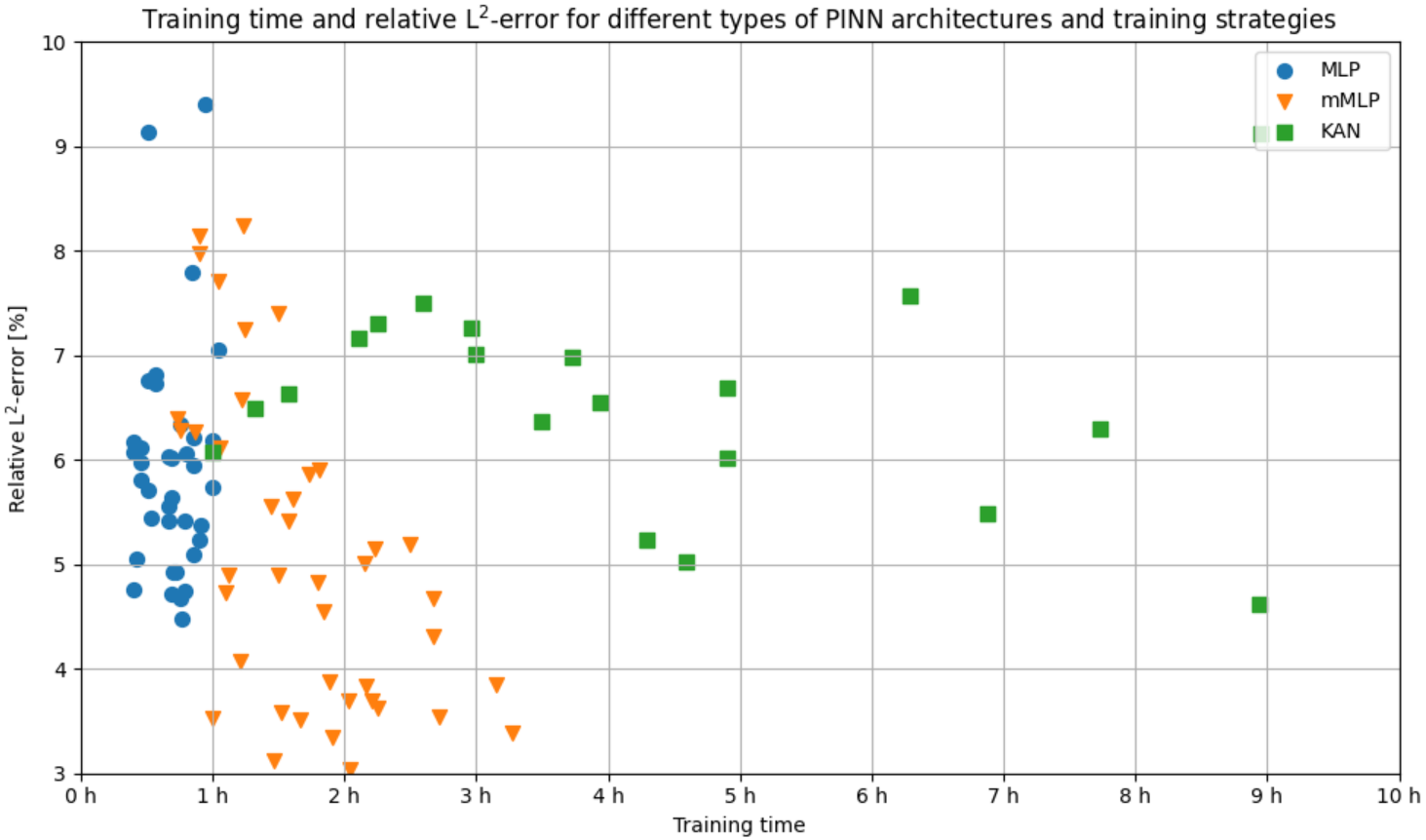
- Modified MLPs lead to lower error but require longer training time
- Best accuracy: $\sim 3.0\ \%$.



