# Scientific Machine Learning

DI Dr. Bogdan Burlacu
bogdan.burlacu@fh-ooe.at

Josef Ressel Centre for Symbolic Regression
Heuristic and Evolutionary Algorithms Laboratory
University of Applied Sciences Upper Austria

# Short bio


**Dipl.-Ing. Dr. Bogdan Burlacu**

Senior Researcher

University of Applied Sciences Upper Austria, Hagenberg Campus

bogdan.burlacu@fh-ooe.at

https://pure.fh-ooe.at/en/persons/bogdan-burlacu

in https://www.linkedin.com/in/bogdanburlacu84/


## Education and work

- **2009** Systems and Computer Engineering, "Gh. Asachi" Technical Univ. Iași, Romania
- **2008-2011** Software dev, SC Software Development Partnership SRL, Iași, Romania
- **2011-present** Heuristic and Evolutionary Algorithms Laboratory, https://heal.heuristiclab.com
- **2017** PhD. Computer Science, Johannes Kepler University Linz, Austria, epub
- **2018-present** Josef Ressel Centre for Symbolic Regression, https://symreg.at
- **2018-present** Lecturer, Human Centered Computing, MSc course *Data Preprocessing and Analysis*

# Short bio

### Open-source projects

HeuristicLab (https://dev.heuristiclab.com) - heuristic and evolutionary optimization framework (C#)
Operon (https://github.com/heal-research/operon) - large-scale symbolic regression (C++)
Vstat (https://github.com/heal-research/vstat) - numerically stable, SIMD-enabled statistics (C++)
Pappus (https://github.com/heal-research/pappus) - interval and affine arithmetic (C++)

https://github.com/heal-research
https://github.com/foolnotion

# Talk outline

Introduction HEAL, SymReg, SciML

Symbolic Regression Overview

Example: Shape Constraints

Example: Inter-atomic Potentials

# Heuristic and Evolutionary Algorithms Laboratory (HEAL)

**Research group**

- Established at FH Upper Austria in 2005
- 5 professors, 17 research associates
- interns, students (bachelor, master)

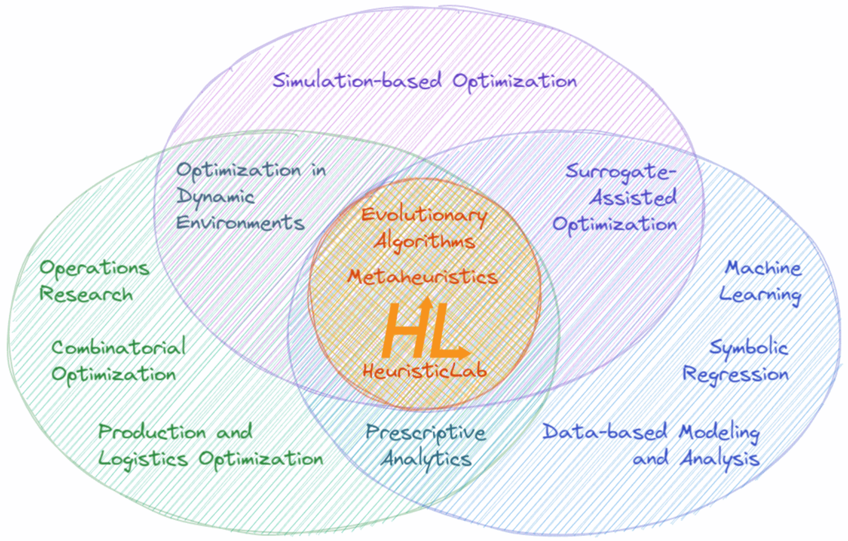
**Research output**

- > 25 research projects, >6M€ funding
- > 200 publications (peer-reviewed)
- > 10 dissertations
- > 60 theses (bachelor and master)

**Scientific and industrial partners**

- https://heal.heuristiclab.com/partners

# Heuristic and Evolutionary Algorithms Laboratory (HEAL)



**HeuristicLab**

- Interactive development, analysis and application of optimization methods
- Plug-in architecture, distributed computing capabilities

**Main focus on combinatorial and symbolic regression/classification problems**

- (probabilistic) traveling salesman, knapsack, bin packing, vehicle routing, job shop scheduling, orienteering, quadratic assignment, and others
- symbolic regression and classification

**Large collection of optimization methods**

- metaheuristics (evolutionary, tabu search, simulated annealing, etc.)
- tree ensemble methods, kernel methods, Gaussian processes, neural networks

# Heuristic and Evolutionary Algorithms Laboratory (HEAL)

**Many interesting optimization problems**

- production scheduling
- stacking in steel industries or container terminals
- dynamic warehouse operations
- cancer diagnosis based on tumor marker data
- properties of synthetic materials
- predictive maintenance
- process modeling based on virtual sensors
  - exhaust gas modeling
  - blast furnace modeling
  - foam quality of firefighting vehicles
  - plasma nitriding
  - granulate homogeneity in plastic recycling machines

**Close cooperation with domain experts**

**Focus on knowledge discovery, insight into the industrial process**

# Josef Ressel Centre for Symbolic Regression (JRZ)

Founded in 2018 as a joint project between the Christian Doppler Research Association, the University of Applied Sciences Upper Austria and three industry partners:

AVL Miba EREMA PLASTIC RECYCLING SYSTEMS

Head of JRZ

Prof. Gabriel Kronberger

Symbolic regression as a technology:

- Algorithm design: deterministic symbolic regression (as opposed to e.g. stochastic/metaheuristic)
- Method development: optimize for accuracy and runtime, keep number of parameters low
- Integration of physical knowledge: shape constraints

**https://symreg.at/**

# Josef Ressel Centre for Symbolic Regression (JRZ)

AVL

- **Development, simulation and testing of powertrain systems**
- Virtual sensors and test bench
- Real-time control on control units
- Reduction of test bench and development times

Miba

- **Friction materials, components for friction systems**
- Improved friction system development through modeling and simulation
- Model- and data warehousing for Tribological systems
- Automatic validation, shorter development cycles

EREMA®
PLASTIC RECYCLING SYSTEMS

- optimize the process for stability and optimal throughput
- virtual sensors
- modeling and optimal control

**What is SciML?**

Blend of traditional scientific mechanistic modeling with machine learning methodologies.

Mechanistic (traditional) modeling: differential equations (by and large)

Machine learning: (deep) neural nets, kernel methods, tree ensemble methods, symbolic regression, etc.

**Main challenges**

Complex applications: complex and multi-scale dynamics, data are sparse and expensive to acquire

High consequence: decisions can have severe outcomes, need for explainable models

Technically challenging: numerical difficulties, computing gradients, running code in HPC environments

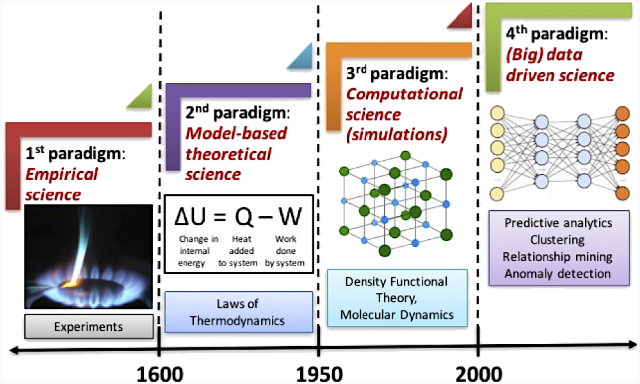
# Scientific Machine Learning

**Focus shift** from *prediction* to *understanding*:

- extracting insights from data requires interpretable models
- models should incorporate physical laws, constraints and other scientific domain knowledge
- physically meaningful behavior and parameters (e.g. Planck's constant, Avogadro's constant)
- robustness and the ability to quantify uncertainty are required for scientific rigour

> "The path from good science to good engineering relies on conducting reproducible simulations that quantitatively explain phenomena, and then being able to document how far those results can be trusted."
> — Erik van der Giessen et al 2020 Modelling Simul. Mater. Sci. Eng. 28 043001, https://doi.org/10.1088/1361-651X/ab7150

