AI in the Sciences and Engineering

Symbolic Regression and Model Discovery

Spring Semester 2024

Siddhartha Mishra
Ben Moseley
ETH zürich

Course timeline

Tutorials		Lectures			
Mon 12:15-14:00 HG E 5		*Wed 08:15-10:00 ML H 44*		*Fri 12:15-13:00 ML H 44*	
19.02.		21.02.	Course introduction	23.02.	Introduction to deep learning I
26.02.	Introduction to PyTorch	28.02.	Introduction to deep learning II	01.03.	Introduction to PDEs
04.03.	Simple DNNs in PyTorch	06.03.	Physics-informed neural networks – introduction	08.03.	Physics-informed neural networks - limitations
11.03.	Implementing PINNs I	13.03.	Physics-informed neural networks – extensions	15.03.	Physics-informed neural networks – theory I
18.03.	Implementing PINNs II	20.03.	Physics-informed neural networks – theory II	22.03.	Supervised learning for PDEs I
25.03.	Operator learning I	27.03.	Supervised learning for PDEs II	29.03.	
01.04.		03.04.		05.04.	
08.04.	Operator learning II	10.04.	Introduction to operator learning I	12.04.	Introduction to operator learning II
15.04.		17.04.	Convolutional neural operators	19.04.	Time-dependent neural operators
22.04.	GNNs	24.04.	Large-scale neural operators	26.04.	Attention as a neural operator
29.04.	Transformers	01.05.		03.05.	Windowed attention and scaling laws
06.05.	Diffusion models	08.05.	Introduction to hybrid workflows I	10.05.	Introduction to hybrid workflows II
13.05.	Coding autodiff from scratch	15.05.	Neural differential equations	17.05.	Diffusion models
20.05.		22.05.	**Introduction to JAX / symbolic regression**	24.05.	**Symbolic regression and model discovery**
27.05.	Intro to JAX / Neural ODEs	29.05.	Guest lecture: AlphaFold	31.05.	Guest lecture: AlphaFold

Lecture overview

- What is model discovery?
- Challenges of symbolic regression
- Function discovery
    - AI Feynman
    - Genetic algorithms
- Model discovery
    - SINDy
    - Other approaches

Lecture overview

- What is model discovery?
- Challenges of symbolic regression
- Function discovery
  - AI Feynman
  - Genetic algorithms
- Model discovery
  - SINDy
  - Other approaches

Learning objectives

- Understand how symbolic regression (SR) algorithms are designed
- Understand how SR is used for function and model discovery

Discovering physics

$$R_{\mu\nu} - \frac{1}{2}Rg_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4}T_{\mu\nu}$$

Discovering physics

$$R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

Curvature of space-time

Stress-energy-momentum content of space-time

$R_{\mu\nu}$ = Ricci curvature tensor
R = scalar curvature
$g_{\mu\nu}$ = metric tensor
Λ = cosmological constant
G = gravitational constant
c = speed of light in vacuum
$T_{\mu\nu}$ = stress-energy tensor

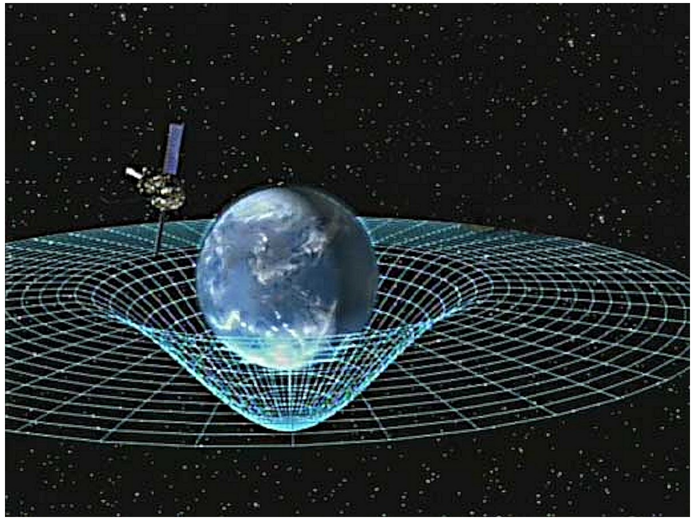


Image source: NASA

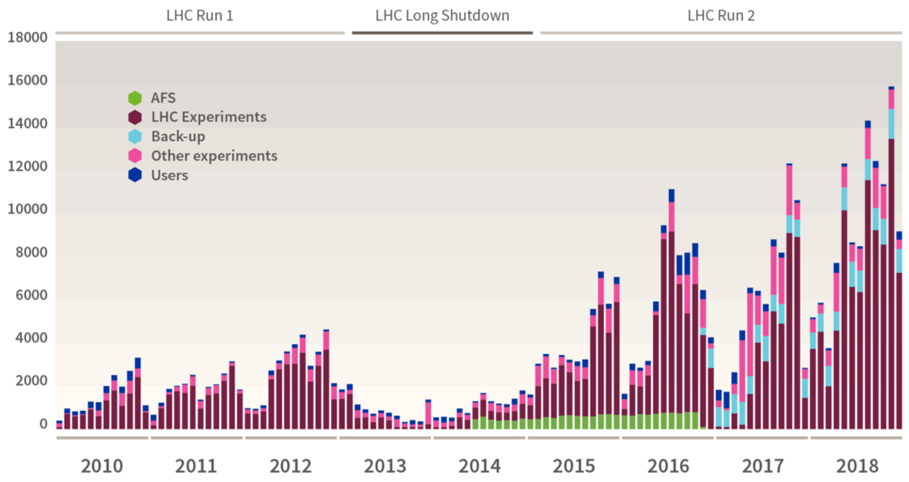
Discovering physics

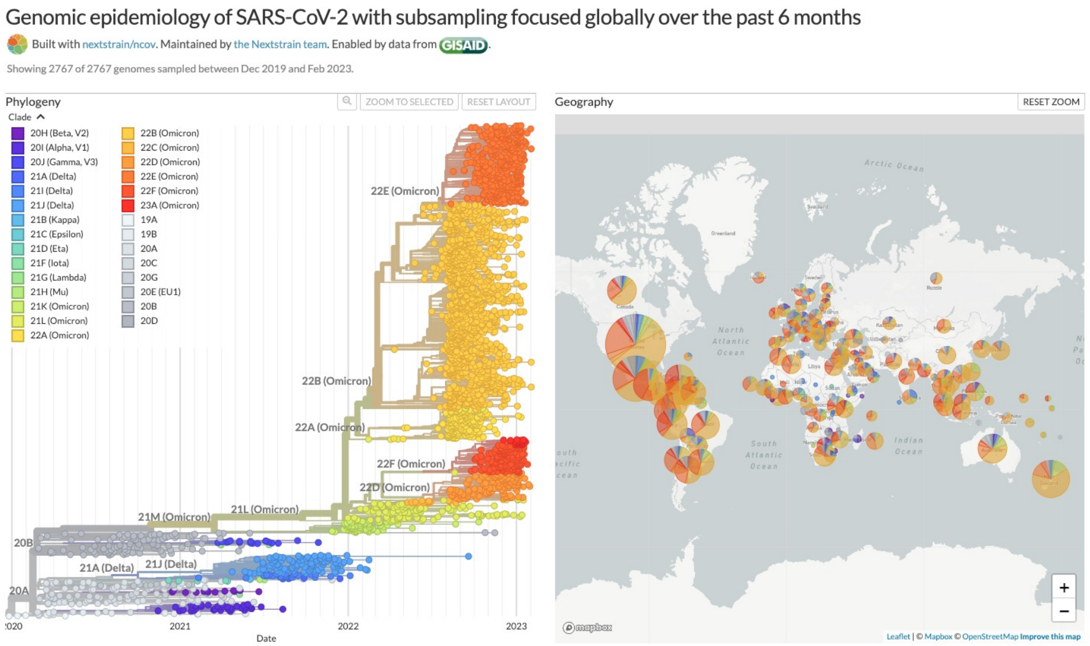
What if AI could discover the laws of physics?

Discovering physics

What if AI could discover the laws of physics?

Data (in terabytes) recorded on tapes at CERN month-by-month (2010–2018) (Source: CERN)




Genomic epidemiology of SARS-CoV-2 with subsampling focused globally over the past 6 months

Built with nextstrain/ncov. Maintained by the Nextstrain team. Enabled by data from GISAID.

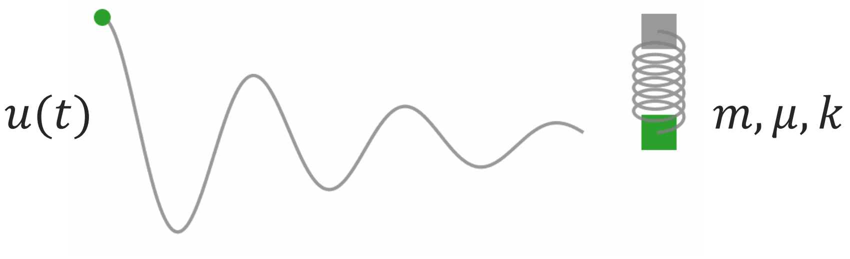
Showing 2767 of 2767 genomes sampled between Dec 2019 and Feb 2023.

Source: Nextstrain

Model discovery

Task:

Given **observations** of a physical system



Find an underlying **model**

$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$

Function discovery

Task:

Given **observations** of some **function** $f(\boldsymbol{x})$,

$$D = \{(\boldsymbol{x}_1, f_1), \ldots, (\boldsymbol{x}_N, f_N)\}$$

Find its **mathematical expression** (= **symbolic regression**)