Fraunhofer IISB

PINN Results for Silicidation Problem

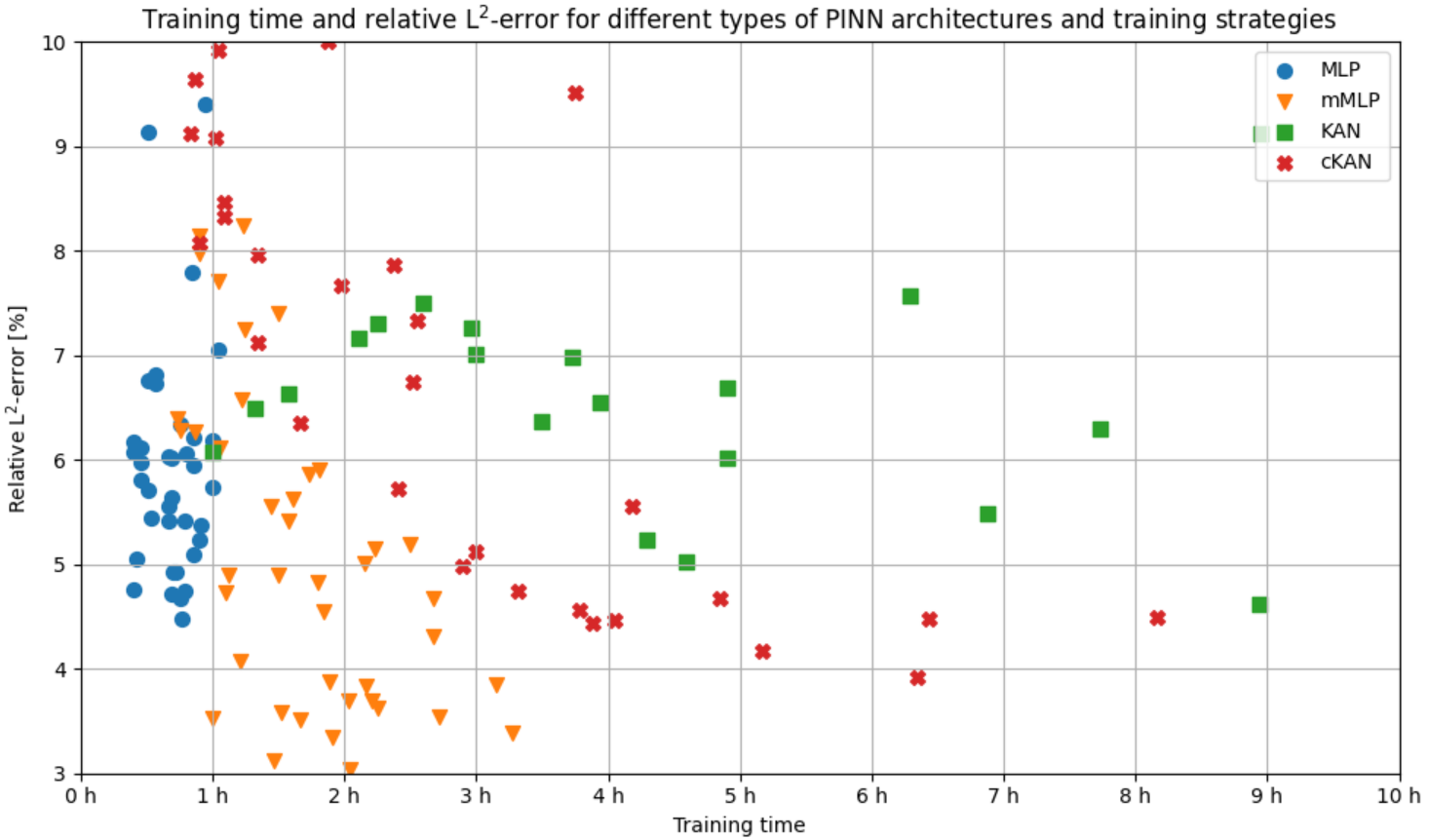
KANs



- KANs need a much longer train time and hardly reach MLP accuracy

PINN Results for Silicidation Problem

cKANs



- KANs need a much longer train time and hardly reach MLP accuracy
- cKANs are faster and more stable to train and reach lower error than KANs, but did not beat mMLPs

Takeaway:

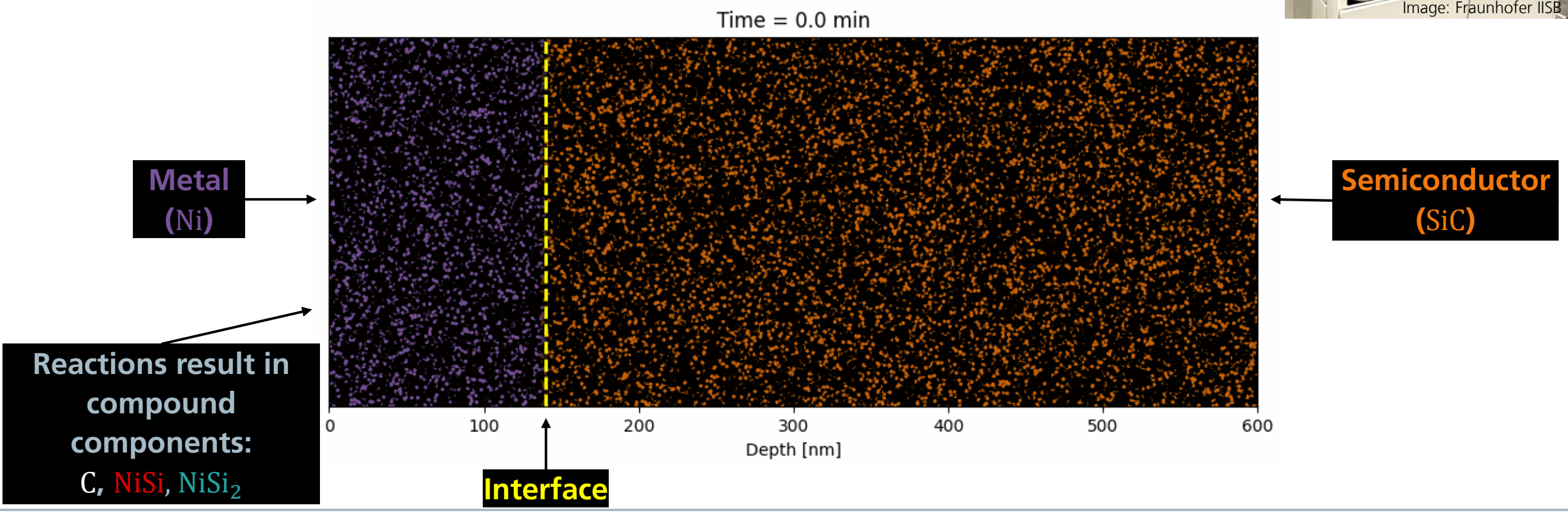
- mMLP reach the lowest error
- (m)MLP reach the best efficiency (error vs. training time)
- Explainable (c)KAN strategies reach good results but require 2-3 times of the (m)MLP training time

Silicidation

Visualisation of the 1D results in a 2D contact.
(Follow-Up Tasks: Resistance calculation & optimization)