# Scientific Machine Learning

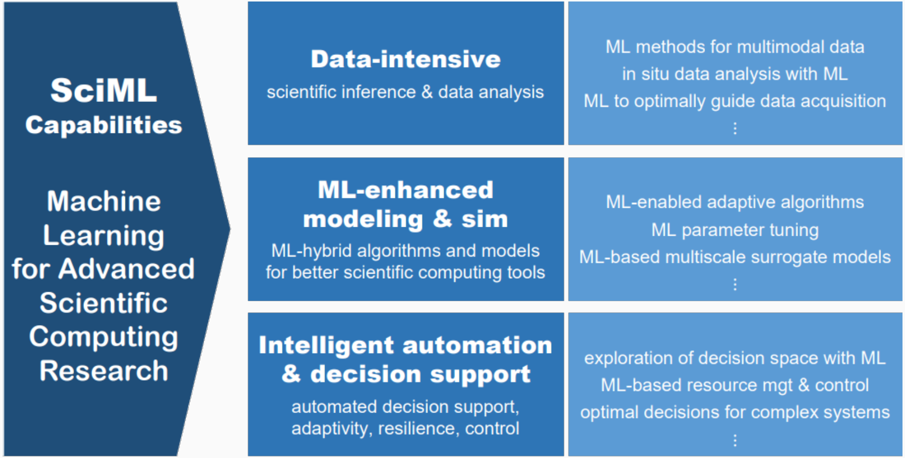



Agrawal and Choudari, *Perspective: Materials informatics and big data: Realization of the "fourth paradigm" of science in materials science*, APL Mater. 4, 053208 (2016), https://doi.org/10.1063/1.4946894

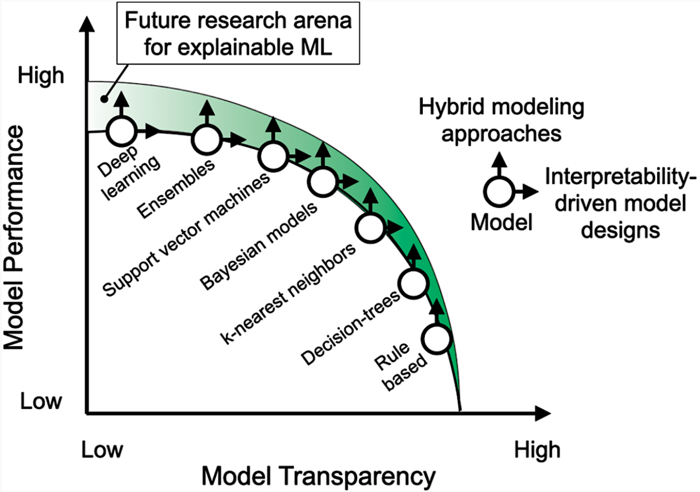
# Scientific Machine Learning


© OSTI.GOV Technical Report: Workshop Report on Basic Research Needs for Scientific Machine Learning

# Scientific Machine Learning

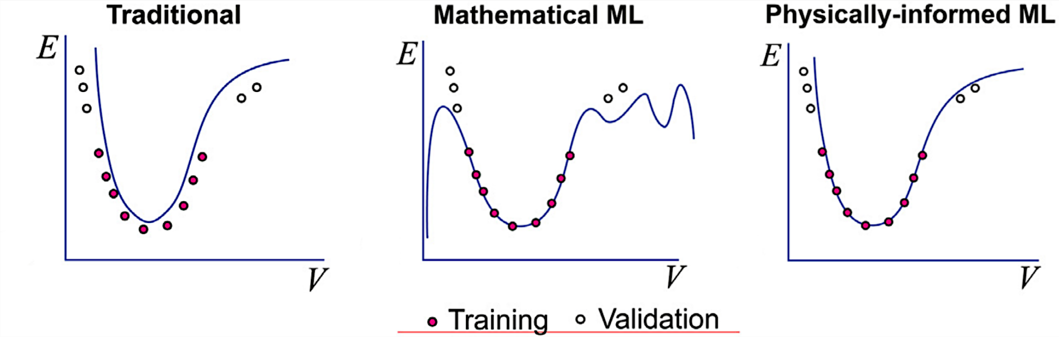



© OSTI.GOV Technical Report: Workshop Report on Basic Research Needs for Scientific Machine Learning




Arrieta et. al, *Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI*, Information Fusion 58, 2020, https://doi.org/10.1016/j.inffus.2019.12.012

# Scientific Machine Learning




Mishin, *Machine-learning interatomic potentials for materials science*, https://doi.org/10.1016/j.actamat.2021.116980

# Symbolic Regression

Finding a symbolic expression that matches data from an unknown function.

Core challenge in physics and engineering
(in principle, NP-hard: Lu et al. 2016 https://doi.org/10.1155/2016/1021378)

An optimization algorithm would have to explore a very large, combinatorial search space:

- Discrete in model structure, continuous in model parameters
- Growing exponentially with the length of the expression

## Symbolic Regression

Let $\mathcal{P}$ be a primitive set (e.g. $+, -, \times, \div, \exp, \sin$, etc.) and $S_{\mathcal{P}}$ the syntactic search space defined by it.

Let $\Phi$ be the space of possible expressions and their parameters, defined as the set of all tuples $(E, \theta)$ where $E \in S_{\mathcal{P}}$ is a symbolic expression and $\theta \in \mathbb{R}^p$ is a vector of coefficients for $E$.

We call a tuple $(E, \theta)$ a *symbolic expression model* $M_{E,\theta} \in \Phi$.

Let $G : \Phi \times \mathbb{R}^{n \times m} \to \mathbb{R}^n$ be a function that evaluates model $M_{E,\theta} \in \Phi$ on some training data $\mathbf{X}$ (with corresponding dependent target $y$) and outputs a prediction $\hat{y} \in \mathbb{R}^n$.

The overall goal of symbolic regression is to find the optimal model

$$M_{\text{opt}} = \underset{M_{E,\theta} \in \Phi}{\arg\min} \frac{1}{2} ||G(M_{E,\theta}, \mathbf{X}) - \mathbf{y}||^2$$

Overarching objective: good extrapolation behavior on unknown (out-of-domain) data.

**Strategies to search model space $\Phi$:**

- Evolutionary: genetic programming, grammatical evolution (guided metaheuristic)
- Generative-enumerative: production rules from a grammar (guided heuristic)
- Hybrid: AI Feynman, DeepSR, EQL, PNN-SR (mixed heuristic)

**Strategies to optimize model coefficients $\theta$:**

- non-linear least squares
- variable projection (if least squares problem is separable)
- stochastic hill climbing
- metaheuristics e.g. CMA-ES

**Algorithm**

- Exact, deterministic, correct, finite number of steps, well understood, proven complexity bounds.

**Heuristic**

- Derived from experience or from empirical evicence
- Hard to study from a theoretical standpoint
- Usually no guarantees about optimality or complexity bounds
- *Useful practical solving tools, producing good results (not necessarily optimal) in a short amount of time*

# Interlude: Heuristics and Metaheuristics

## HEURISTICS FOR INTEGER PROGRAMMING USING SURROGATE CONSTRAINTS

Fred Glover, *University of Colorado*

### ABSTRACT

This paper proposes a class of surrogate constraint heuristics for obtaining approximate, near optimal solutions to integer programming problems. These heuristics are based on a simple framework that illuminates the character of several earlier heuristic proposals and provides a variety of new alternatives. The paper also proposes additional heuristics that can be used either to supplement the surrogate constraint procedures or to provide independent solution strategies. Preliminary computational results are reported for applying one of these alternatives to a class of nonlinear generalized set covering problems involving approximately 100 constraints and 300-500 integer variables. The solutions obtained by the tested procedure had objective function values twice as good as values obtained by standard approaches (*e.g.*, reducing the best objective function values of other methods from 85 to 40 on the average. Total solution time for the tested procedure ranged from ten to twenty seconds on the CDC 6600.

### INTRODUCTION

Heuristic solution methods for integer programming have maintained a noticeably separate existence from algorithms. Algorithms have long constituted the more respectable side of the family, assuring an optimal solution in a finite number of steps. Methods that merely claim to be clever, and do not boast an entourage of supporting theorems and proofs, are accorded a lower status. Algorithms are conceived in analytic purity in the high citadels of academic research, heuristics are midwifed by expediency in the dark corners of the practitioner's lair.