$$f = \frac{1}{e^{\frac{E-\mu}{k_B T}} + 1}$$

$$PV = nRT$$

$$F = k\frac{q_1 q_2}{r^2}$$

$$E = h\nu$$

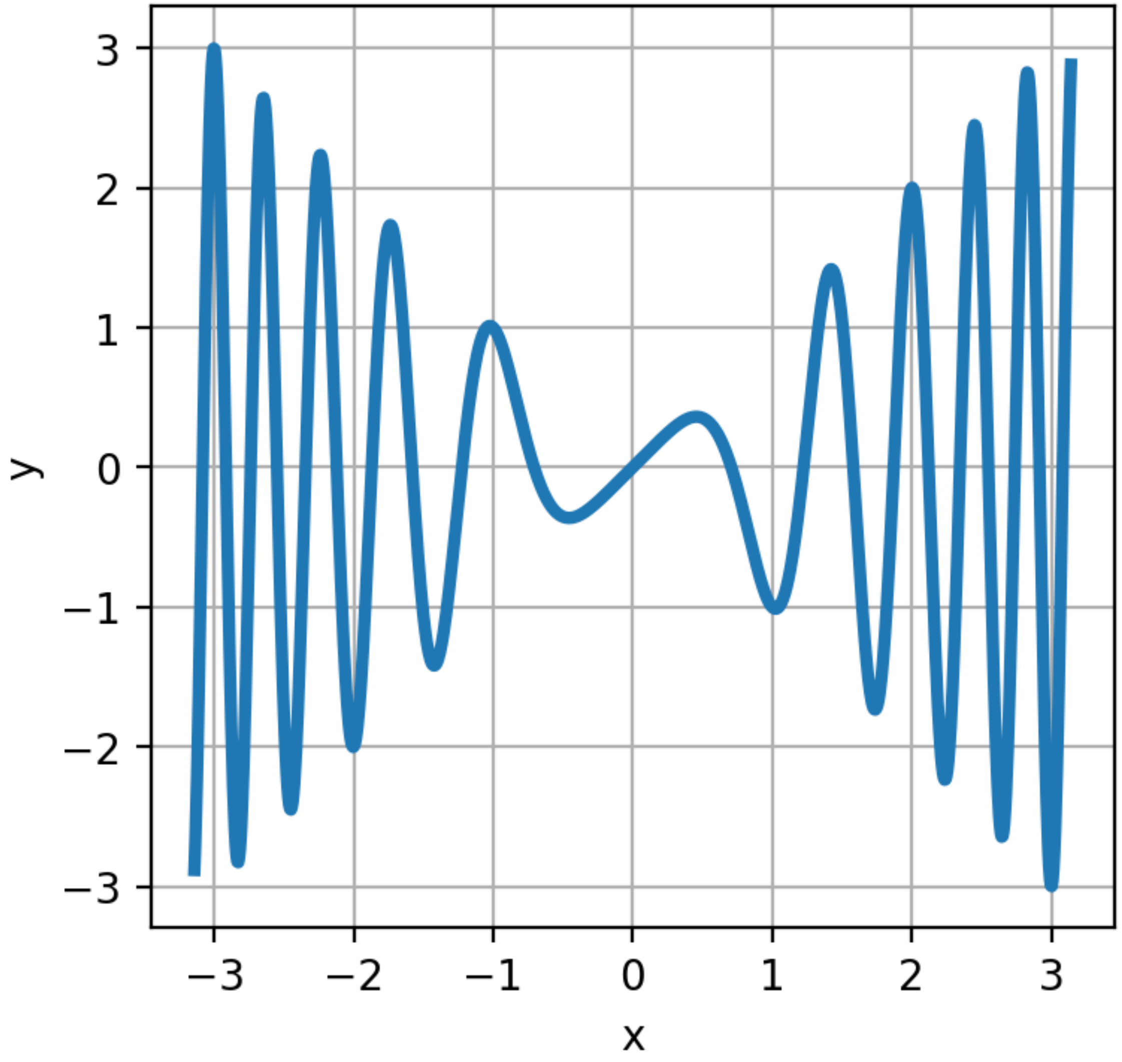
$$P = \sigma A T^4$$

$$E = \frac{mc^2}{\sqrt{1 - v^2/c^2}}$$

$$V = IR$$

$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$

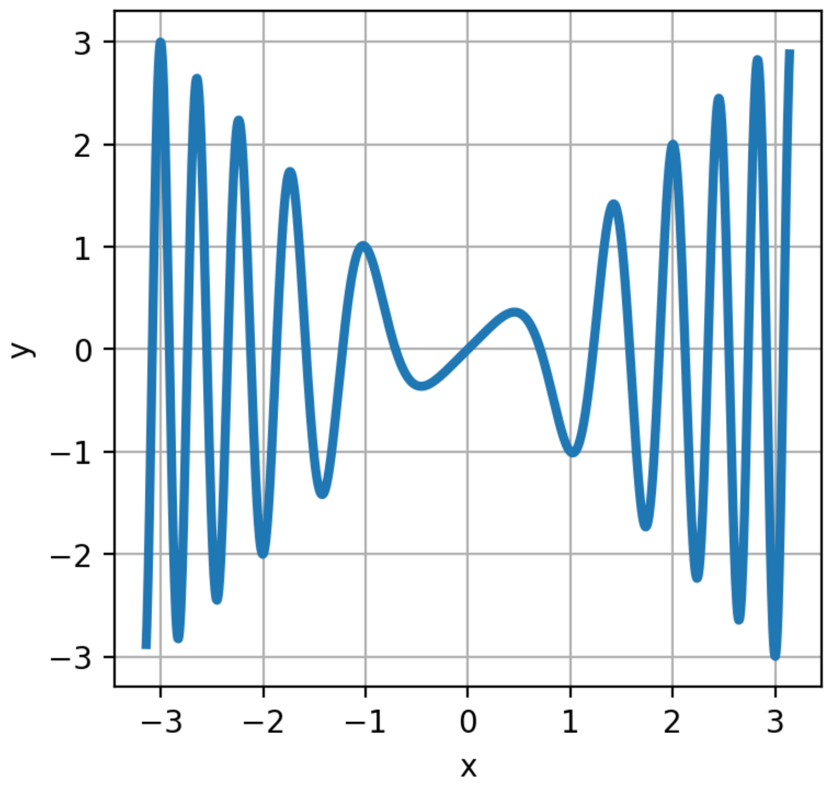
Challenge: guess the function



$f(x) = ?$

Challenge: guess the function

$y = x\cos(\pi x^2)$



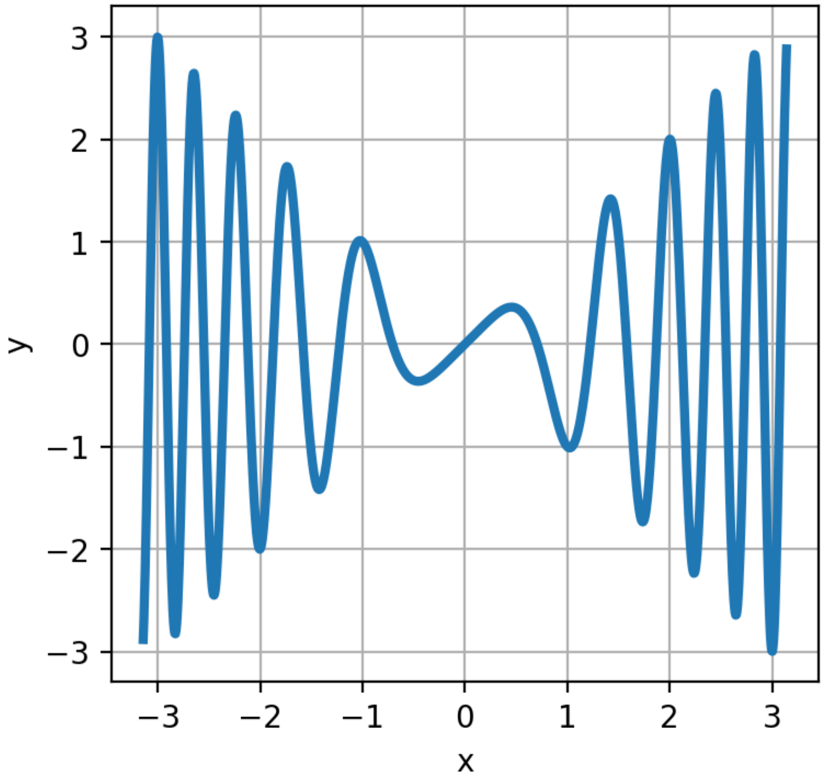
How I might guess this function:

1. It’s oscillatory
2. Frequency increases as x increases
3. Amplitude grows linearly
4. Use location of peaks and troughs to derive coefficients

$\Rightarrow y = x\cos(\pi x^2)$

Symbolic regression vs function fitting

$y = x\cos(\pi x^2)$



How I might guess this function:

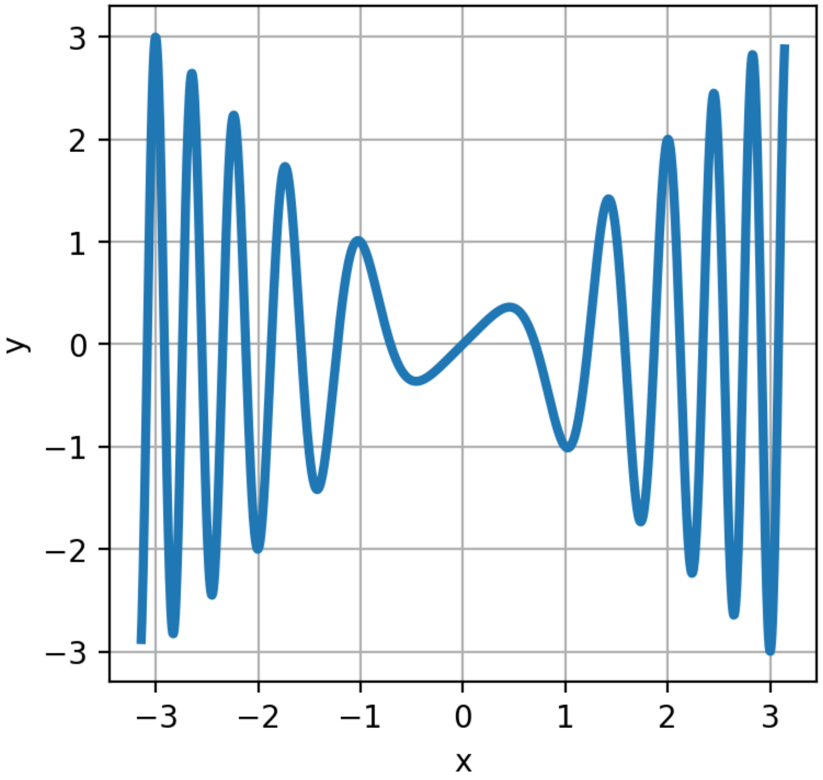
1. It's oscillatory
2. Frequency increases as x increases
3. Amplitude grows linearly
4. Use location of peaks and troughs to derive coefficients
   $\Rightarrow y = x\cos(\pi x^2)$

How a neural network would fit this function:

1. Assume the function has some prior form, e.g. $y = \boldsymbol{w}_2\sigma(\boldsymbol{w}_1 x + \boldsymbol{b}_1) + b_2$
2. Find coefficients which best fit data

Symbolic regression vs function fitting

$y = x\cos(\pi x^2)$



How I might guess this function:

1. It's oscillatory
2. Frequency increases as x increases
3. Amplitude grows linearly
4. Use location of peaks and troughs to derive coefficients
   $\Rightarrow y = x\cos(\pi x^2)$

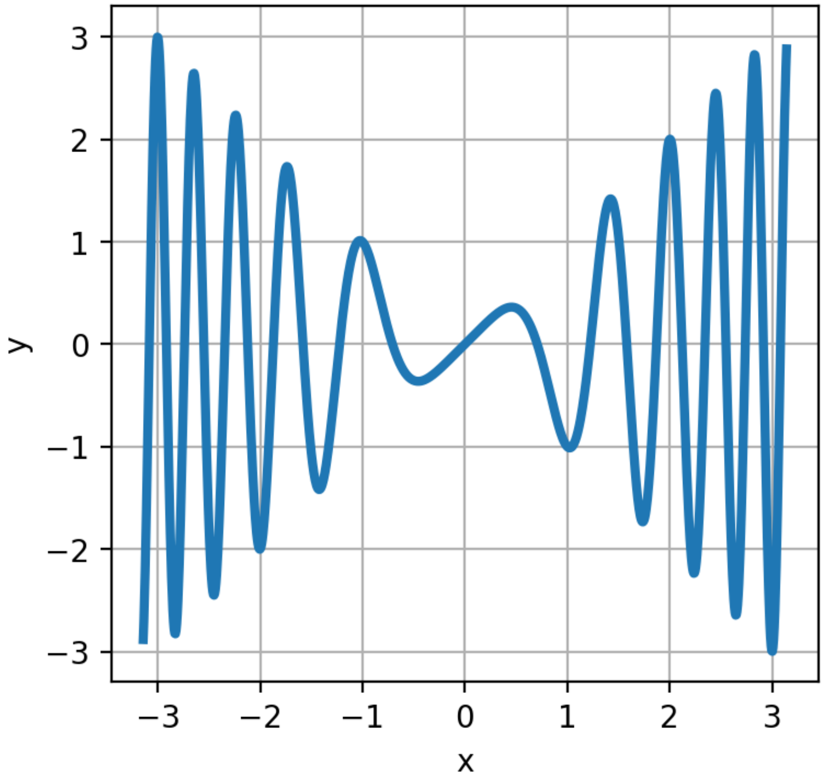
How a neural network would fit this function:

1. Assume the function has some prior form, e.g. $y = \boldsymbol{w}_2\sigma(\boldsymbol{w}_1 x + \boldsymbol{b}_1) + b_2$
2. Find coefficients which best fit data

Q: why is SR often harder than function fitting?

Challenges of symbolic regression

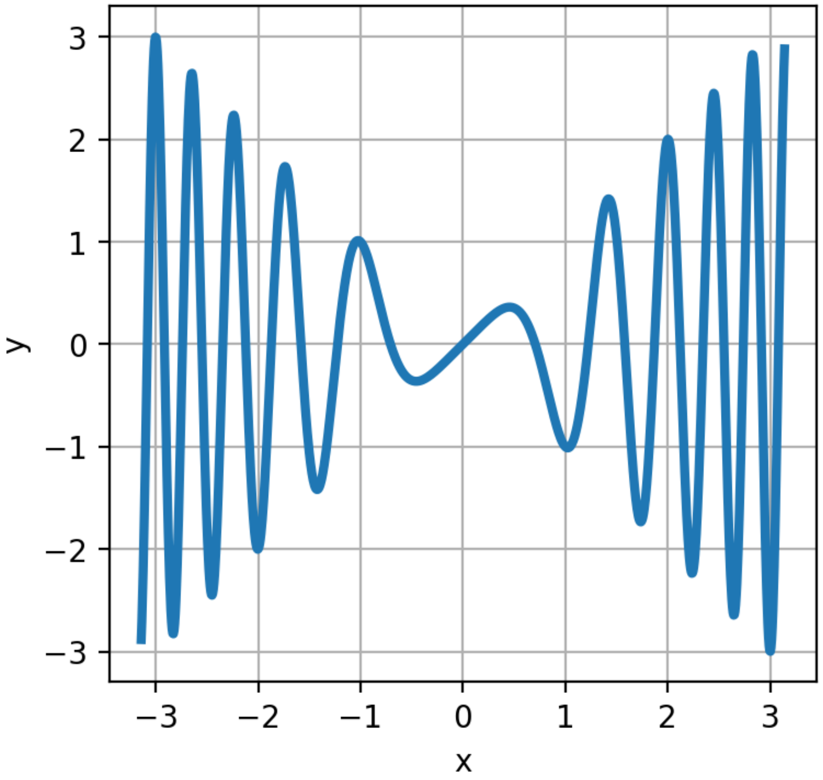
$y = x\cos(\pi x^2)$



Q: why is SR often harder than function fitting?

Challenges of symbolic regression

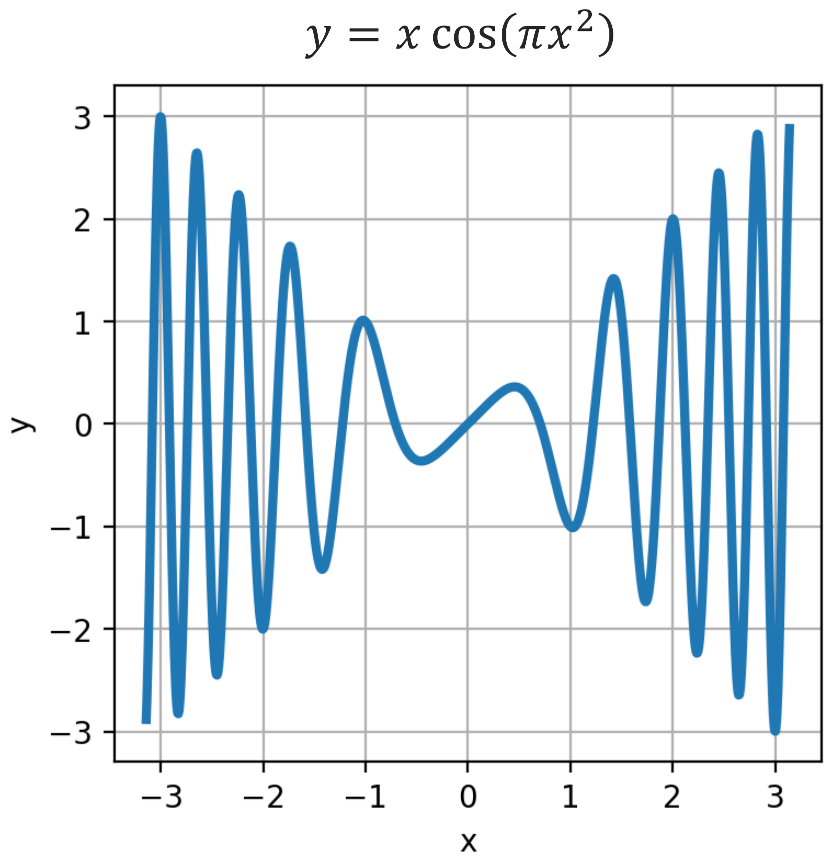
$y = x\cos(\pi x^2)$



Q: why is SR often harder than function fitting?