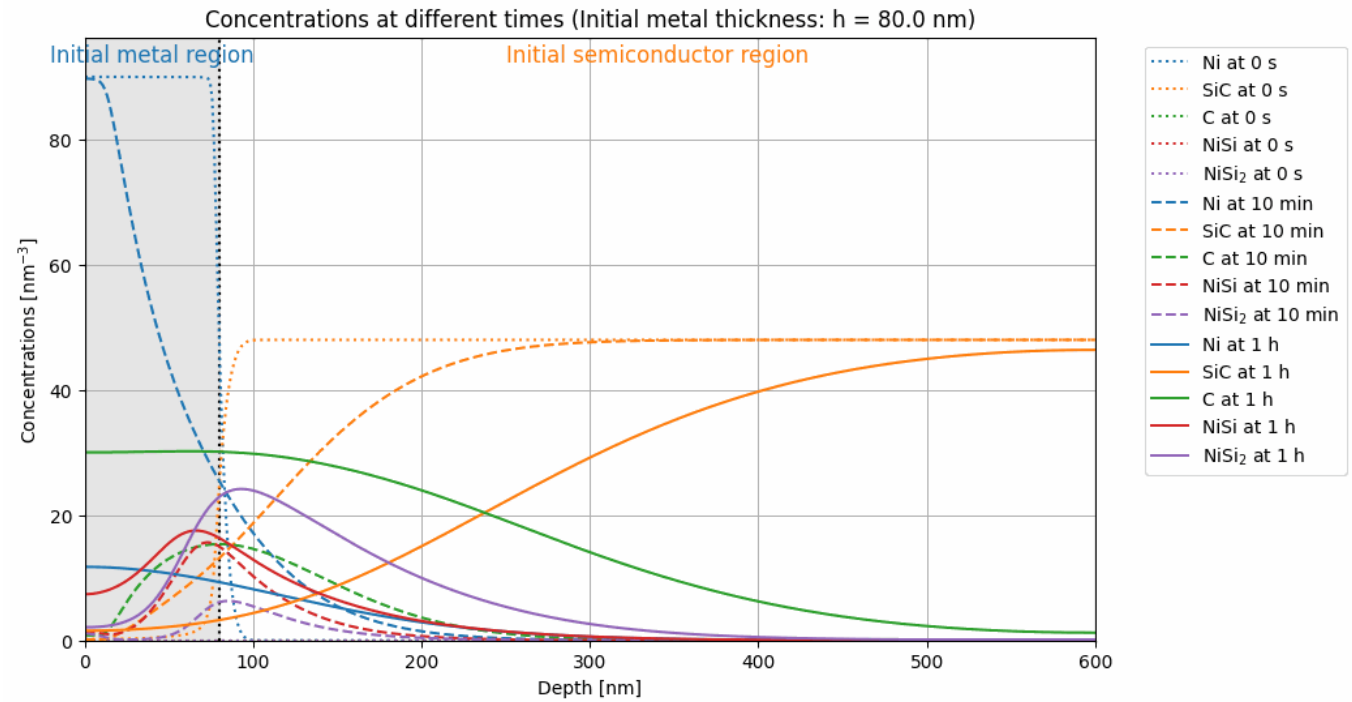
Apply RTP („rapid thermal processing“)

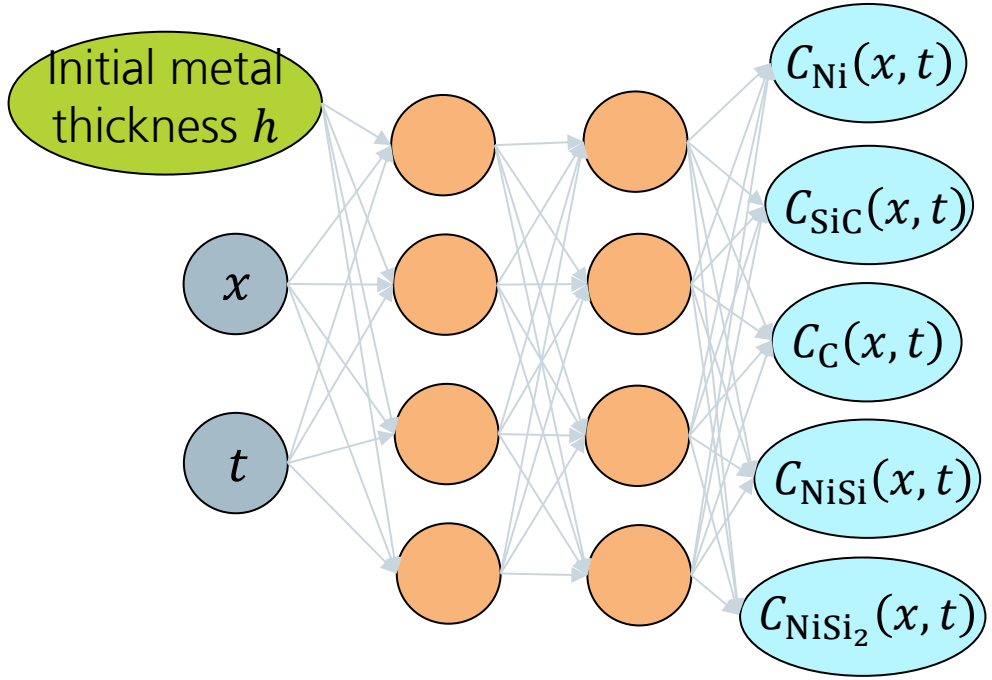
Image: Fraunhofer IISB



Extension to Parametric Model

- So far: Always considered fixed setting.
- Can be extended to „physics-informed neural operator“, e.g., by making initial metal thickness part of input.