Recently, however, there has been a growing recognition that the algorithms are not always successful, and that their heuristic cousins deserve a chance to prove their mettle. Partly this comes from an emerging awareness that algorithms and heuristics are not as different as once supposed – algorithms, after all, are merely fastidious heuristics in which epsilons and deltas abide by the dictates of mathematical etiquette. It may even be said that algorithms exhibit a somewhat compulsive aspect, being denied the freedom that would allow an occasional inconsistency or an exception to ultimate convergence. (Unfortunately, ultimate convergence sometimes acquires a religious significance; it seems not to happen in this world.)

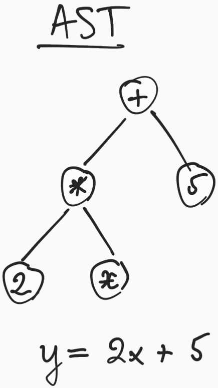
**Metaheuristic**

Iterative master process that guides and modifies the operations of subordinate heuristics.

- combines different concepts for *exploring* and *exploiting* the search space
- uses learning strategies in order to structure information
- may manipulate a complete or incomplete solution or set of solutions at each iteration
- subordinate heuristics may be high or low level procedures, or simple local search, or just a construction method

May be inspired from natural behaviors, emergent phenomena, etc.

May push the analogy too far https://github.com/fcampelo/EC-bestiary




- Abstract syntax trees typically used for representation
- Easy to manipulate
- Fast evaluation and amenable to autodiff (e.g. using duals $a + b\varepsilon, \varepsilon^2 = 0$)
- Can be compiled to machine code (LLVM, asmjit), CUDA/openCL kernels
- Straightforward to transform (algebraic rules/identities) or prune
- Derivatives can be computed symbolically
- Coefficients tunable with non-linear least squares (e.g. leaf nodes $2, 5$)

# Symbolic Regression

## Genetic Programming

- Simulates the process of natural evolution ("survival of the fittest")
- Stochastic search guided by heuristics and a cost ("fitness") function
- Relies on genetic operators: selection, crossover, mutation
- Very general: no need for *a priori* knowledge, can find model structure and coefficients

## Extensions

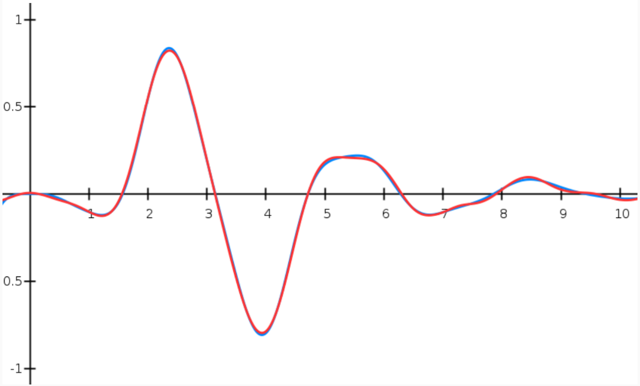
- Memetic approaches (+ local optimization: gradient-based, trust region etc.)
- Parallel/distributed approaches: island models, parallel tempering
- Multi-objective strategies (Pareto-based, decomposition-based)

## Caveats

- Non-deterministic
- Can produce overly complex expressions
- Tradeoff between parsimony and accuracy

## Genetic Programming

$$\frac{c_0 \sin(c_1 \cos(c_2\, x) + c_3 \cos(c_4\, x)}{c_5\, x + \cos(c_6\, x + c_7 \sin(\sin(\cos(c_8\, x))) + c_9}$$



**Deep learning and other approaches**

- PINNs: “physics-informed NN” - eqn. learners, PDE solvers (https://doi.org/10.1038/s41598-021-92278-w)
- GNN: deep learning with inductive biases (https://doi.org/10.48550/arXiv.2006.11287)
- AI-Feynman: divide and conquer, heuristic approach (https://arxiv.org/abs/2006.10782)
- DeepSR: RNN + risk seeking policy gradients (https://arxiv.org/abs/1912.04871)
- EQL: equation learning network (http://proceedings.mlr.press/v80/sahoo18a.html)
- PNNs: parsimonious neural networks (https://www.nature.com/articles/s41598-021-92278-w)

# Physics-Informed Neural Networks (PINNs)

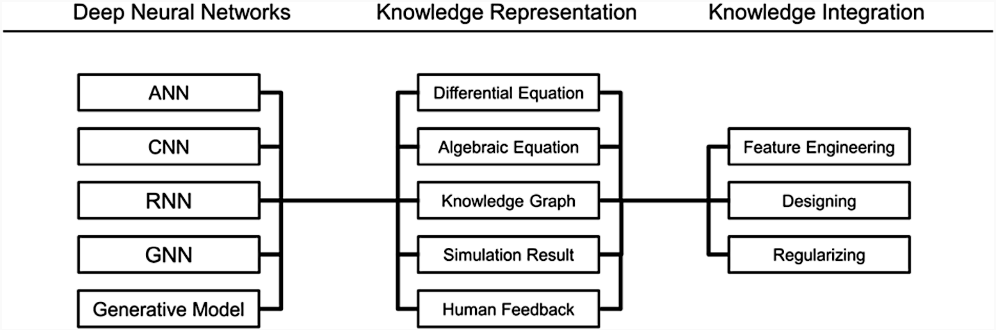
**Motivation**

- black box nature of deep learning hinders explainability and new scientific discovery
- critical need to tackle "knowledge integration" in the deep learning pipeline
- knowledge integration can be: feature engineering, network architecture design, regularization, etc.

**Approach**

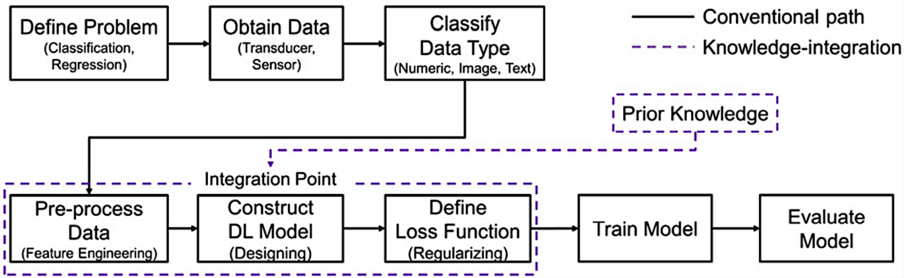
- encode model equations (e.g. PDEs) as part of the network itself
- taking into acount the physics of the problem
- loss function includes PDE residuals as well as terms for initial and boundary conditions