- Need to learn **entire expression**, not just coefficients, and we may not know its **length**

Challenges of symbolic regression

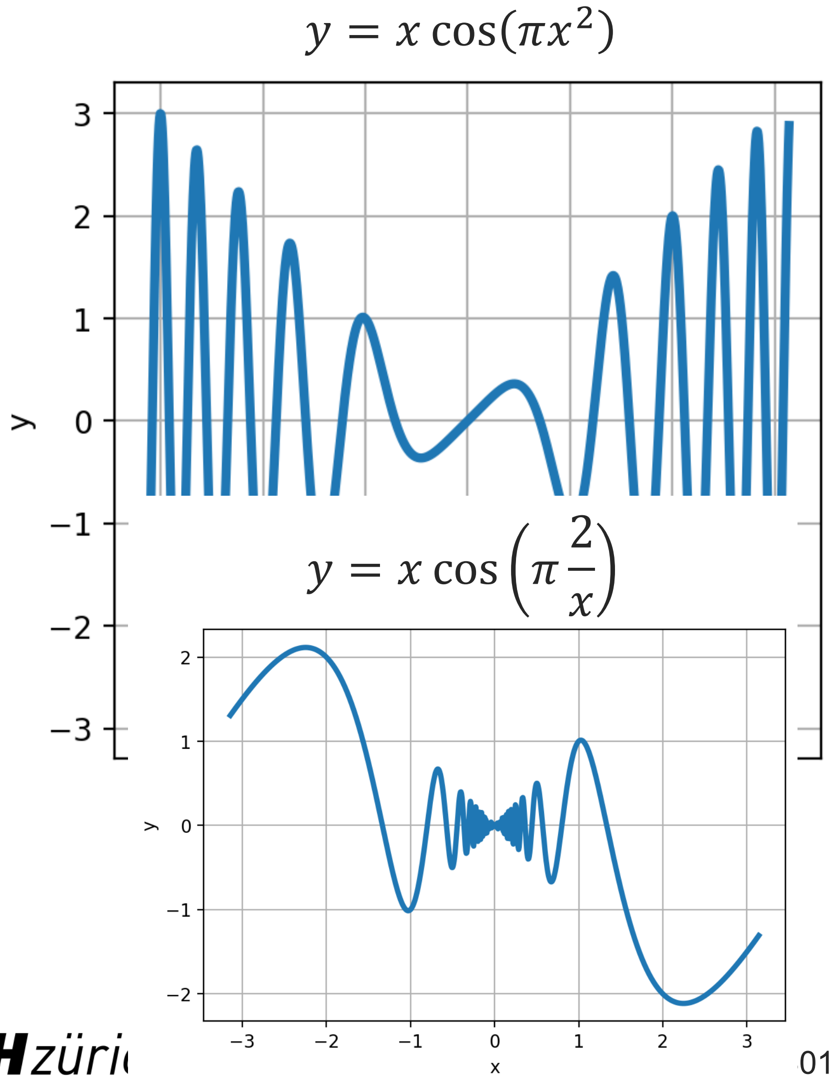


Q: why is SR often harder than function fitting?

- Need to learn **entire expression**, not just coefficients, and we may not know its **length**
- The search space is **exponential**
  - There are s^n strings of length n for a library of s “elementary operators” (+, -, /, *, sin, cos, ...)

⚠

Challenges of symbolic regression



$y = x\cos(\pi x^2)$

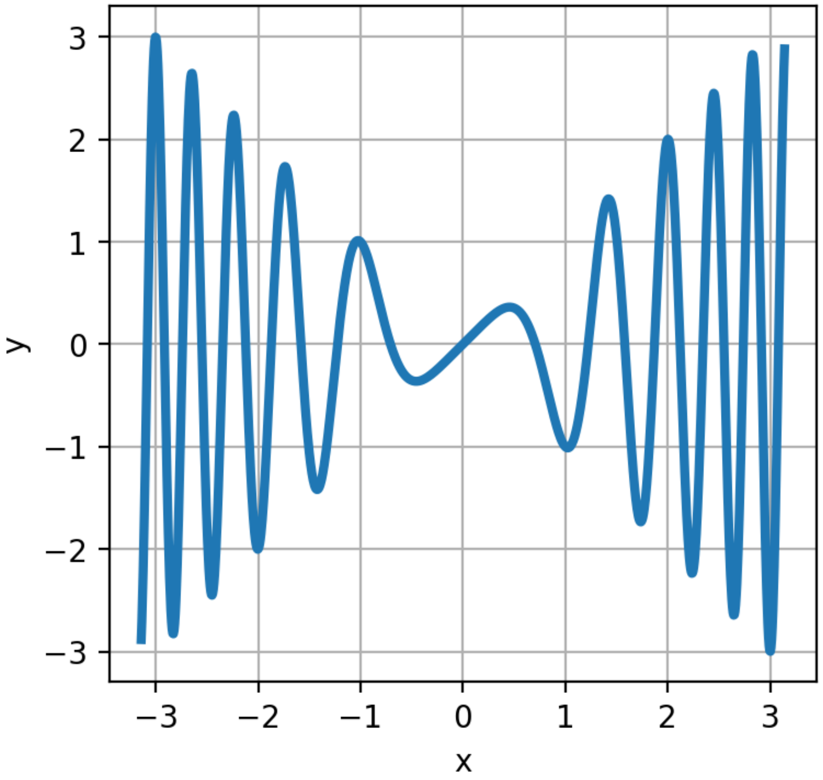
$y = x\cos\left(\pi \frac{2}{x}\right)$

Q: why is SR often harder than function fitting?

- Need to learn **entire expression**, not just coefficients, and we may not know its **length**
- The search space is **exponential**
  - There are s^n strings of length n for a library of s “elementary operators” (+, -, /, *, sin, cos, ...)
- There is typically **not** a **smooth** interpolation between different expressions (= lack of **differentiability**)

Challenges of symbolic regression

$y = x\cos(\pi x^2)$



Q: why is SR often harder than function fitting?

- Need to learn **entire expression**, not just coefficients, and we may not know its **length**

- The search space is **exponential**
  - There are s^n strings of length n for a library of s "elementary operators" (+, -, /, *, sin, cos, ...)

- There is typically **not** a **smooth** interpolation between different expressions (= lack of **differentiability**)

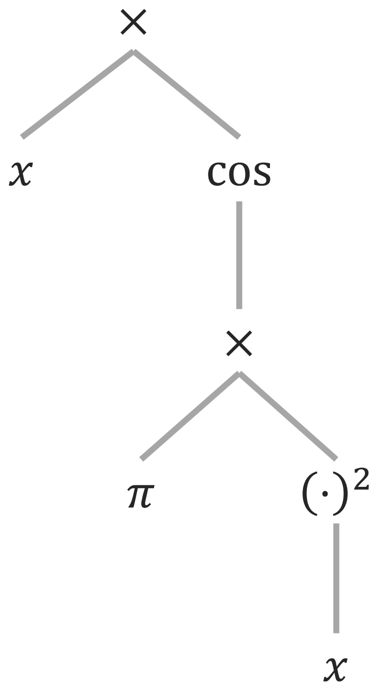
- With only a finite number of observations (N), there may be **many valid** expressions (ill-posed)

⚠

Mathematical expressions as trees

$$y = x\cos(\pi x^2)$$

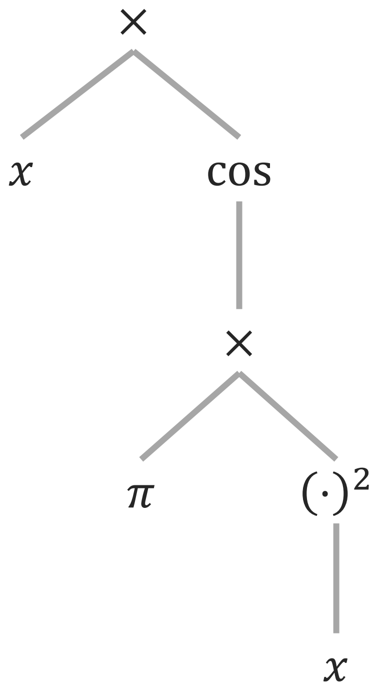
Image credits: Google



Mathematical expressions as trees

$$y = x\cos(\pi x^2)$$

Image credits: Google



Root: expression

Nodes: operations

Branches: either unary or binary

Leaves: reals

Tree depth: 4

Search space

Tree depth: 2, Library: $\{+, \times, {}^\wedge 2, \cos, \sin\}$

$\cos(\cos(x))$

$\sin(\cos(x))$

$\cos^2(x)$

$\cos(x) + \cos(x)$

$\cos(x)\cos(x)$

$\cos(x) + \sin(x)$

$\cos(x)\sin(x)$

$\cos(x) + x^2$

$\cos(x)\, x^2$

$\cos(x) + x + x$

$\cos(x)x + x$

$\cos(x) + xx$

$\cos(x)xx$

$\cos(\sin(x))$

$\sin(\sin(x))$

$\sin^2(x)$

$\sin(x) + \cos(x)$

$\sin(x)\cos(x)$

$\sin(x) + \sin(x)$

$\sin(x)\sin(x)$

$\sin(x) + x^2$

$\sin(x)\, x^2$

$\sin(x) + x + x$

$\sin(x)x + x$

$\sin(x) + xx$

$\sin(x)xx$

$\cos(x^2)$

$\sin(x^2)$

x^{2^2}

$x^2 + \cos(x)$

$x^2\cos(x)$

$x^2 + \sin(x)$

$x^2\sin(x)$

$x^2 + x^2$

x^2x^2

$x^2 + x + x$

$x^2x + x$

$x^2 + xx$

x^2xx

$\cos(x + x)$

$\sin(x + x)$

$x + x^2$

$x + x + \cos(x)$

$x + x\cos(x)$

$x + x + \sin(x)$

$x + x\sin(x)$

$x + x + x^2$

$x + xx^2$

$x + x + x + x$

$x + xx + x$

$x + x + xx$

$x + xxx$

$\cos(xx)$

$\sin(xx)$

xx^2

$xx + \cos(x)$

$xx\cos(x)$

$xx + \sin(x)$

$xx\sin(x)$

$xx + x^2$

xxx^2

$xx + x + x$

$xxx + x$

$xx + xx$

$xxxx$

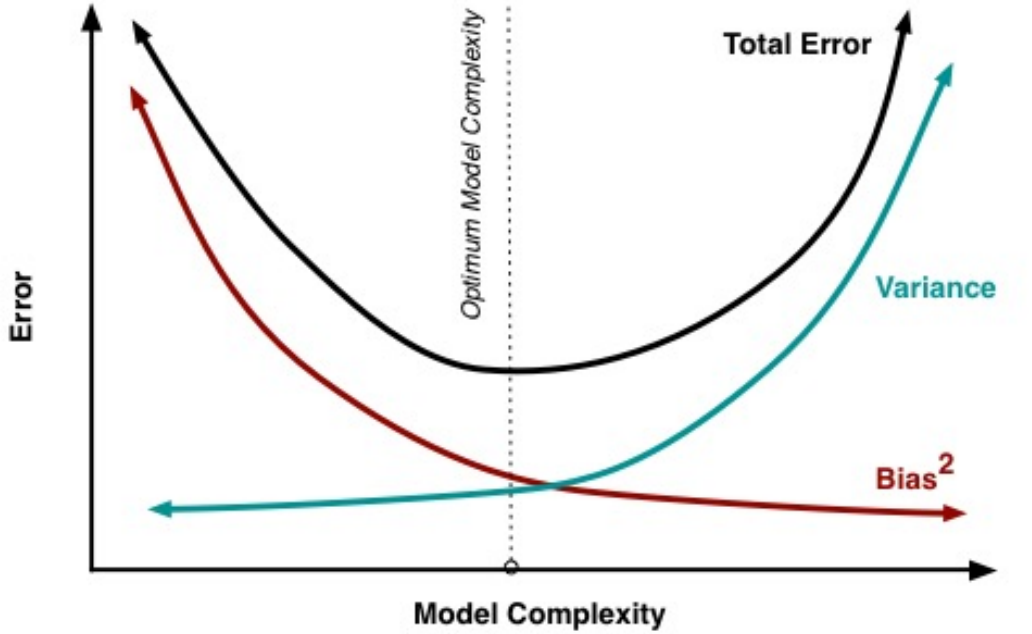
65 expressions

Pruning

Image credits: Seattle Department of Construction and Inspections

Occam’s razor

The simplest explanation is usually the best one



Source: http://scott.fortmann-roe.com/docs/BiasVariance.html

1 Normally a lot of stars or at least the moon are visible in the night sky... ***Where are the stars*** or the moon in this image?!