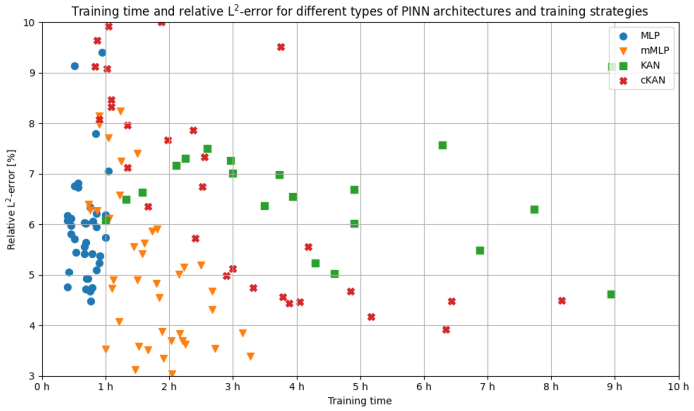


- Evaluation in general settings without needing to retrain model.
- Fast to evaluate surrogate model that can, e.g., be used in optimization pipelines.
- More details provided in the talk by Dr. Christopher Straub later!

SciML Sum-Up

- **Reference simulation:**
  - Time-adaptive FDM solver PROMIS (developed by P. Pichler et al., written in Fortran) with spacial step size 2 nm
  - Simulation time (CPU): ~ 5 seconds

- **SciML – PINN Training time for static setup**
  - Training via DeepXDE using Pytorch on a Tesla Quadro RTX 5000
  - Training time: ~0.5h MLP, ~1h mMLP, 4h-7h KAN and 4h-9h cKAN
  - Inference: 1-2ms for 3 time-steps (for all architectures and the same for CPU / GPU)



- **SciML – PINN Training time for parametric setup (PINO)**
  - Training via DeepXDE using Pytorch on a Tesla V100
  - Training time: 3h mMLP
  - Inference: CPU 28ms for 3 time-steps and 121 thickness values (as in animation)
    GPU 7ms for 3 time-steps and 121 thickness values (as in animation)

Parameteric Setup with 3 timesteps and 121 thickness values:

Classic FDM: 10min
PINO: ~28ms (Inference on CPU)