# Physics-Informed Neural Networks (PINNs)




Kim et al. 2021 https://doi.org/10.1007/s12206-021-0342-5

# Physics-Informed Neural Networks (PINNs)



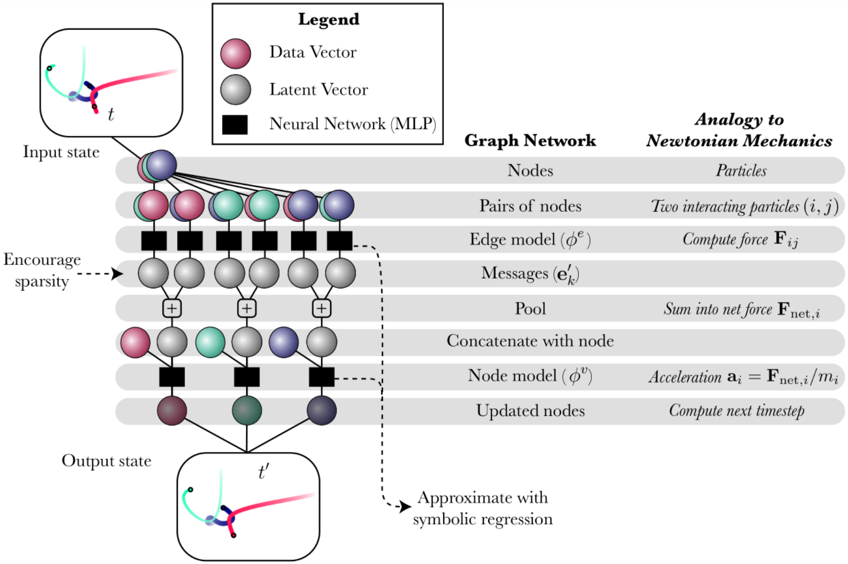

Kim et al. 2021 https://doi.org/10.1007/s12206-021-0342-5

# Deep Learning with Inductive Biases (Cranmer et al. 2020)

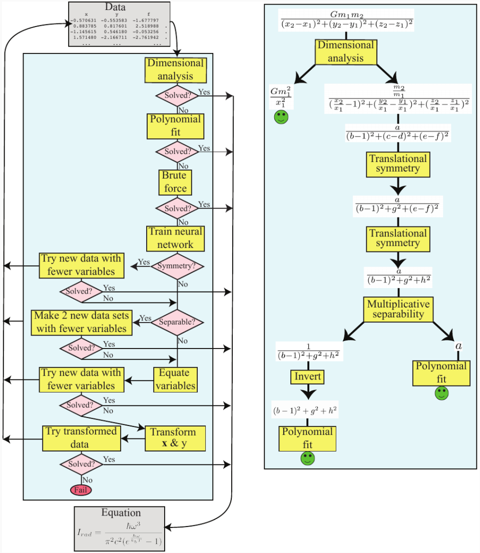
**Approach**

- Inductive bias:
  - GNN equivariant under particle permutations, end-to-end differentiable
  - separate and interpretable internal functions: edge model $\Phi^e$, node model $\Phi^v$, global model $\Phi^u$
- Train GNN in a supervised setting encouraging sparse latent representations
  - sparsity promoted via regularization terms to the loss
- Apply symbolic regression to learned components to extract explicit physical relations
  - fit compact closed-form analytical expressions for $\Phi^e$, $\Phi^v$, $\Phi^u$ using SR (*eureka* software)
- Able to learn Newtonian and Hamiltonian dynamics, prediction of dark matter halo ("overdensity")

# Deep Learning with Inductive Biases (Cranmer et al. 2020)

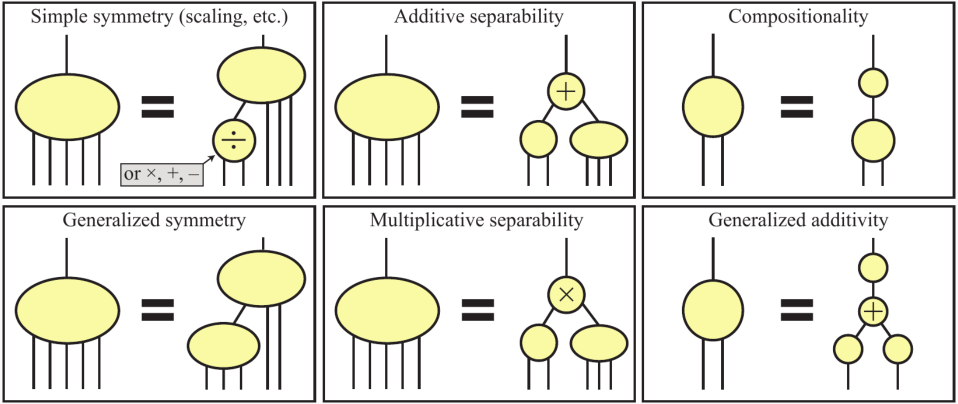


# AI Feynman (Udrescu et al. 2020)




- Collection of heuristics for discovering physical laws
- AST represented as a graph
- Recursive application of solvers and problem decomposition heuristics
- Exploits neural networks, graph modularity, hypothesis testing, normalizing flows

# AI Feynman




https://cbmm.mit.edu/publications/ai-feynman-20-pareto-optimal-symbolic-regression-exploiting-graph-modularity

# AI Feynman

| | Equation | Symmetries |
|---|---|---|
| 1 | $\delta = -5.41 + 4.9\frac{\alpha-\beta+\gamma/\chi}{3\chi}$ | TC |
| 2 | $\chi = 0.23 + 14.2\frac{\alpha+\beta}{3\gamma}$ | TS |
| 3 | $\beta = 213.80940889\left(1 - e^{-0.54723748542\alpha}\right)$ | |
| 4 | $\delta = 6.87 + 11\sqrt{\alpha\beta\gamma}$ | P |
| 5 | $V = \left[R_1^{-1} + R_2^{-1} + R_3^{-1} + R_4^{-1}\right]^{-1} I_0 \cos\omega t$ (Parallel resistors) | PGSM |
| 6 | $I_0 = \frac{V_0}{\sqrt{R^2+\left(\omega L - \frac{1}{\omega C}\right)^2}}$ (RLC circuit ) | MG |
| 7 | $I = \frac{V_0 \cos\omega t}{\sqrt{R^2+\left(\omega L - \frac{1}{\omega C}\right)^2}}$ (RLC circuit) | MG |
| 8 | $V_2 = \left(\frac{R_2}{R_1+R_2} - \frac{R_x}{R_x+R_3}\right)V_1$ (Wheatstone bridge) | SGMA |
| 9 | $v = c\frac{(v_1+v_2+v_3)/c+v_1v_2v_3/c^3}{1+(v_1v_2+v_1v_3+v_2v_3)/c^2}$ (Velocity addition) | AG |
| 10 | $v = c\frac{(v_1+v_2+v_3+v_4)/c+(v_2v_3v_4+v_1v_3v_4+v_1v_2v_4+v_1v_2v_3)/c^3}{1+(v_1v_2+v_1v_3+v_1v_4+v_2v_3+v_2v_4+v_3v_4)/c^2+v_1v_2v_3v_4/c^4}$ (Velocity addition) | GA |
| 11 | $z = (x^4 + y^4)^{1/4}$ ($L_4$-norm) | AC |
| 12 | $w = xyz - z\sqrt{1-x^2}\sqrt{1-y^2} - y\sqrt{1-x^2}\sqrt{1-z^2} - x\sqrt{1-y^2}\sqrt{1-z^2}$ | GA |
| 13 | $z = \frac{xy+\sqrt{1-x^2-y^2+x^2y^2}}{y\sqrt{1-x^2}-x\sqrt{1-y^2}}$ | A |
| 14 | $z = y\sqrt{1-x^2} + x\sqrt{1-y^2}$ | A |
| 15 | $z = xy - \sqrt{1-x^2}\sqrt{1-y^2}$ | A |
| 16 | $r = \frac{a}{\cot(\alpha/2)+\cot(\beta/2)}$ (Incircle) | GMAC |

https://cbmm.mit.edu/publications/ai-feynman-20-pareto-optimal-symbolic-regression-exploiting-graph-modularity

# Deep Symbolic Regression (Petersen et al. 2021)

## Approach

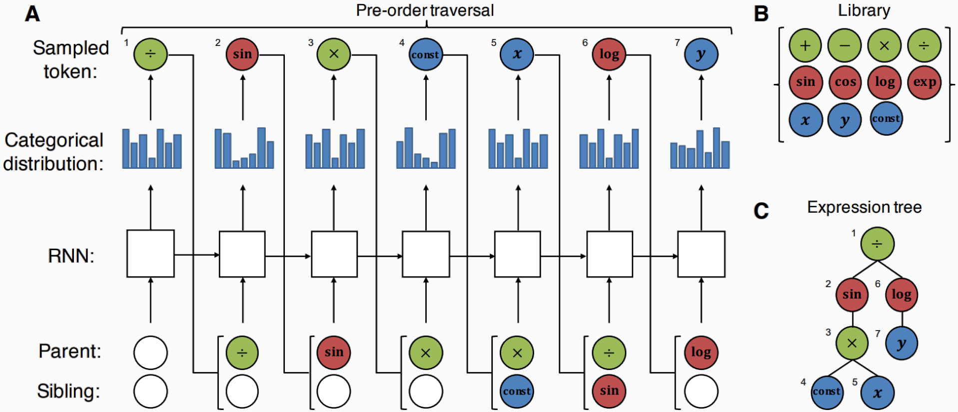
- use a large model (neural network) to search the space of small models (symbolic expressions)
- exploits background knowledge about the form of the expressions
- generates mathematical expressions using a recurrent neural network (RNN)
- the RNN is trained via a risk-seeking policy gradient
- AST represented as sequence of tokens (preorder)
- AST generated sequentially using sampling from RNN-generated token distribution + constraints
- tree coefficients further optimized with nonlinear least squares (BFGS)

## Reward function

$$R(\tau) = \frac{1}{1 + \mathrm{NRMSE}} \quad \text{("squashed" NRMSE)}$$

$$\mathrm{NRMSE} = \frac{1}{\sigma_y}\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(X_i)\right)^2}$$