2 The moon has no atmosphere, hence there is no wind. ***Why is the flag not hanging down***?

3 Notice the ***used plastic water bottle*** on the soil. Something is not right here: Americans don't drink water

4 In the background, one can clearly see Stanley Kubrick, ***without space suit***. On the moon, he would die immediately due to the lack of oxygen!

5 ***Suspicious laugh***

https://9gag.com/gag/5163763

Requirements

To successfully solve a symbolic regression problem, we need:

1. An **assumption** (prior) on the structure of the expression

2. A **search** algorithm

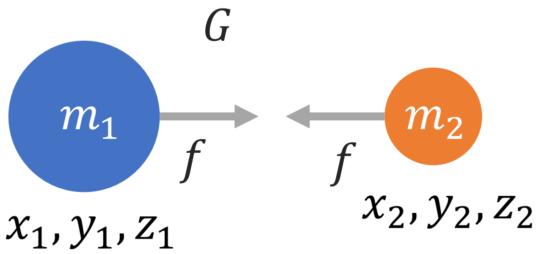
… there's a lot of innovation in both areas!

See e.g. here for state-of-the-art reviews:
Makke & Chawla, Interpretable scientific discovery with symbolic regression: a review, AI Review (2024)
Landajuela et al, A Unified Framework for Deep Symbolic Regression, NeurIPS (2022)

AI Feynman

Idea: look for “hidden simplicities” in the expression

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



What simplicities does this function have?

Udrescu and Tegmark, AI Feynman: A physics-inspired method for symbolic regression. Science Advances (2020)
Udrescu et al, AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity. NeurIPS (2020)

AI Feynman

$$\overset{\text{N}}{f}(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{\overset{\text{Nm}^2/\text{kg}^2}{G}\,\overset{\text{kg}}{m_1 m_2}}{\underset{\text{m}}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

1. **Units** must match!

AI Feynman

$$\overset{\text{N}}{f}(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{\overset{\text{Nm}^2/\text{kg}^2}{G}\,\overset{\text{kg}}{m_1 m_2}}{\underset{\text{m}}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

1. **Units** must match!

⇒ f can transformed into a **dimensionless** function, g

$$f = \frac{Gm_1^2}{x_1^2} \frac{\frac{m_2}{m_1}}{\left(\frac{x_2}{x_1} - 1\right)^2 + \left(\frac{y_2}{x_1} - \frac{y_1}{x_1}\right)^2 + \left(\frac{z_2}{x_1} - \frac{z_1}{x_1}\right)^2}$$

$$\equiv \frac{Gm_1^2}{x_1^2} \frac{a}{(b-1)^2 + (c-d)^2 + (e-\mathrm{f})^2} \equiv Cg(a, b, c, d, e, \mathrm{f})$$

AI Feynman

$$\overset{\text{N}}{f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2)} = \frac{\overset{\text{Nm}^2/\text{kg}^2}{G}\,\overset{\text{kg}}{m_1 m_2}}{\underset{\text{m}}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

1. **Units** must match!

⇒ f can transformed into a **dimensionless** function, g

$$f = \frac{Gm_1^2}{x_1^2} \frac{\frac{m_2}{m_1}}{\left(\frac{x_2}{x_1} - 1\right)^2 + \left(\frac{y_2}{x_1} - \frac{y_1}{x_1}\right)^2 + \left(\frac{z_2}{x_1} - \frac{z_1}{x_1}\right)^2}$$

$$\equiv \frac{Gm_1^2}{x_1^2} \frac{a}{(b-1)^2 + (c-d)^2 + (e-\mathrm{f})^2} \equiv Cg(a, b, c, d, e, \mathrm{f})$$

What does this do to the search space?

AI Feynman

$$\overset{\text{N}}{f}(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{\overset{\text{Nm}^2/\text{kg}^2}{G}\, m_1 \overset{\text{kg}}{m_2}}{\underset{\text{m}}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

1. **Units** must match!

$\Rightarrow f$ can transformed into a **dimensionless** function, g

$$f = \frac{Gm_1^2}{x_1^2} \frac{\frac{m_2}{m_1}}{\left(\frac{x_2}{x_1} - 1\right)^2 + \left(\frac{y_2}{x_1} - \frac{y_1}{x_1}\right)^2 + \left(\frac{z_2}{x_1} - \frac{z_1}{x_1}\right)^2}$$

$$\equiv \frac{Gm_1^2}{x_1^2} \frac{a}{(b-1)^2 + (c-d)^2 + (e-\mathrm{f})^2} \equiv Cg\boxed{(a, b, c, d, e, \mathrm{f})}$$

What does this do to the search space? => Reduces the number of variables

AI Feynman

$$\overset{\text{N}}{f}(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{\overset{\text{Nm}^2/\text{kg}^2}{G}\,\overset{\text{kg}}{m_1} m_2}{\underset{\text{m}}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}}$$

1. **Units** must match!

$$f = Cg(a, b, c, d, e, \mathrm{f})$$

See the paper for how C and the dimensionless variables can be determined (given only the units of f and its independent variables)

Udrescu and Tegmark, AI Feynman: A physics-inspired method for symbolic regression. Science Advances (2020)

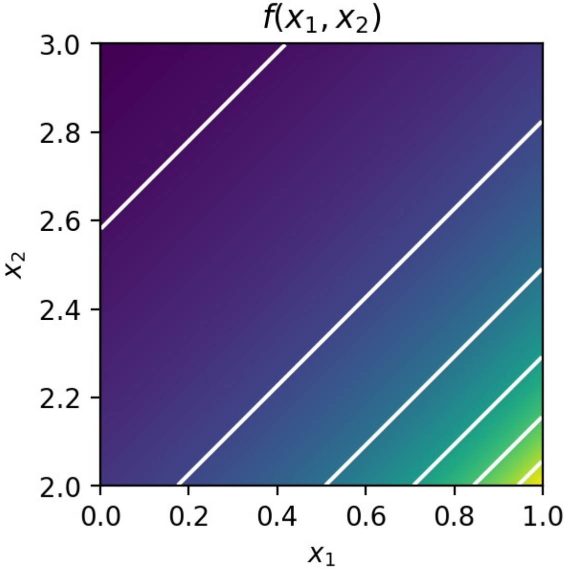
AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{Gm_1m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

2. Translational **symmetry**

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



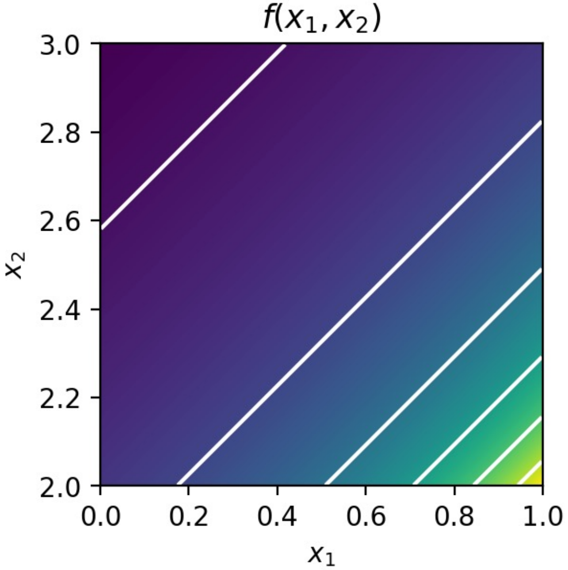
2. Translational **symmetry**

$$f(\ldots, x_1, x_2, \ldots) = g(\ldots, x_2 - x_1, \ldots)$$

How does knowing this reduce the search space?

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



2. Translational **symmetry**

$$f(\dots, x_1, x_2, \dots) = g(\dots, x_2 - x_1, \dots)$$

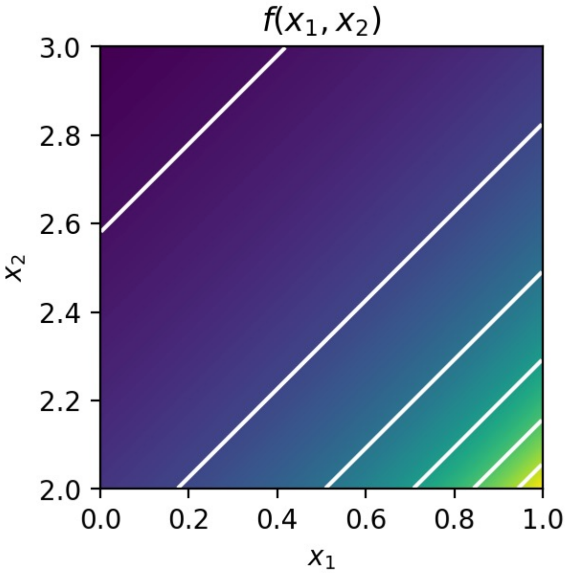
We can write

$$f = g(G, m_1, m_2, d_1, d_2, d_3)$$
$$d_1, d_2, d_3 = (x_2 - x_1), (y_2 - y_1), (z_2 - z_1)$$

Which again reduces the number of variables

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



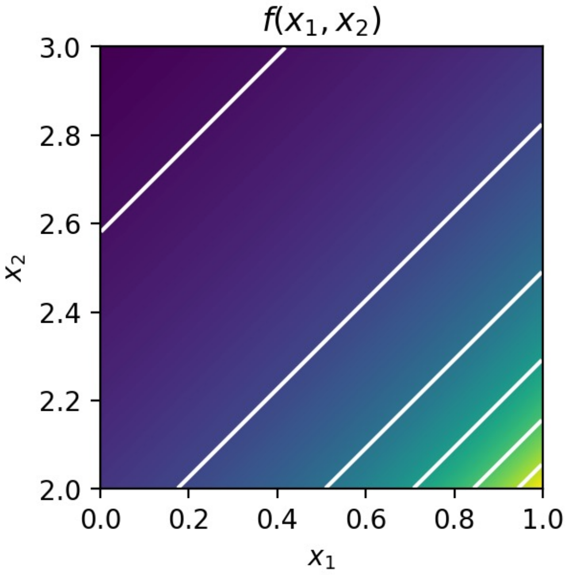
2. Translational **symmetry**

$$f(\dots, x_1, x_2, \dots) = g(\dots, x_2 - x_1, \dots)$$

How can we test for symmetry (given the ability to query f)?

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



2. Translational **symmetry**

$$f(\dots, x_1, x_2, \dots) = g(\dots, x_2 - x_1, \dots)$$

How can we test for symmetry (given the ability to query f)?

For some constant a, test if:

$$f(\dots, x_1, x_2, \dots) = f(\dots, x_1 + a, x_2 + a, \dots) \quad \forall \boldsymbol{x}$$

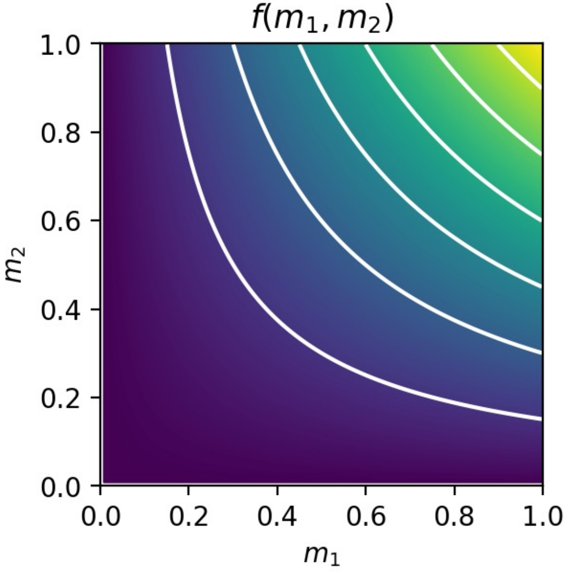
AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{Gm_1m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

3. Multiplicative **separability**

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



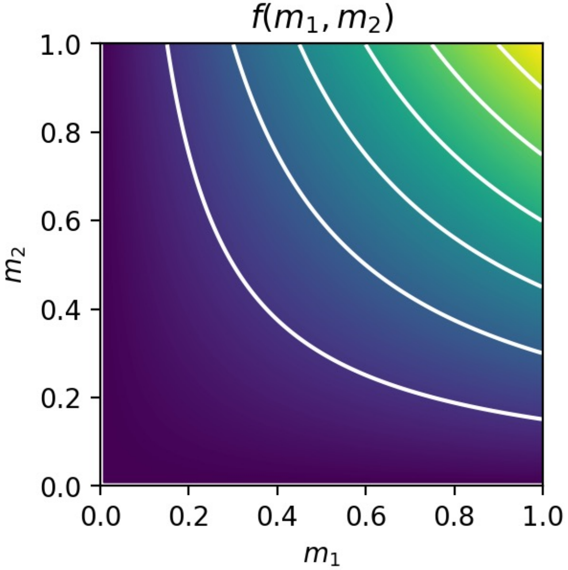
3. Multiplicative **separability**

$$f = g(G)h(m_1)i(m_2)j(x_1, x_2, y_1, y_2, z_1, z_2)$$

How does knowing this reduce the search space?