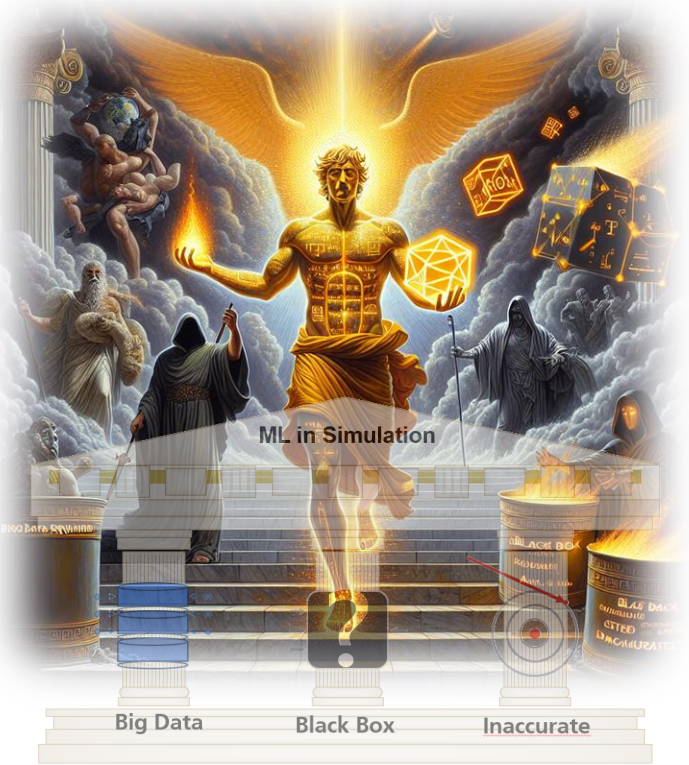
Conclusion and outlook

Key takeaways

- SciML bridges simulation, ML (and measurement) to counter the myths:
  not inaccurate, not big-data-only, not a black box (phy-loss & KAN improves interpretability)
- PINNs solve the Ni–SiC reaction–diffusion PDE with reflecting boundaries;
  PINO extends to parametric surrogates (e.g., thickness h)
- mMLP PINNs delivered the best accuracy/efficiency
  cKAN was more stable than KAN but did not beat mMLP

Outlook

- Expand SciML to higher-dimensional inputs
  (temperature, diffusivities, time grids; multi-parameter, multi-physics, …)
- Improve convergence for complex, non-linear, multi-scale problems, the integration of data, and larger NN sizes
- Integration of SciML into engineering calculations/optimizations and process control (RTP)



Contact

Andreas Rosskopf
Department Head
Modeling and Artificial Intelligence
Tel. +49 9131 761-153
andreas.rosskopf@iisb.fraunhofer.de

Fraunhofer Institute for Integrated Systems and Device Technology IISB
Schottkystraße 10
91058 Erlangen
http://www.iisb.fraunhofer.de

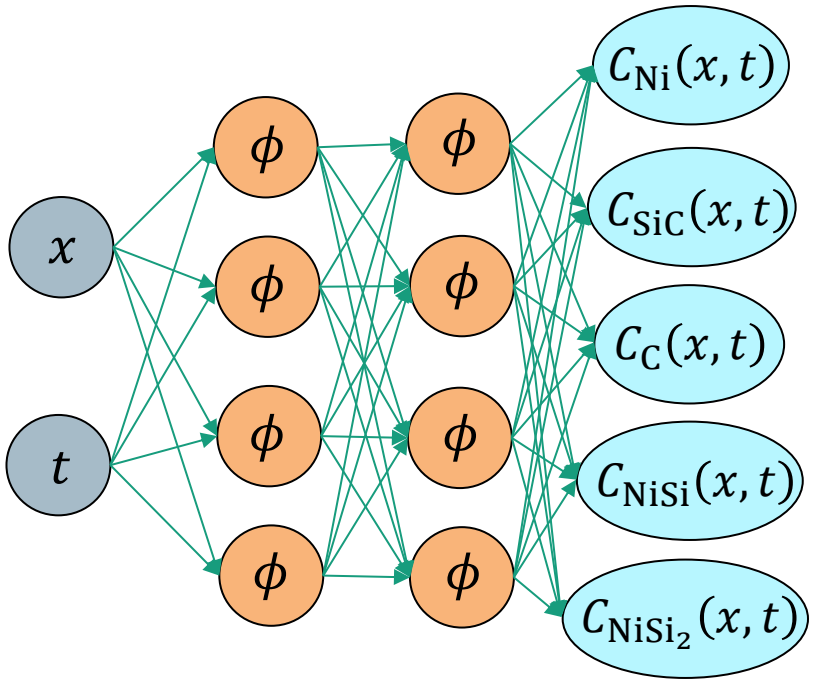
Fraunhofer IISB

Fraunhofer-Institut für Integrierte Systeme und Bauelemente-technologie IISB

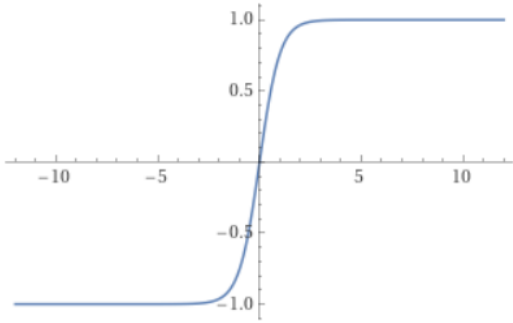
PINN Architectures

MLP

Most common architecture: „multilayer perceptron“ (MLP) = „fully connected neural network“.



- Trainable: Weights on edges.
- Fixed non-linear activation function ϕ on nodes, e.g., $\phi = \tanh$.