# Deep Symbolic Regression



# Deep Symbolic Regression

**Algorithm 1** Deep symbolic regression with risk-seeking policy gradient

**input** learning rate $\alpha$; entropy coefficient $\lambda_{\mathcal{H}}$; risk factor $\varepsilon$; batch size $N$; reward function $R$
**output** Best fitting expression $\tau^\star$

Initialize RNN with parameters $\theta$, defining distribution over expressions $p(\cdot|\theta)$
**repeat**
    $\mathcal{T} \leftarrow \{\tau^{(i)} \sim p(\cdot|\theta)\}_{i=1}^{N}$ ▷ Sample $N$ expressions (Alg. 2 in Appendix A)
    $\mathcal{T} \leftarrow \{\text{OptimizeConstants}(\tau^{(i)}, R)\}_{i=1}^{N}$ ▷ Optimize constants w.r.t. reward function
    $\mathcal{R} \leftarrow \{R(\tau^{(i)})\}_{i=1}^{N}$ ▷ Compute rewards
    $R_\varepsilon \leftarrow (1-\varepsilon)$-quantile of $\mathcal{R}$ ▷ Compute reward threshold
    $\mathcal{T} \leftarrow \{\tau^{(i)} : R(\tau^{(i)}) \geq R_\varepsilon\}$ ▷ Select subset of expressions above threshold
    $\mathcal{R} \leftarrow \{R(\tau^{(i)}) : R(\tau^{(i)}) \geq R_\varepsilon\}$ ▷ Select corresponding subset of rewards
    $\hat{g}_1 \leftarrow \text{ReduceMean}((\mathcal{R} - R_\varepsilon)\nabla_\theta \log p(\mathcal{T}|\theta))$ ▷ Compute risk-seeking policy gradient
    $\hat{g}_2 \leftarrow \text{ReduceMean}(\lambda_{\mathcal{H}} \nabla_\theta \mathcal{H}(\mathcal{T}|\theta))$ ▷ Compute entropy gradient
    $\theta \leftarrow \theta + \alpha(\hat{g}_1 + \hat{g}_2)$ ▷ Apply gradients
    **if** $\max \mathcal{R} > R(\tau^\star)$ **then** $\tau^\star \leftarrow \tau^{(\arg\max \mathcal{R})}$ ▷ Update best expression
**return** $\tau^\star$

Table 1: Recovery rate comparison of DSR and five baselines on the Nguyen symbolic regression benchmark suite. A bold value represents statistical significance ($p < 10^{-3}$) across all benchmarks.

| Benchmark | Expression | DSR | PQT | VPG | GP | Eureqa | Wolfram |
|---|---|---|---|---|---|---|---|
| Nguyen-1 | $x^3 + x^2 + x$ | 100% | 100% | 96% | 100% | 100% | 100% |
| Nguyen-2 | $x^4 + x^3 + x^2 + x$ | 100% | 99% | 47% | 97% | 100% | 100% |
| Nguyen-3 | $x^5 + x^4 + x^3 + x^2 + x$ | 100% | 86% | 4% | 100% | 95% | 100% |
| Nguyen-4 | $x^6 + x^5 + x^4 + x^3 + x^2 + x$ | 100% | 93% | 1% | 100% | 70% | 100% |
| Nguyen-5 | $\sin(x^2)\cos(x) - 1$ | 72% | 73% | 5% | 45% | 73% | 2% |
| Nguyen-6 | $\sin(x) + \sin(x + x^2)$ | 100% | 98% | 100% | 91% | 100% | 1% |
| Nguyen-7 | $\log(x+1) + \log(x^2+1)$ | 35% | 41% | 3% | 0% | 85% | 0% |
| Nguyen-8 | $\sqrt{x}$ | 96% | 21% | 5% | 5% | 0% | 71% |
| Nguyen-9 | $\sin(x) + \sin(y^2)$ | 100% | 100% | 100% | 100% | 100% | – |
| Nguyen-10 | $2\sin(x)\cos(y)$ | 100% | 91% | 99% | 76% | 64% | – |
| Nguyen-11 | $x^y$ | 100% | 100% | 100% | 7% | 100% | – |
| Nguyen-12 | $x^4 - x^3 + \frac{1}{2}y^2 - y$ | 0% | 0% | 0% | 0% | 0% | – |
| | Average | **83.6%** | 75.2% | 46.7% | 60.1% | 73.9% | – |

# Equation Learner (Sahoo et al. 2018)

## Approach

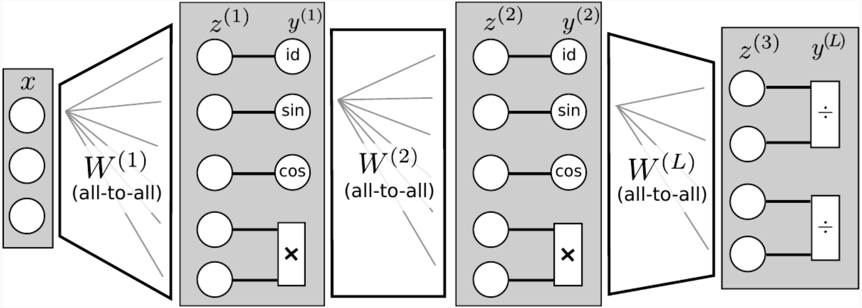
- fully connected feed forward network structure
- nodes/neurons include base functions (e.g. $+$, $-$, $\times$, $\div$, $\sin$, $\cos$, etc)
- compositions of base functions are generated across network layers
- able to learn relatively simple formula, able to control cart-pendulum system

## Regularized division

- division introduces poles with abrupt changes in convexity and diverging function values
- real systems are assumed to be unable to generate data at the pole itself (because natural quantities do not diverge)
- therefore a single branch of the hyperbola $1/b, b > 0$ suffices as a basis function
- regularized division only used in the output layer

$$h^{\theta}(a, b) = \begin{cases} \frac{a}{b} & \text{if } b > \theta \\ 0 & \text{otherwise} \end{cases}$$

# Equation Learner (Sahoo et al. 2018)




- trained by stochastic gradient descent (mini batches + Adam)
- Lasso-like loss function (fully differentiable in its free parameters)
- penalty term $P^\theta$ for small and negative denominators

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} ||f(x_i) - y_i||^2 + \lambda \sum_{l=1}^{L} |W^{(l)}|_1 + P^\theta$$

# Parsimonious Neural Networks (Desai and Strachan 2021)

**Application in Material Science**

Learn equations of motion that govern the Hamiltonian dynamics of a particle

Discover melting laws from experimental data

**Approach**

- data obtained generated using molecular dynamics simulations
- evolutionary optimization of network structure and weights (+ backpropagation)
- four possible activation functions: linear, relu, tanh, elu
- search space $\sim 10^{21}$ possible network structures

# Parsimonious Neural Networks (Desai and Strachan 2021)

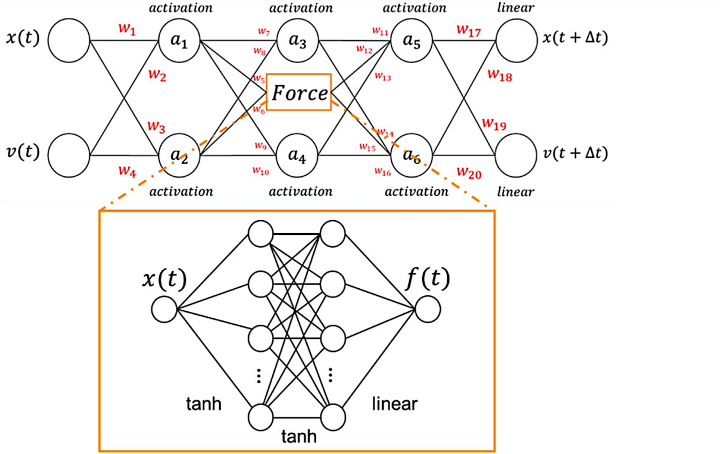
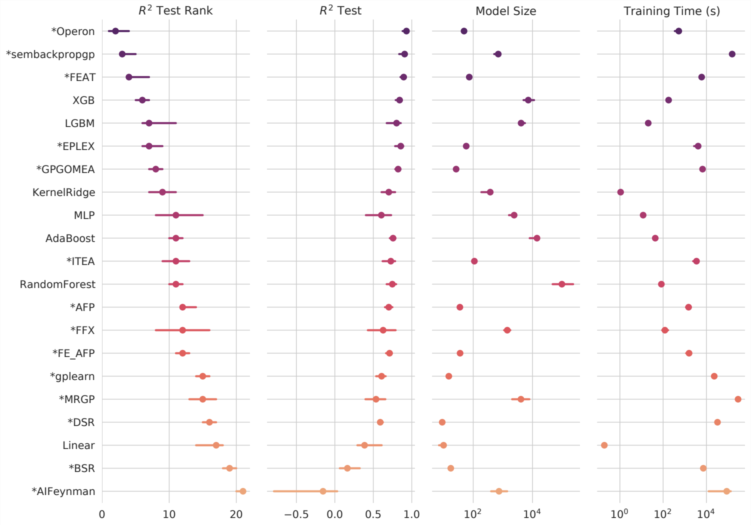



**Figure 1.** Neural network used as the starting point to find the parsimonious neural network as the network that explains the data in the simplest manner possible. The force sub-network is highlighted in orange and is fed into the neural network as a pre-trained model, whose weights are subsequently kept fixed throughout.