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{Gm_1m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



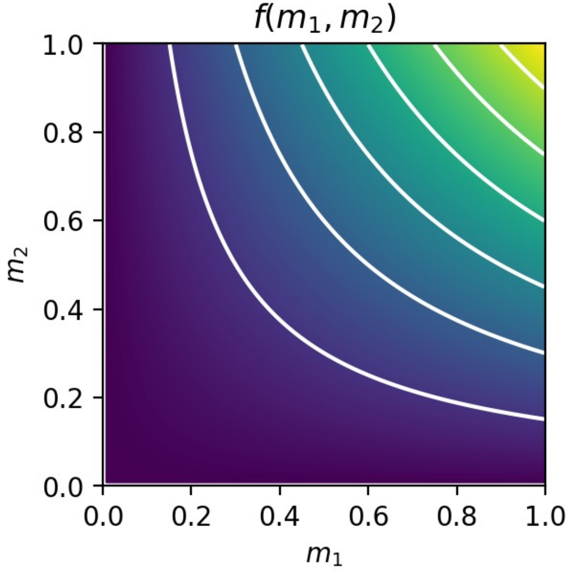
3. Multiplicative **separability**

$$f = g(G)h(m_1)i(m_2)j(x_1, x_2, y_1, y_2, z_1, z_2)$$

Allows us to carry out four **independent** searches for g, h, i, j

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



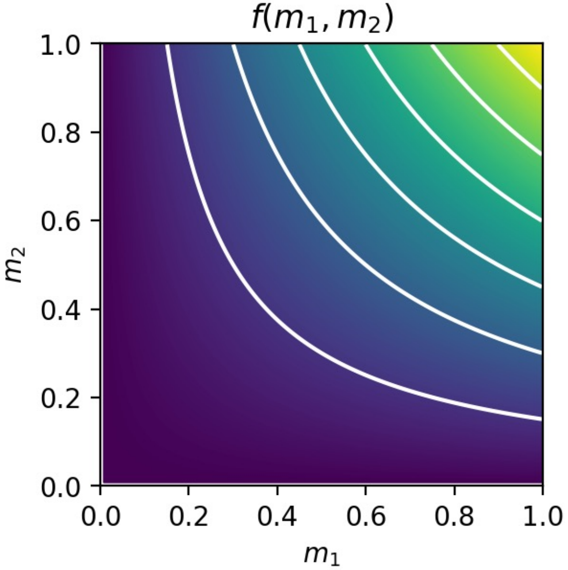
3. Multiplicative **separability**

$$f = g(G)h(m_1)i(m_2)j(x_1, x_2, y_1, y_2, z_1, z_2)$$

How can we test e.g. $f(x_1, x_2) = g(x_1)h(x_2)$ (given the ability to query f)?

AI Feynman

$$f(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2) = \frac{G m_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



3. Multiplicative **separability**

$$f = g(G)h(m_1)i(m_2)j(x_1, x_2, y_1, y_2, z_1, z_2)$$

How can we test e.g. $f(x_1, x_2) = g(x_1)h(x_2)$ (given the ability to query f)?
For some constants c_1 and c_2, test if:

$$f(x_1, x_2) = \frac{f(x_1, c_2)f(c_1, x_2)}{f(c_1, c_2)} \quad \forall \boldsymbol{x}$$

AI Feynman

Mystery function

$$f\,(G, m_1, m_2, x_1, x_2, y_1, y_2, z_1, z_2)$$

Dimensionality analysis

$$= \frac{Gm_1^2}{x_1^2}\alpha(a, b, c, d, e, \mathrm{f}), \qquad a, b, c, d, e, \mathrm{f} = \frac{m_2}{m_1}, \frac{x_2}{x_1}, \frac{y_2}{x_1}, \frac{y_1}{x_1}, \frac{z_2}{x_1}, \frac{y_1}{x_1}$$

Requires us to query f

Symmetry testing

$$= \frac{Gm_1^2}{x_1^2}\beta(a, b, g, h), \qquad g, h = (c - d), (e - \mathrm{f})$$

Separability testing

$$= \frac{Gm_1^2}{x_1^2}a\gamma(b, g, h)$$

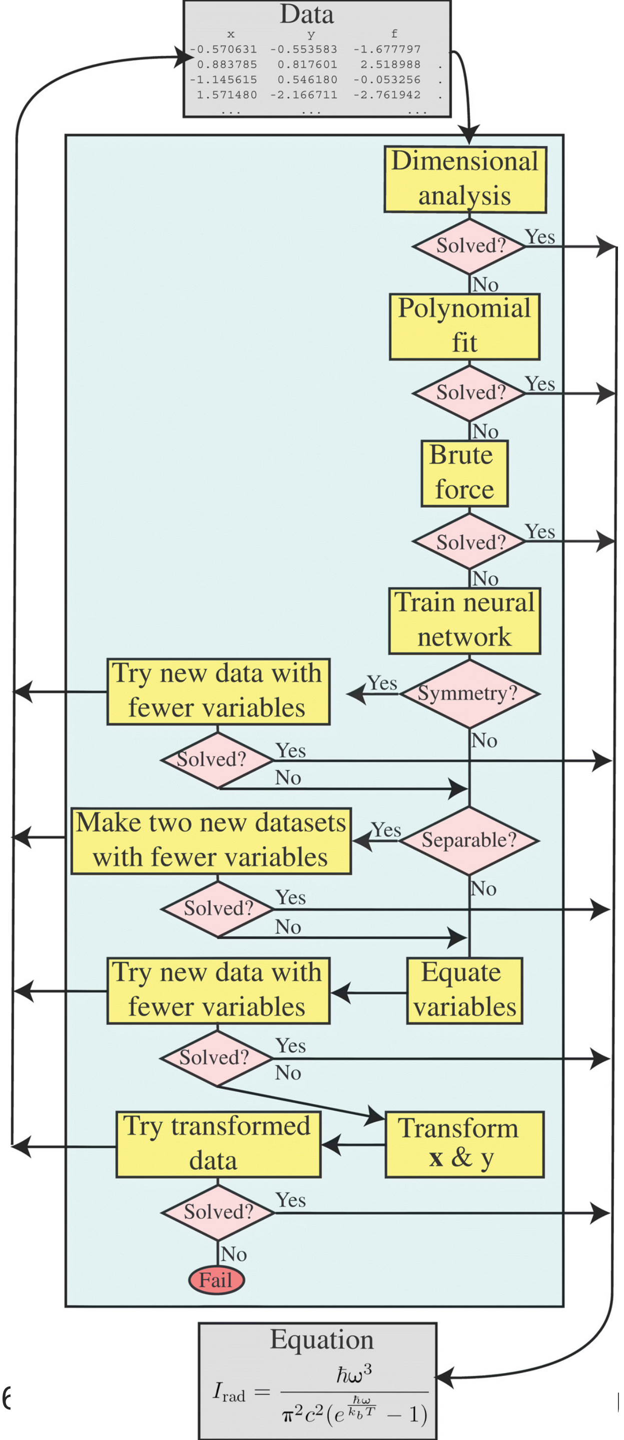
Brute-force search

$$= \frac{Gm_1^2}{x_1^2}a\frac{1}{(b-1)^2 + g^2 + h^2}$$

Re-substitute variables

$$= \frac{Gm_1 m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Full workflow



Data

x	y	f	
-0.570631	-0.553583	-1.677797	
0.883785	0.817601	2.518988	.
-1.145615	0.546180	-0.053256	.
1.571480	-2.166711	-2.761942	.
...	...	...	

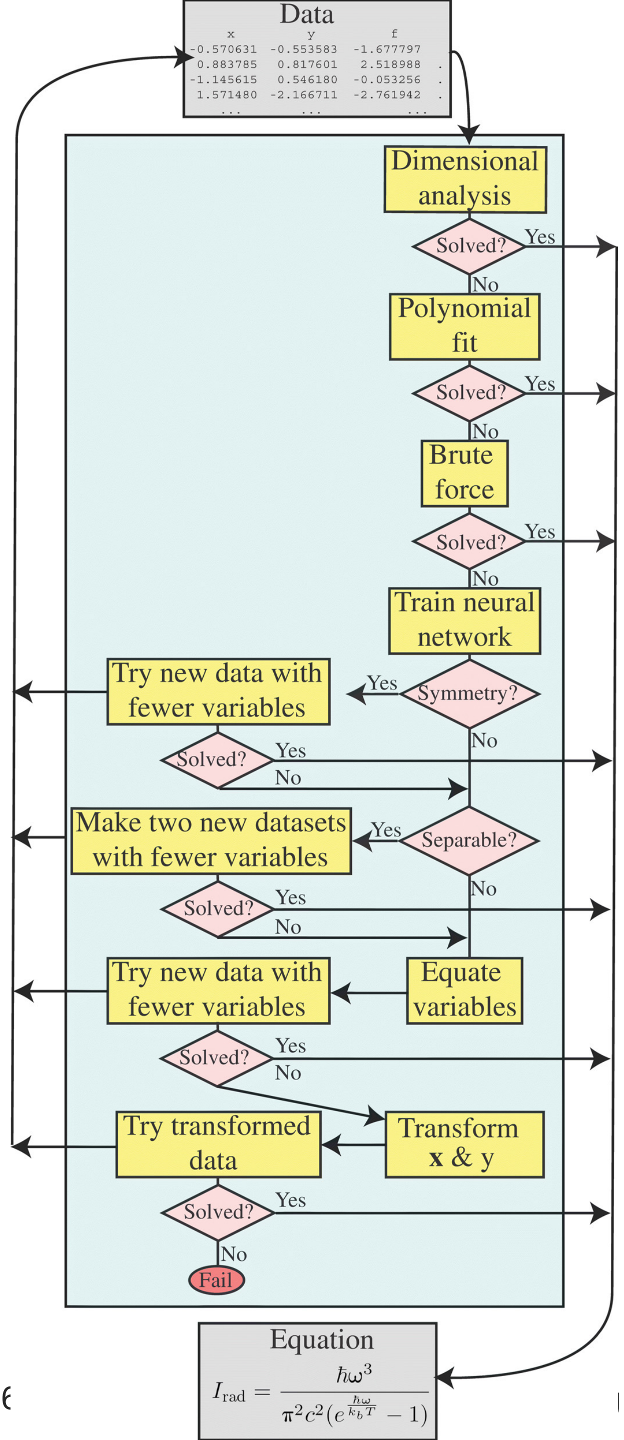
Equation

$$I_{\text{rad}} = \frac{\hbar\omega^3}{\pi^2 c^2 (e^{\frac{\hbar\omega}{k_b T}} - 1)}$$

Udrescu and Tegmark, AI Feynman: A physics-inspired method for symbolic regression. Science Advances (2020)

Full workflow

A neural network $NN(\boldsymbol{x}, \boldsymbol{\theta}) \approx f(\boldsymbol{x})$ is trained simply so we can query $f(\boldsymbol{x})$ anywhere



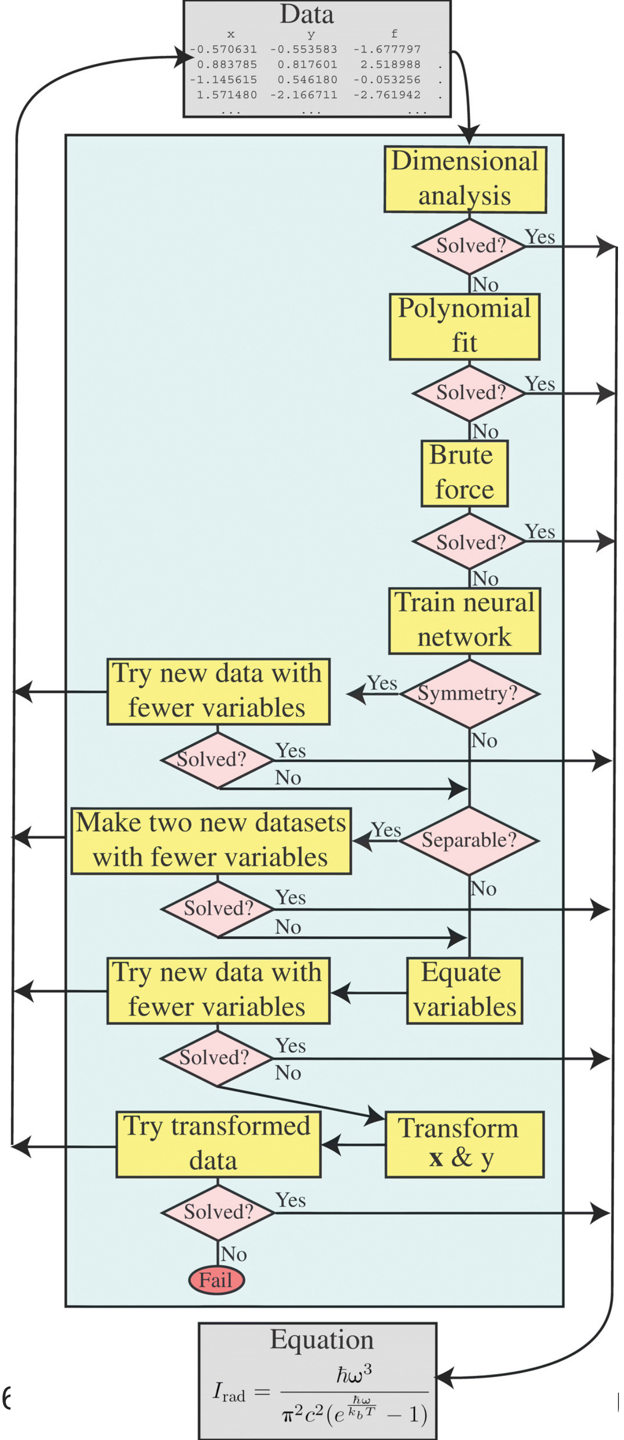
Udrescu and Tegmark, AI Feynman: A physics-inspired method for symbolic regression. Science Advances (2020)

Full workflow

A neural network $NN(\boldsymbol{x}, \boldsymbol{\theta}) \approx f(\boldsymbol{x})$ is trained simply so we can query $f(\boldsymbol{x})$ anywhere

AI Feynman looks for ways to **simplify** the expression to make the search **easier**

Udrescu and Tegmark, AI Feynman: A physics-inspired method for symbolic regression. Science Advances (2020)



Data

x	y	f	
-0.570631	-0.553583	-1.677797	
0.883785	0.817601	2.518988	.
-1.145615	0.546180	-0.053256	.
1.571480	-2.166711	-2.761942	.
...	...	...	