PINN Architectures

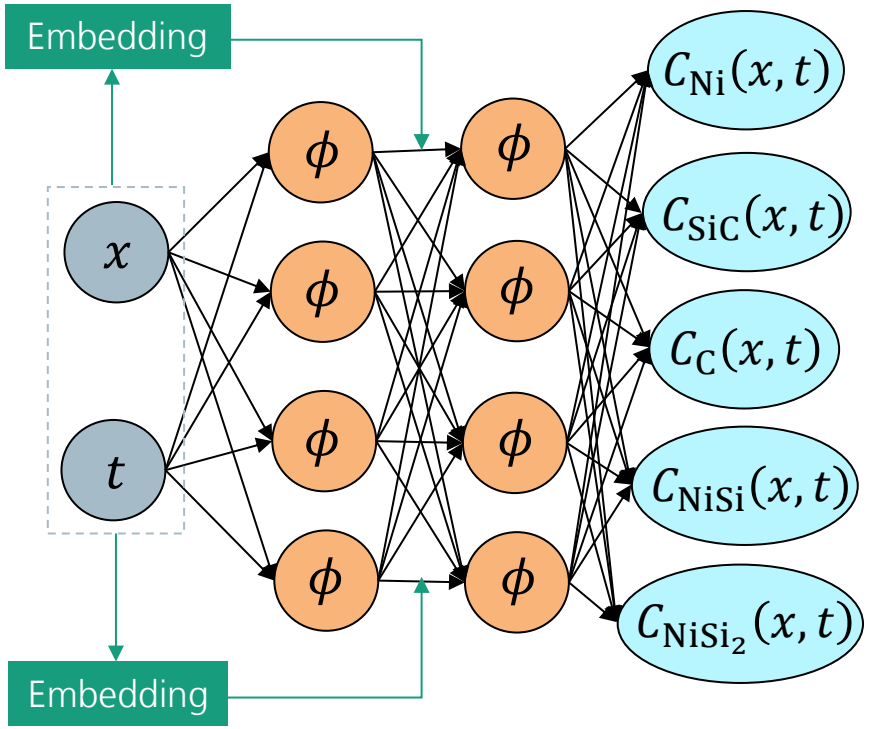
mMLP

Methods and Algorithms for Scientific Computing

Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks

Authors: Sifan Wang, Yujun Teng, and Paris Perdikaris | AUTHORS INFO & AFFILIATIONS

Extension of MLP: „modified MLP“ (mMLP).



- Learnable embedding of input coordinates that is used in all subsequent layers.
- Better approximation of multiplicative interactions between input coordinates.
- Inspired by attention mechanism.