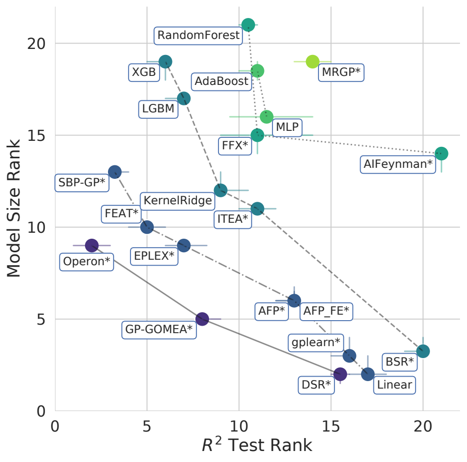
Desai and Strachan 2021 https://www.nature.com/articles/s41598-021-92278-w

# Methods Comparison




La Cava et al, NeurIPS Datasets and Benchmarks 2021, https://arxiv.org/abs/2107.14351

# Methods Comparison




La Cava et al, NeurIPS Datasets and Benchmarks 2021, https://arxiv.org/abs/2107.14351

**Operon** (Burlacu et al 2020)

- evolutionary system on top of Ceres (http://ceres-solver.org), Eigen (https://gitlab.com/libeigen/eigen), Taskflow (https://taskflow.github.io/)
- tree AST as postorder sequence
- Levenberg-Marquardt coefficient tuning

**SRBench**

https://github.com/cavalab/srbench

# Example: Shape Constraints

Shape constraints are general concept applicable to different forms of regression analysis.
(but not all methods equally amenable)

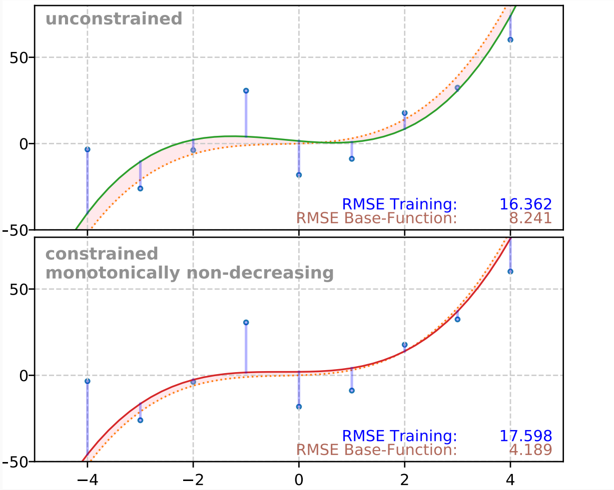
**Motivation: ensure conformance to physical expectation**

- Boundedness / Non-negativity / non-positivity (function image)
- Monotonicity (first derivative)
- Convexity / concavity (second derivative)

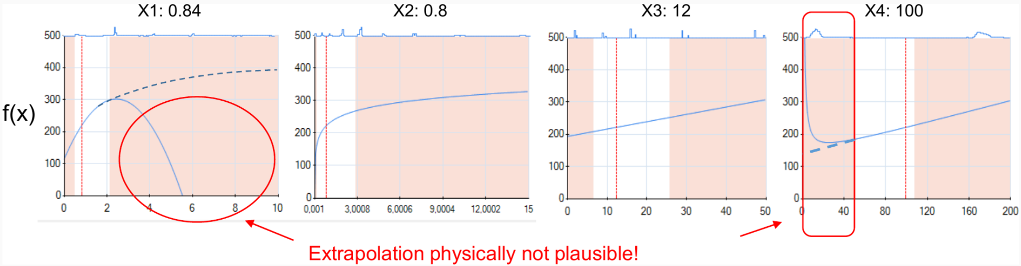
**Approach**

- interval arithmetic (pessimistic approach)
- sampling the input space (optimistic approach)
- hard vs. soft penalties
- single- vs. multi-objective

# Example: Shape Constraints



# Example: Shape Constraints



# Example: Shape Constraints

**Interval Arithmetic**

- method for calculating output ranges of mathematical expressions
- rules for computing output intervals of arithmetic and elementary operators
- intervals represented as $[a, b] = \{x \in \mathbb{R} | a \leqslant x \leqslant b\}$

**Weakness**

Does not track dependencies between operator arguments (“dependency problem”)

- $f(x) = x - x, x \in [a, b]$ – IA: $[a - b, b - a]$, actual: $[0, 0]$
- $f(x) = x^2 - x, x \in [-1, 1]$ – IA: $[-1, 2]$, actual: $[-\frac{1}{4}, 2]$

IA tends to overestimate the output range, actual conforming models might get discarded.

## Example: Shape Constraints

**Basic properties of Interval Arithmetic (cf. Warwick Tucker - Validated Numerics, 2011)**

- Interval addition and multiplication are commutative and associative
- IA is not distributive in general but sub-distributive
  $\forall a, b, c \in \mathbb{IR}, a(b + c) \subseteq ab + ac$
- Monotone functions
  If $f : X \to \mathbb{R}$ is non-decreasing, then $f(X) = [f(\underline{X}), f(\overline{X})]$
- Inclusion isotonicity
  $f : \mathbb{IR}^n \to \mathbb{IR}$ is *inclusion isotonic* if $\forall x, y \in \mathbb{IR}^n, x \subseteq y \implies f(x) \subseteq f(y)$
- Interval extension
  $f : \mathbb{IR}^n \to \mathbb{IR}$ is an *interval extension* of $g : \mathbb{R}^n \to \mathbb{R}$ if $\forall x \in \mathbb{R}^n, g(x) = g(x)$

**Fundamental theorem of Interval Analysis**

If $f : \mathbb{IR}^n \to \mathbb{IR}$ is inclusion isotonic and an interval extension then $g(x) \subseteq f(x), \forall x \in \mathbb{IR}^n$.

# Example: Shape Constraints for Symbolic Regression

**Basic properties of Interval Arithmetic**

Usually the input space of the model is limited for example to a $d$-dimensional box

$$\mathcal{S} = [\underline{x_1}, \overline{x_1}] \times [\underline{x_2}, \overline{x_2}] \times \ldots \times [\underline{x_d}, \overline{x_d}] \subseteq \mathbb{R}^d$$

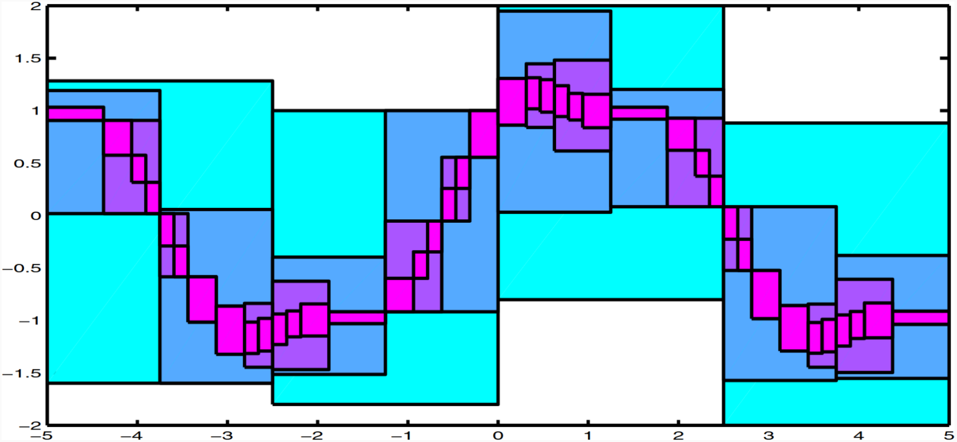
**IA** is both inclusion isotonic and an interval extension

Possible to improve its accuracy by discretizing $\mathcal{S}$ and computing the union of discretized results.