Equation

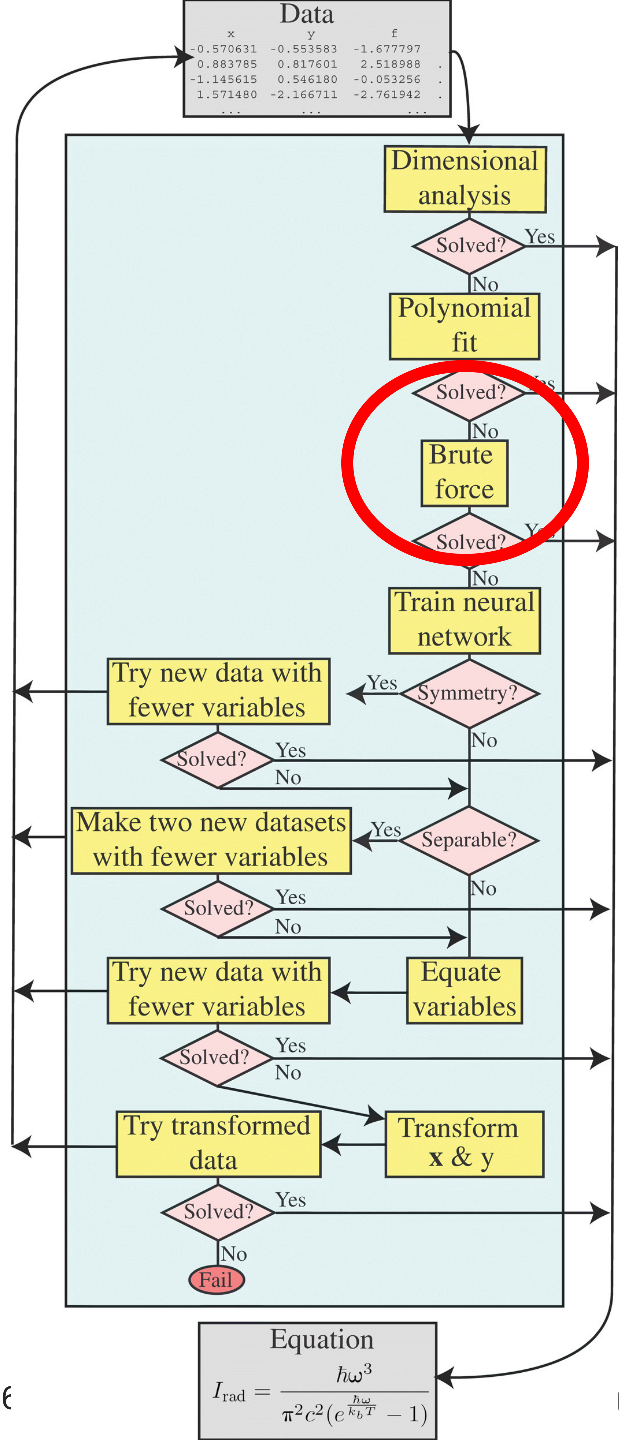
$$I_{\mathrm{rad}} = \frac{\hbar\omega^3}{\pi^2 c^2 (e^{\frac{\hbar\omega}{k_b T}} - 1)}$$

Full workflow

A neural network $NN(\boldsymbol{x}, \boldsymbol{\theta}) \approx f(\boldsymbol{x})$ is trained simply so we can query $f(\boldsymbol{x})$ anywhere

AI Feynman looks for ways to **simplify** the expression to make the search **easier**

Udrescu and Tegmark, AI Feynman: A physics-inspired method for symbolic regression. Science Advances (2020)



Data

x	y	f
-0.570631	-0.553583	-1.677797
0.883785	0.817601	2.518988
-1.145615	0.546180	-0.053256
1.571480	-2.166711	-2.761942
...	...	...

Equation

$$I_{\mathrm{rad}} = \frac{\hbar\omega^3}{\pi^2 c^2 (e^{\frac{\hbar\omega}{k_b T}} - 1)}$$

Even so, the resulting search problem may still be **hard** to solve ⚠

We may be able to **improve** on brute-force (combinatorial) search

Requirements

To successfully solve a symbolic regression problem, we need:

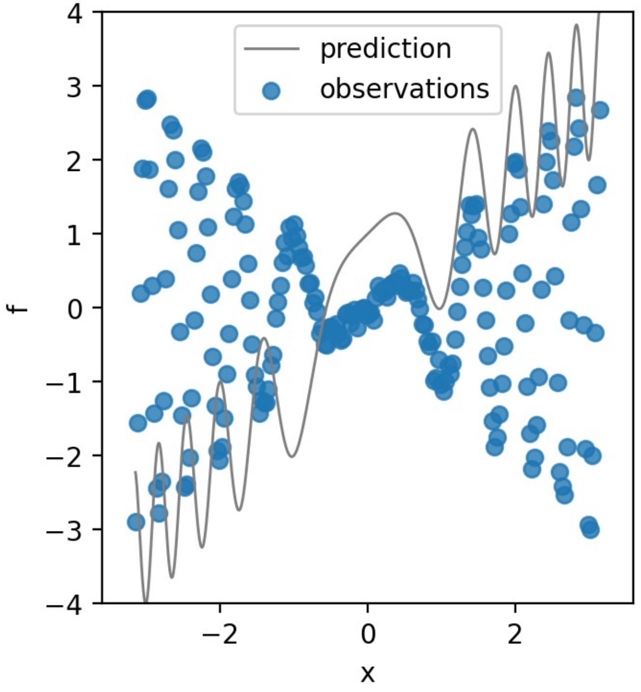
1. An **assumption** (prior) on the structure of the expression ✅

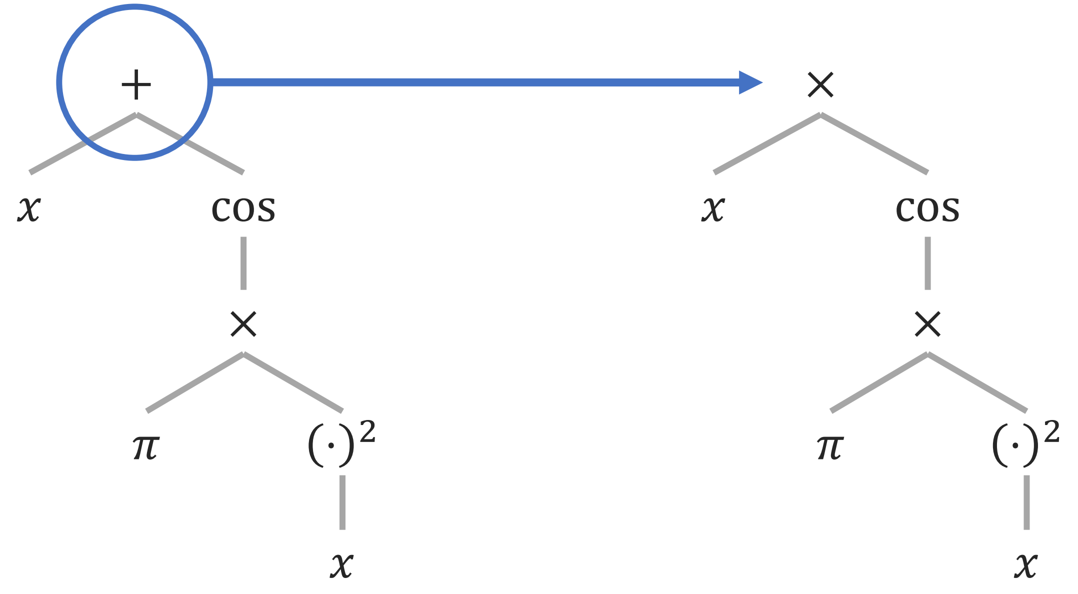
2. A **search** algorithm

… there’s a lot of innovation in both areas!

See e.g. here for state-of-the-art reviews:
Makke & Chawla, Interpretable scientific discovery with symbolic regression: a review, AI Review (2024)
Landajuela et al, A Unified Framework for Deep Symbolic Regression, NeurIPS (2022)

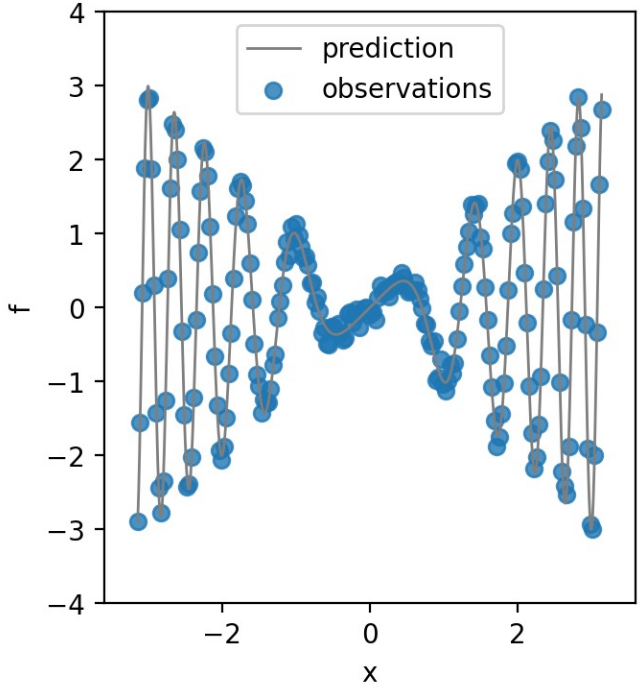
Mutation



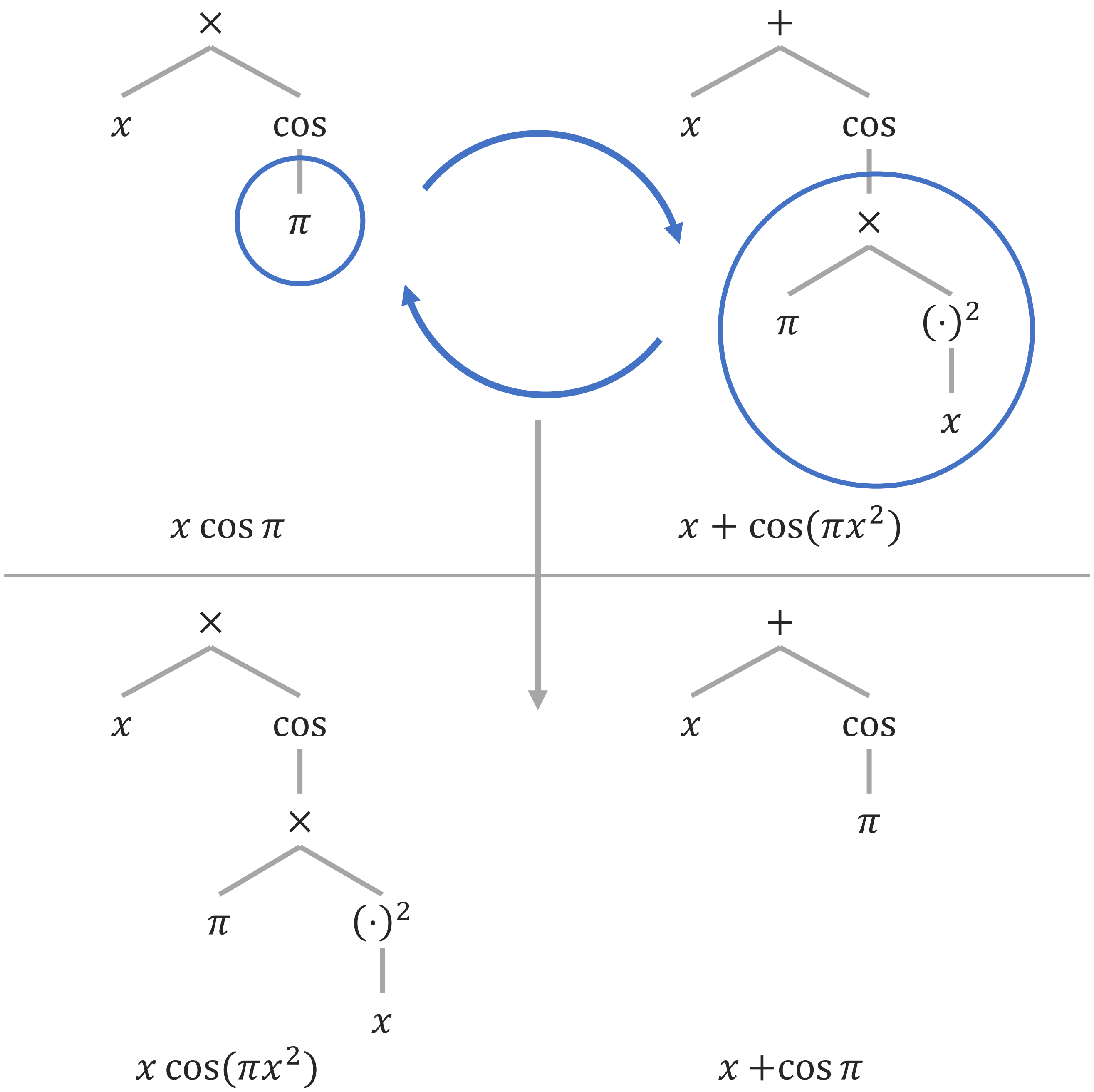


$x + \cos(\pi x^2)$

$x \cos(\pi x^2)$



Crossover


×
x
$\cos$
π
+
x
$\cos$
×
π
$(\cdot)^2$
x
$x \cos \pi$
$x + \cos(\pi x^2)$
×
x
$\cos$
×
π
$(\cdot)^2$
x
+
x
$\cos$
π
$x \cos(\pi x^2)$
$x + \cos \pi$