PINN Architectures

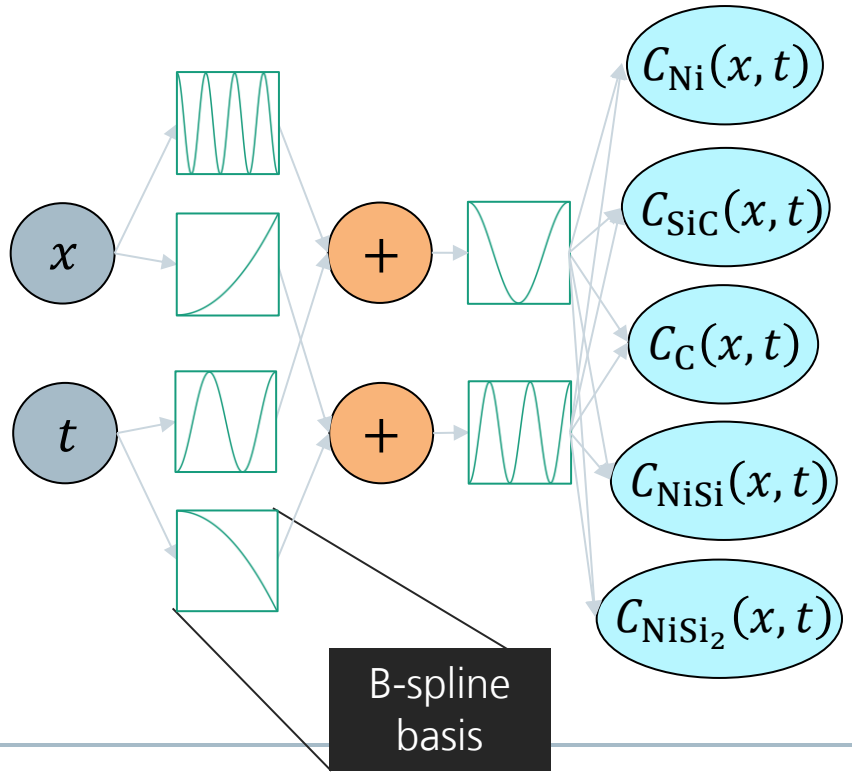
KAN

Alternative to MLPs: „Kolmogorov-Arnold Networks“:

Published in April ‘24.
~800 citations.

KAN: Kolmogorov–Arnold Networks

Ziming Liu[1,4*] Yixuan Wang[2] Sachin Vaidya[1] Fabian Ruehle[3,4]
James Halverson[3,4] Marin Soljačić[1,4] Thomas Y. Hou[2] Max Tegmark[1,4]
[1] Massachusetts Institute of Technology
[2] California Institute of Technology
[3] Northeastern University
[4] The NSF Institute for Artificial Intelligence and Fundamental Interactions



- Learn non-linear functions in nodes via B-spline basis representation.
- Fixed sum operation on edges.
- Better interpretability of model by analyzing learned functions → allows to recover analytical solution formula.

PINN Architectures

cKAN

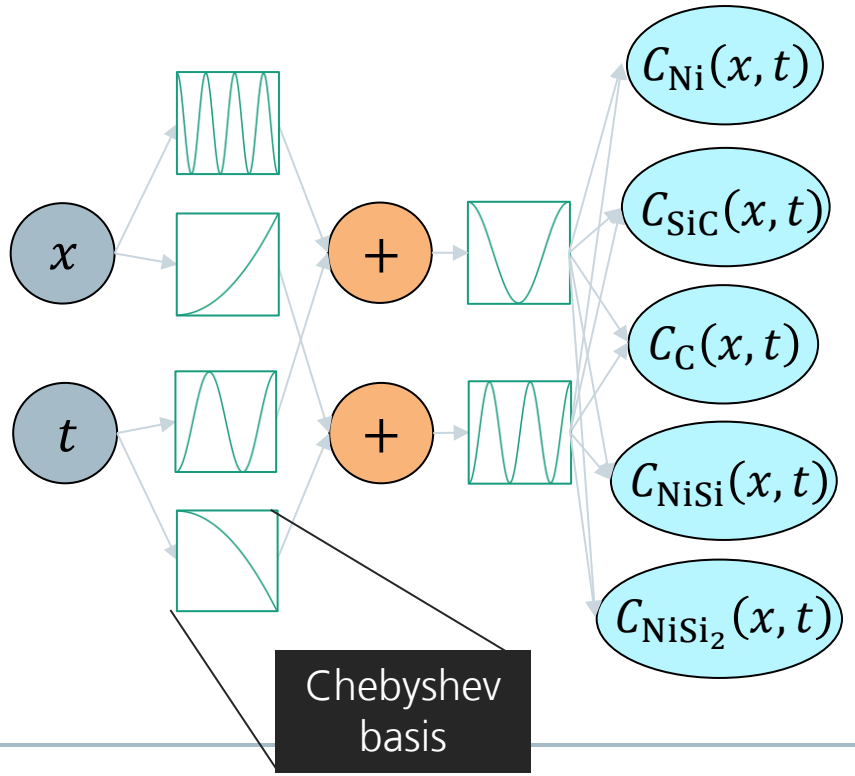
Adaption of KANs: „Chebyshev KANs“ (cKA

Chebyshev Polynomial-Based Kolmogorov-Arnold Networks: An Efficient Architecture for Nonlinear Function Approximation

Sidharth SS[1], Gokul R[1], Anas K P[1], and Keerthana AR[2]

[1]Indian Institute of Science Education and Research Mohali, India
[2]Indian Institute of Science Education and Research Thiruvananthapuram, India

May 2024



- Use Chebyshev polynomial basis representation of learnable functions instead of b-splines.