Greedy approaches, branch-and-bound heuristics for more accurate output.

# Example: Shape Constraints for Symbolic Regression

**Tightening the bounds for $f(x) = \cos^3(x) + \sin(x)$**




W. Tucker - Validated Numerics for Pedestrians, ©2005 European Mathematical Society

# Example: Shape Constraints

Physics textbook equations from the Feynman Symbolic Regression Database
(Udrescu and Tegmark 2020, https://doi.org/10.1126/sciadv.aay2631)

| Instance | Expression |
|---|---|
| I.6.20 | $\exp\left(\frac{-\left(\frac{\theta}{\sigma}\right)^2}{2}\right)\frac{1}{\sqrt{2\pi\sigma}}$ |
| I.9.18 | $\frac{G\,m1\,m2}{(x2-x1)^2+(y2-y1)^2+(z2-z1)^2}$ |
| I.30.5 | $\operatorname{asin}\left(\frac{lambd}{nd}\right)$ |
| I.32.17 | $\frac{1}{2}\epsilon c\,Ef^2\,\frac{8\pi r^2}{3}\,\frac{\omega^4}{(\omega^2-\omega_0{}^2)^2}$ |
| I.41.16 | $\frac{h\,\omega^3}{\pi^2\,c^2\left(\exp\left(\frac{h\omega}{kbT}\right)-1\right)}$ |
| I.48.20 | $\frac{m\,c^2}{\sqrt{1-\frac{v^2}{c^2}}}$ |
| II.35.21 | $n_{rho}\,mom\tanh\left(\frac{momB}{kbT}\right)$ |
| III.9.52 | $\frac{p_d Ef\,t}{h}\sin\left(\frac{(\omega-\omega_0)\,t}{2}\right)^2$ |
| III.10.19 | $mom\sqrt{Bx^2+By^2+Bz^2}$ |

| Instance | Input space | Constraints |
|---|---|---|
| I.6.20 | $(\sigma,\theta)\in[1..3]^2$ | $([0..\infty],0,-1)$ |
| I.9.18 | $(x1,y1,z1,m1,m2,G,x2,y2,z2)\in[3..4]^3\times[1..2]^6$ | $([0..\infty],-1,-1,-1,1,1,1,1,1,1)$ |
| I.30.5 | $(lambd,n,d)\in[1..5]^2\times[2..5]$ | $([0..\infty],1,-1,-1)$ |
| I.32.17 | $(\epsilon,c,Ef,r,\omega,\omega_0)\in[1..2]^5\times[3..5]$ | $([0..\infty],1,1,1,1,1,-1)$ |
| I.41.16 | $(\omega,T,h,kb,c)\in[1..5]^5$ | $([0..\infty],0,1,-1,1,-1)$ |
| I.48.20 | $(m,v,c)\in[1..5]\times[1..2]\times[3..20]$ | $([0..\infty],1,1,1)$ |
| II.35.21 | $(n_{rho},mom,B,kb,T)\in[1..5]^5$ | $([0..\infty],1,1,1,-1,-1)$ |
| III.9.52 | $(p_d,Ef,t,h,\omega,\omega_0)\in[1..3]^4\times[1..5]^2$ | $([0..\infty],1,1,0,-1,0,0)$ |
| III.10.19 | $(mom,Bx,By,Bz)\in[1..5]^4$ | $([0..\infty],1,1,1,1)$ |

# Example: Shape Constraints

**Recent work by us (idea much older!)**

- G. Kronberger, C. Haider, M. Kommenda, B. Burlacu (FH OÖ / JRZ)
- F. O. de Franca (CMCC, HAL, Federal Univ. of ABC, Santo Andre, Brazil)
- **Kronberger et al.** – *Shape-constrained Symbolic Regression – Improving Extrapolation with Prior Knowledge*, Evolutionary Computation, Oct '21 https://arxiv.org/abs/2103.15624v1

**Methods comparison**

- test setting included varying levels of Gaussian noise over the training data
- comparison between evolutionary metaheuristics, polynomial regression, auto-sklearn (Feurer et al. 2015) with/without shape constraints

**Results**

- it's complicated! :)
- shape constraints ensure conforming models, but not necessarily better generalization (test error)
- however, shape-constrained models perform better on noisy data

# Example: Atomic Potentials (Materials Science)

**Numerical simulations** needed to investigate new materials and their properties.

**Molecular Dynamics (MD) and Monte Carlo simulations**

- Need a realistic model of inter-atomic interactions, represented usually by interaction potentials.
- Only realistic interaction models are able to produce quantitatively reliable outputs

**Modeling interatomic potentials**

- Empirical / classical approach: very fast, simple analytical formulas, not very accurate
- Quantum: density functional theory (DFT) and others, very accurate, very slow $O(N^3)...O(N^4)$
- Machine learning: use data generated by quantum methods, generate surrogate potentials

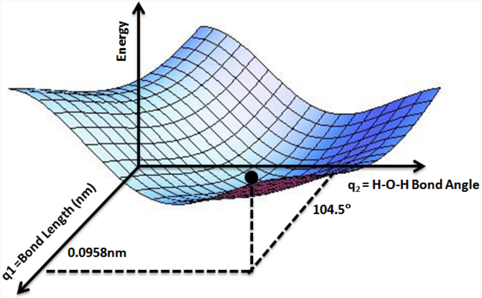
# Potential Energy Surfaces (PES)

Play a pivotal role in (large-scale) particle simulations

Describe the relation between atomic positions and their potential energy

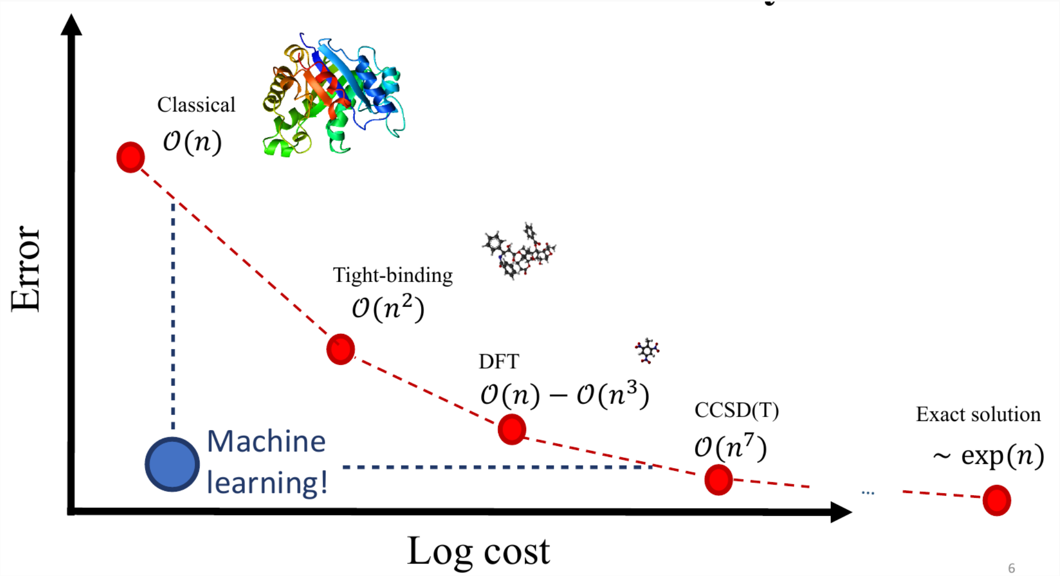
Very computationally expensive

ML methods recently emerged as a means to model PES at lower computational cost




**source**: https://en.wikipedia.org/wiki/Potential_energy_surface

# Potential Energy Surfaces (PES)



# Machine-learning Potentials

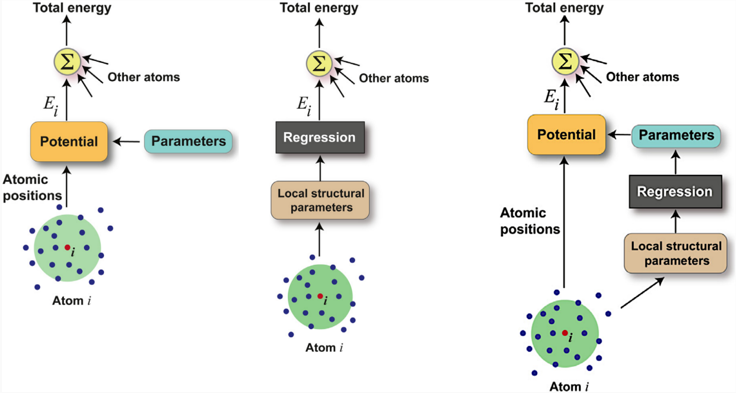
Learn the relationship between chemical structure and potential energy.