Genetic search algorithms

1. Start with a random population of trees

2. Loop:

    1. Select "fittest" trees
        - E.g. based on test error

    2. Apply "genetic operators" with specified probabilities
        - Mutation
        - Crossover

    3. Remove "oldest" trees

3. Until an acceptable solution is found

PySR

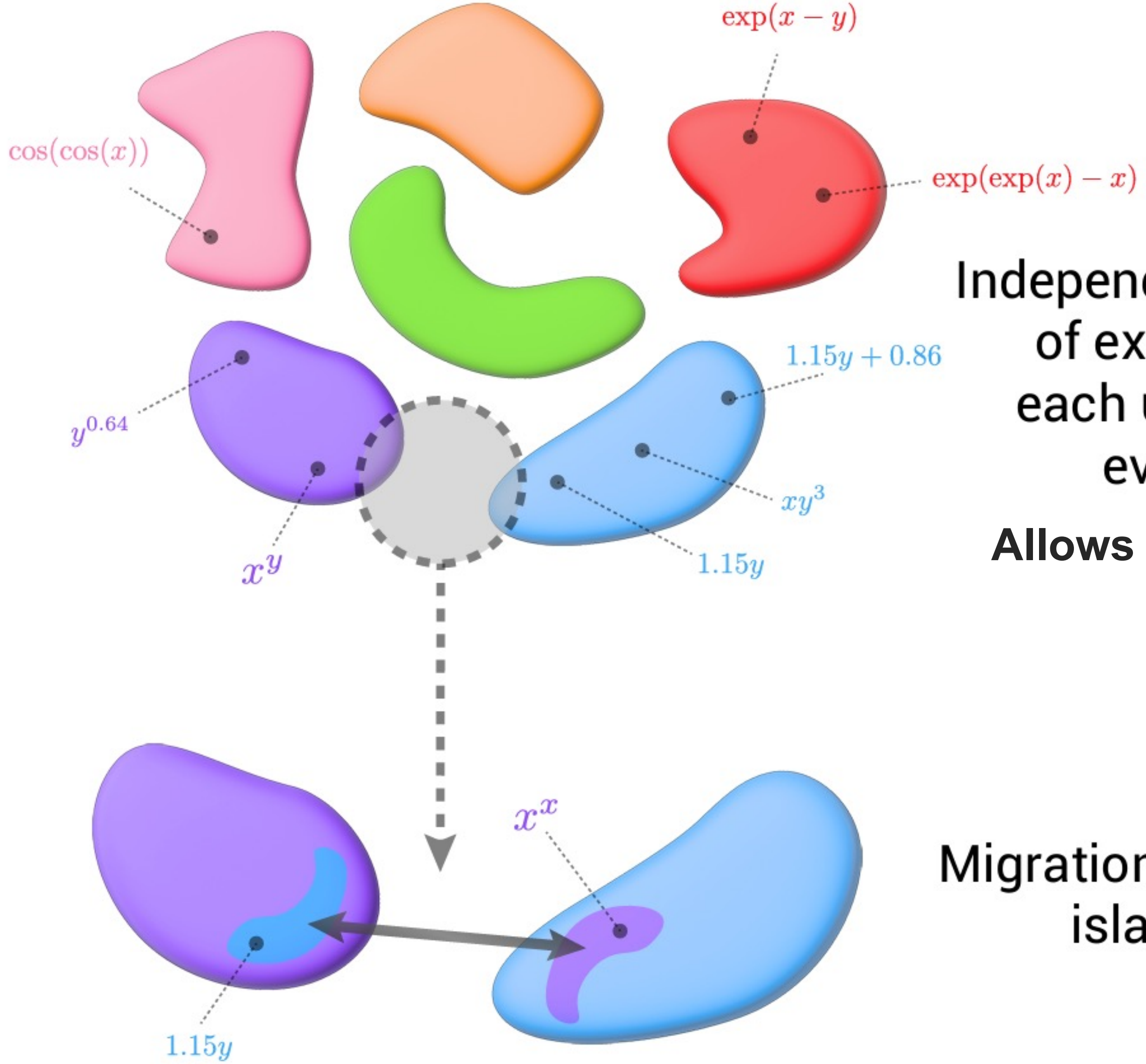
PySR and SymbolicRegression.jl

PySR

Source:
github.com/MilesCranmer/PySR

Cranmer, Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl, ArXiv (2023)

Tournament selection



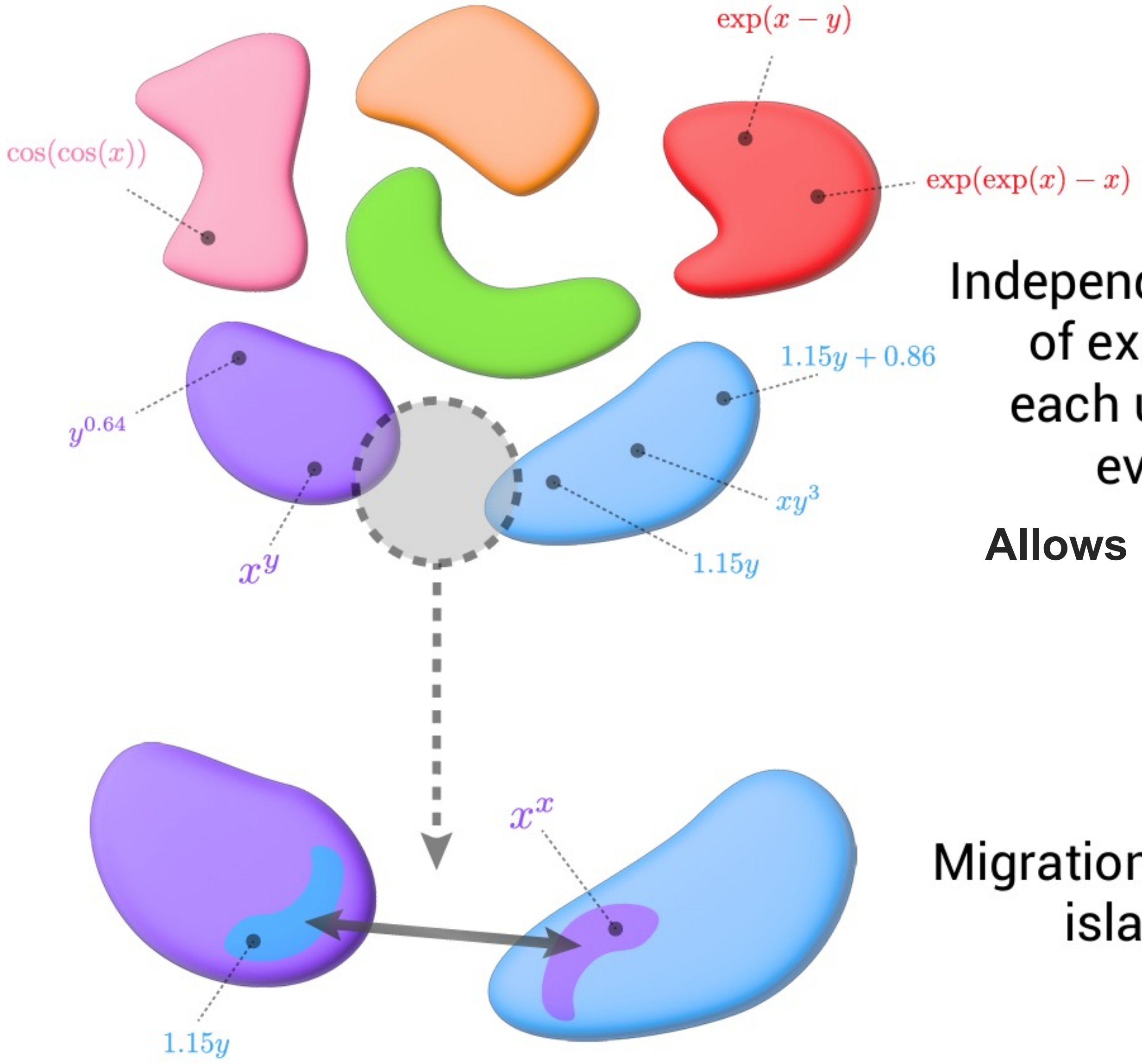
Independent "islands" of expressions, each undergoing evolution

Allows parallelisation

Migration between islands

Cranmer, Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl, ArXiv (2023)

Tournament selection



Independent “islands” of expressions, each undergoing evolution

Allows parallelisation

Migration between islands

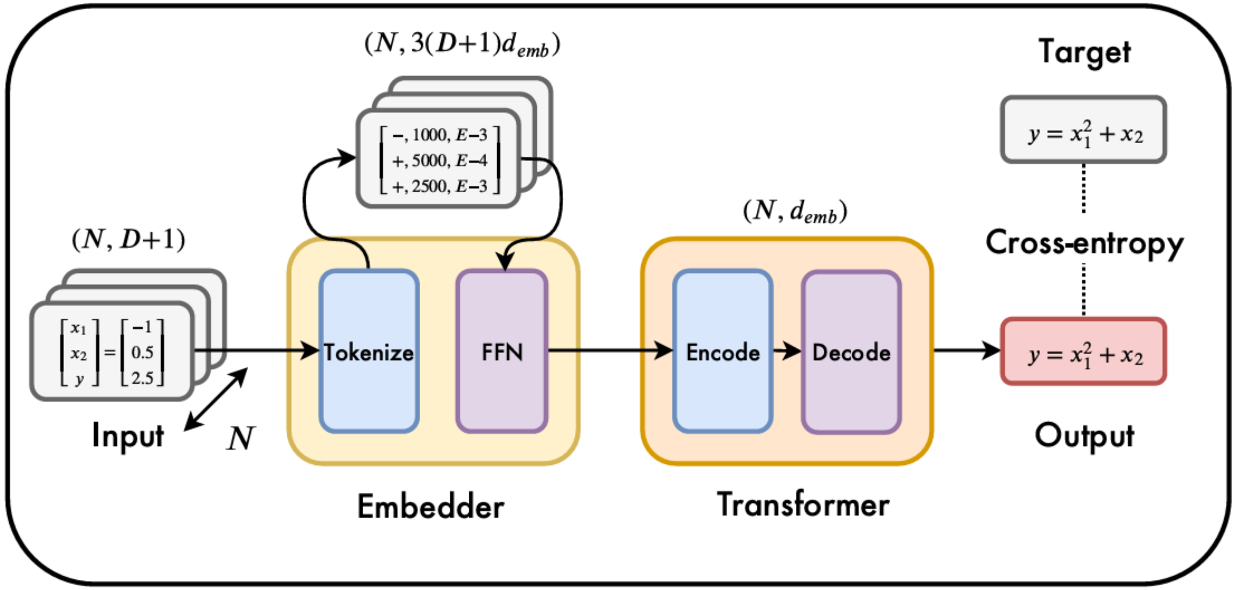
How does the number of “islands” affect performance?