**General requirements**

- general applicability and absence of ad-hoc approximations (potentials that are transferable)
- accuracy comparable to first-principles methods (including high-order many-body effects)
- very high efficiency to enable large simulations
- the ability to describe chemical reactions and arbitrary atomic configurations
- the ability to be automatically constructed and systematically improved
- **the ability to provide physical insight**

**Currently available potentials are far from satisfying all the needs**

# Machine-learning Potentials



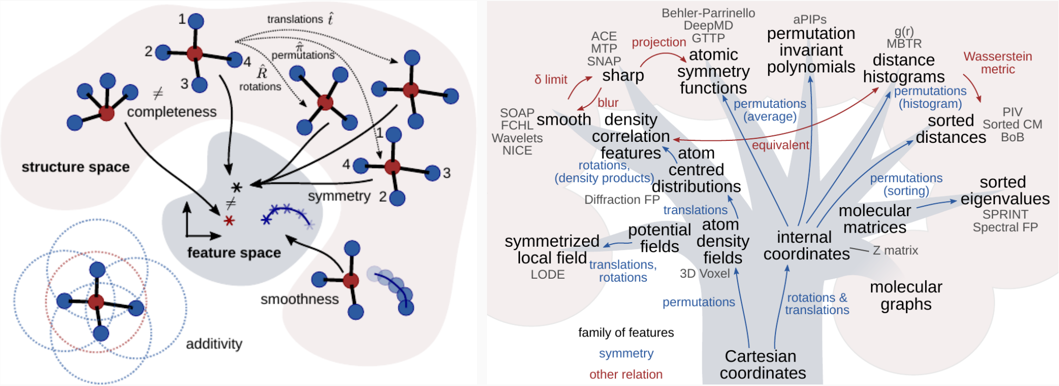

Mishin 2021, https://doi.org/10.1016/j.actamat.2021.116980

**Traditional** (left): based on physical understanding, simple functional form, highly approximate

**ML-based** (middle): regression based on structural parameters that capture the 3N-D configuration space

**Physics-informed ML** (right): local structural parameters mapped to parameters of a physics-based PES

## Capturing and mapping structural parameters = big challenge




Musil et. al, *Physics-Inspired Structural Representations for Molecules and Materials*, Chem Rev. 2021 Aug 25, https://doi.org/10.1021/acs.chemrev.1c00021

# Machine-learning Potentials

## Many approaches

- neural networks (most popular)
- polynomial fitting
- moment tensor potentials (linear combination of polynomial basis functions)
- Gaussian processes
- spectral neighbour analysis
- support vector machines
- interpolating moving least squares
- symbolic regression

## Ongoing issues

- Generality and transferability (often too system-specific)
- Conceptual problems related to incorporating rotational, translational and permutational invariance
- Model complexity and explainability
- Training data requirements

**"Black-box" methods**

- almost zero capability of including physical information into the functional forms
- must exclusively rely on the physics-inspired features considered in atomic descriptors
- increased mathematical and computational complexity of resulting interaction models
- requires large amounts of training data

**"White-box" methods**

- emerging as an alternative for ML-based potentials (https://doi.org/10.1063/1.5126336)
- provides researchers with analytic equations, which expectably would have better interpretability
- more amenable to the integration of physical knowledge
- not requiring a priori knowledge or predefined functional forms
- explicit representation that can be interpreted by domain experts
- smaller number of parameters and better efficiency

# Symbolic Regression Potentials

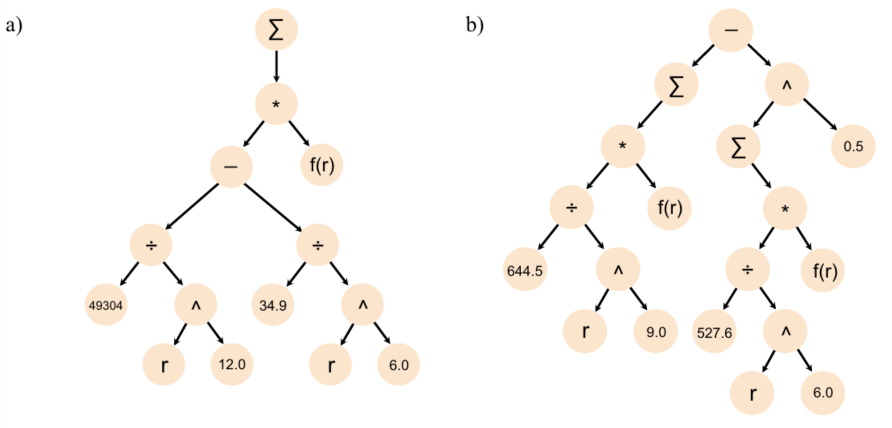
## Training data

- training data available from DFT simulations
- data consists of simulation snapshots with atomic positions and energy
- usually in special formats (VASP, POSCAR)
- each “data row” is a list of coordinates describing the atomic clusters and an energy value

## Symbolic regression approach

- pairwise atomic distances computed from Cartesian coordinates
- distances potentially filtered by *inner* and *outer* radius parameters
- functional transformation of the input (distances, forces) followed by $\sum$

# Symbolic Regression Potentials




Hernandez et al., *Fast, accurate, and transferable many-body interatomic potentials by symbolic regression*, 2019, https://doi.org/10.1038/s41524-019-0249-1

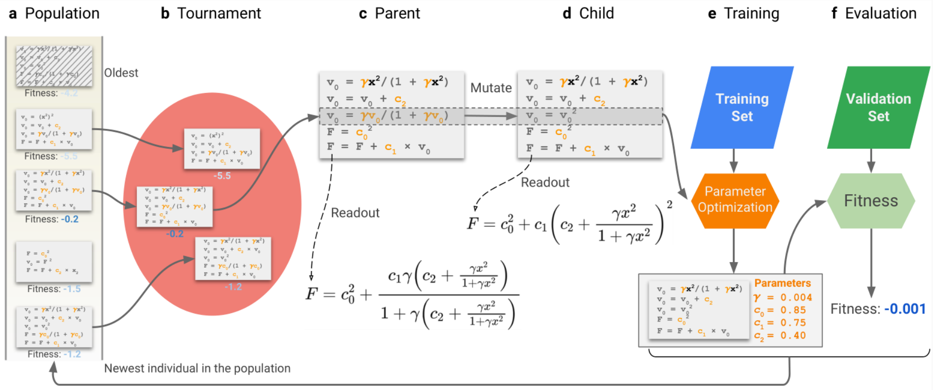
https://gitlab.com/muellergroup/poet

# Symbolic Functional Evolutionary Search (Ma et al. 2022)

Evolving from existing functional form ($\omega$B97M $-$ V)

Automatically constructs accurate functionals in the symbolic form

New found functional GAS22 (Google Accelerated Science 22) performs better on main-group chemistry




Ma et el., *Evolving Symbolic Density Functionals*, 2022, https://doi.org/10.48550/arXiv.2203.02540

# Conclusion

**Machine learning, critical part of the 4th paradigm of science.**

Scientific-ML is an emerging field focusing on applications in computational science.

Explainability and interpretability become increasingly important properties of ML models.

Existing methods do not quite meet the needs of scientific users.

Models should respect or incorporate physical laws.

Several hybrid approaches are emerging with the goal of incorporating physical knowledge.

Often, ML methods require non-trivial, task-specific modications

SciML methods typically need to run in performance-critical environments.