Cranmer, Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl, ArXiv (2023)

Other search algorithms

Goal: find f given $D = \{(\boldsymbol{x}_1, f_1), \ldots, (\boldsymbol{x}_N, f_N)\}$

- **Directly** (no search) using a neural network (e.g. Transformer)



$(N, 3(D+1)d_{emb})$
Target
$\begin{bmatrix} -, 1000, E{-}3 \\ +, 5000, E{-}4 \\ +, 2500, E{-}3 \end{bmatrix}$
$y = x_1^2 + x_2$
(N, d_{emb})
Cross-entropy
$(N, D+1)$
$\begin{bmatrix} x_1 \\ x_2 \\ y \end{bmatrix} = \begin{bmatrix} -1 \\ 0.5 \\ 2.5 \end{bmatrix}$
Tokenize
FFN
Encode
Decode
$y = x_1^2 + x_2$
Input
N
Output
Embedder
Transformer

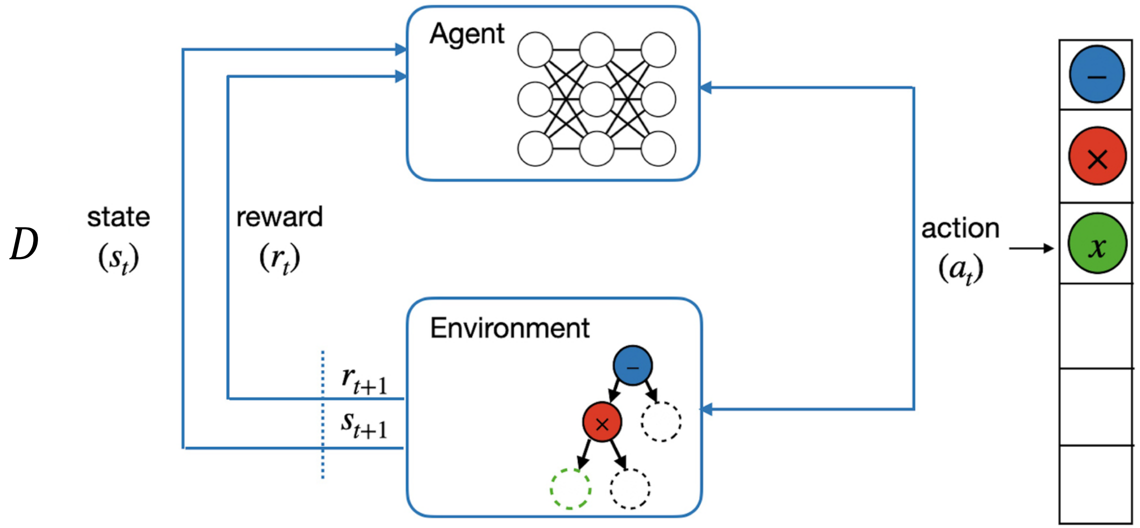
Kamienny et al, End-to-end symbolic regression with transformers, NeurIPS (2022)

Other search algorithms

Goal: find f given $D = \{(\boldsymbol{x}_1, f_1), \dots, (\boldsymbol{x}_N, f_N)\}$

- **Directly** (no search) using a neural network (e.g. Transformer)

- By using **reinforcement learning** (building expressions incrementally)



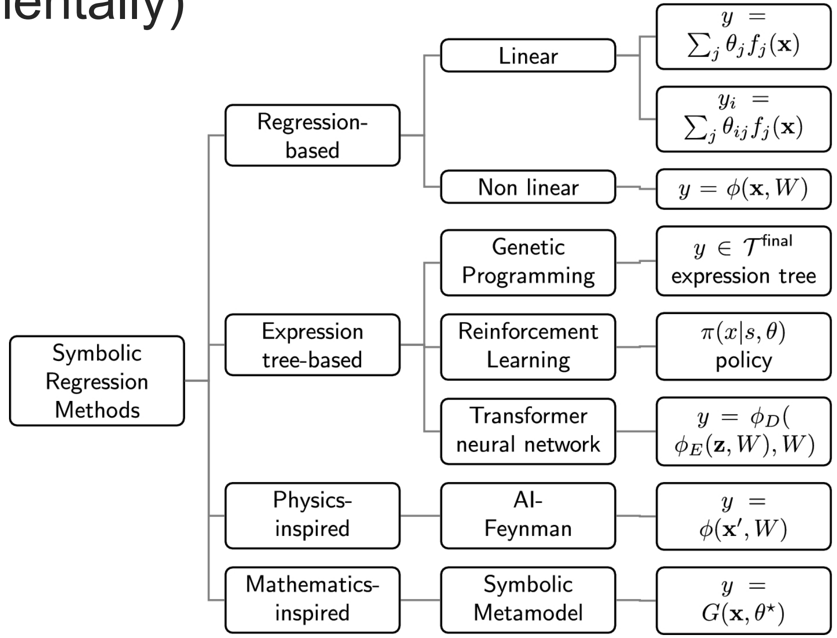
Makke & Chawla, Interpretable scientific discovery with symbolic regression: a review, AI Review (2024)

Petersen et al, Deep symbolic regression: recovering mathematical expressions from data via policy gradients, ICLR (2021)

Other search algorithms

Goal: find f given $D = \{(\boldsymbol{x}_1, f_1), \ldots, (\boldsymbol{x}_N, f_N)\}$

- **Directly** (no search) using a neural network (e.g. Transformer)
- By using **reinforcement learning** (building expressions incrementally)
- By learning a **tree search** algorithm
- + many others…



Symbolic Regression Methods
Regression-based
Linear
$y = \sum_j \theta_j f_j(\mathbf{x})$
$y_i = \sum_j \theta_{ij} f_j(\mathbf{x})$
Non linear
$y = \phi(\mathbf{x}, W)$
Expression tree-based
Genetic Programming
$y \in \mathcal{T}^{\text{final}}$ expression tree
Reinforcement Learning
$\pi(x|s,\theta)$ policy
Transformer neural network
$y = \phi_D(\phi_E(\mathbf{z}, W), W)$
Physics-inspired
AI-Feynman
$y = \phi(\mathbf{x}', W)$
Mathematics-inspired
Symbolic Metamodel
$y = G(\mathbf{x}, \theta^\star)$

Makke & Chawla, Interpretable scientific discovery with symbolic regression: a review, AI Review (2024)

Lecture overview

- What is model discovery?
- Challenges of symbolic regression
- Function discovery
    - AI Feynman
    - Genetic algorithms
- Model discovery
    - SINDy
    - Other approaches

Learning objectives

- Understand how symbolic regression (SR) algorithms are designed
- Understand how SR is used for function and model discovery

5 min break

Function discovery

Task:

Given **observations** of some **function** $f(\boldsymbol{x})$,

$$D = \{(\boldsymbol{x}_1, f_1), \ldots, (\boldsymbol{x}_N, f_N)\}$$

Find its **mathematical expression**

$$PV = nRT$$

$$F = k\frac{q_1 q_2}{r^2}$$

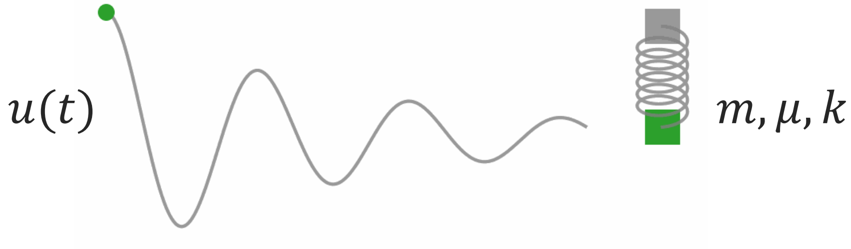
$$E = h\nu$$

$$P = \sigma A T^4$$

Model discovery

Task:

Given **observations** of a physical system



Find an underlying **model**

$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$

Function discovery

Task:

Given **observations** of some **function** $f(\boldsymbol{x})$,

$$D = \{(\boldsymbol{x}_1, f_1), \dots, (\boldsymbol{x}_N, f_N)\}$$

Find its **mathematical expression**

$$PV = nRT$$

$$F = k\frac{q_1 q_2}{r^2}$$

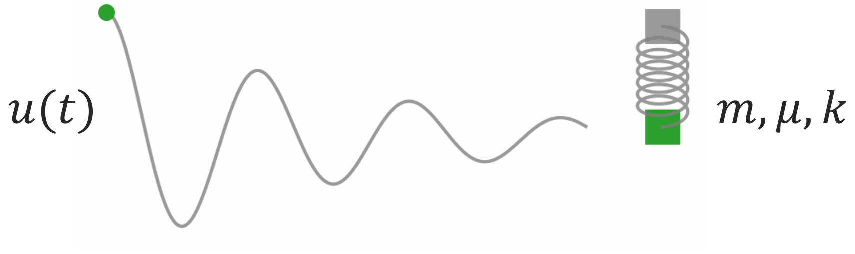
$$E = h\nu$$

$$P = \sigma A T^4$$

Model discovery

Task:

Given **observations** of a physical system



Find an underlying **model**

$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$

- Both can use **symbolic regression** for discovery
- Model discovery usually combines SR with **domain constraints** and adds **extra operators** (e.g. derivatives)

SINDy Sparse Identification of Nonlinear Dynamics

Assume an unknown dynamical system has the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x})$$

Task:

Given many examples

$$
\begin{aligned}
&D = \{ \\
&([\boldsymbol{x}_1(t_1), \dot{\boldsymbol{x}}_1(t_1)], \dots, [\boldsymbol{x}_1(t_M), \dot{\boldsymbol{x}}_1(t_M)]), \\
&\dots \\
&([\boldsymbol{x}_N(t_1), \dot{\boldsymbol{x}}_N(t_1)], \dots, [\boldsymbol{x}_N(t_M), \dot{\boldsymbol{x}}_N(t_M)]) \\
&\}
\end{aligned}
$$

Find $\boldsymbol{f}(\boldsymbol{x})$

Brunton et al, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, PNAS, (2016)

SINDy

Assume an unknown dynamical system has the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x})$$

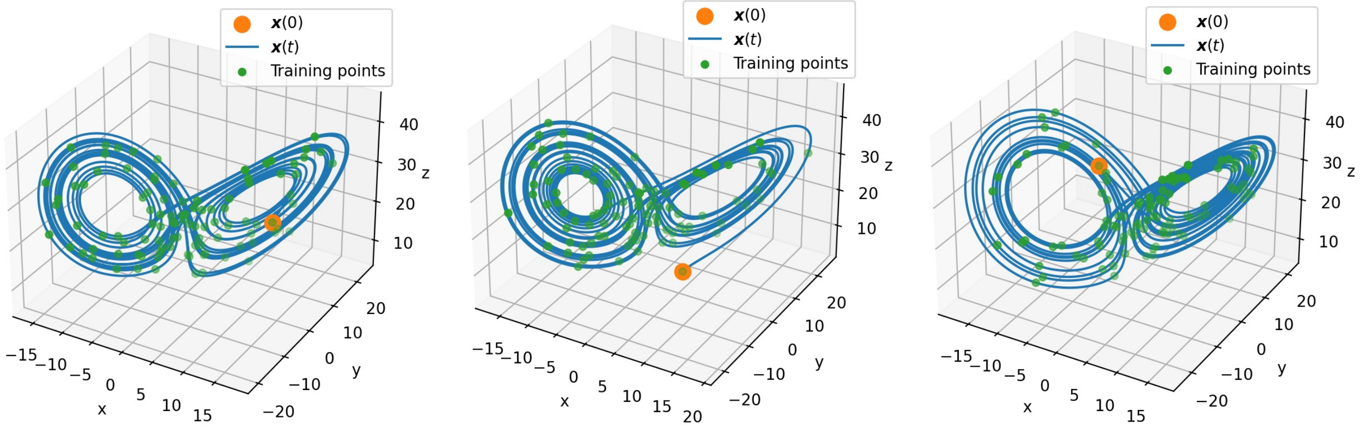
For example, the Lorenz system

$$\begin{aligned}\dot{x} &= \sigma(y - x)\\ \dot{y} &= x(\rho - z) - y\\ \dot{z} &= xy - \beta z\end{aligned}$$

Task:

Given many examples

$$D = \{$$
$$([\boldsymbol{x}_1(t_1), \dot{\boldsymbol{x}}_1(t_1)], \dots, [\boldsymbol{x}_1(t_M), \dot{\boldsymbol{x}}_1(t_M)]),$$
$$\dots$$
$$([\boldsymbol{x}_N(t_1), \dot{\boldsymbol{x}}_N(t_1)], \dots, [\boldsymbol{x}_N(t_M), \dot{\boldsymbol{x}}_N(t_M)])$$
$$\}$$



Find $\boldsymbol{f}(\boldsymbol{x})$

Brunton et al, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, PNAS, (2016)

SINDy

Assume an unknown dynamical system has the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x})$$

Task:

Given many examples

$$D = \{$$
$$([\boldsymbol{x}_1(t_1), \dot{\boldsymbol{x}}_1(t_1)], \dots, [\boldsymbol{x}_1(t_M), \dot{\boldsymbol{x}}_1(t_M)]),$$
$$\dots$$
$$([\boldsymbol{x}_N(t_1), \dot{\boldsymbol{x}}_N(t_1)], \dots, [\boldsymbol{x}_N(t_M), \dot{\boldsymbol{x}}_N(t_M)])$$
$$\}$$

Find $\boldsymbol{f}(\boldsymbol{x})$

Brunton et al, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, PNAS, (2016)

Note:

We are given measurements of $\dot{\boldsymbol{x}} = \boldsymbol{f}$

Then the training data can simply be written as

$$D = \{(\boldsymbol{x}_1, \boldsymbol{f}_1), \dots, (\boldsymbol{x}_{NM}, \boldsymbol{f}_{NM})\}$$

Which is **the same SR task** as above, except that we need to find a **vector-valued** function

SINDy

Assume that $\boldsymbol{f}(\boldsymbol{x})$ can be written as

$$\boldsymbol{f}^T(\boldsymbol{x}) = \boldsymbol{\phi}^T(\boldsymbol{x})\Lambda$$

Where $\boldsymbol{\phi}(\boldsymbol{x})$ is a **library** of expressions

And Λ is an (unknown) **sparse** matrix of coefficients

E.g.

$$\boldsymbol{f}^T(\boldsymbol{x}) = \overset{\boldsymbol{\phi}^T(\boldsymbol{x})}{\begin{pmatrix} 1 & x & y & z & xz & \dots \end{pmatrix}} \overset{\Lambda}{\begin{pmatrix} 0 & 0 & 0 \\ -\sigma & \rho & 0 \\ \sigma & -1 & 0 \\ 0 & 0 & -\beta \\ 0 & -1 & 0 \\ \dots & \dots & \dots \end{pmatrix}}$$

$$= \begin{pmatrix} \sigma(y-x) & x(\rho - z) - y & xy - \beta z \end{pmatrix}$$

SINDy

Assume that $\boldsymbol{f}(\boldsymbol{x})$ can be written as

$$\boldsymbol{f}^T(\boldsymbol{x}) = \boldsymbol{\phi}^T(\boldsymbol{x})\Lambda$$

Where $\boldsymbol{\phi}(\boldsymbol{x})$ is a **library** of expressions

And Λ is an (unknown) **sparse** matrix of coefficients

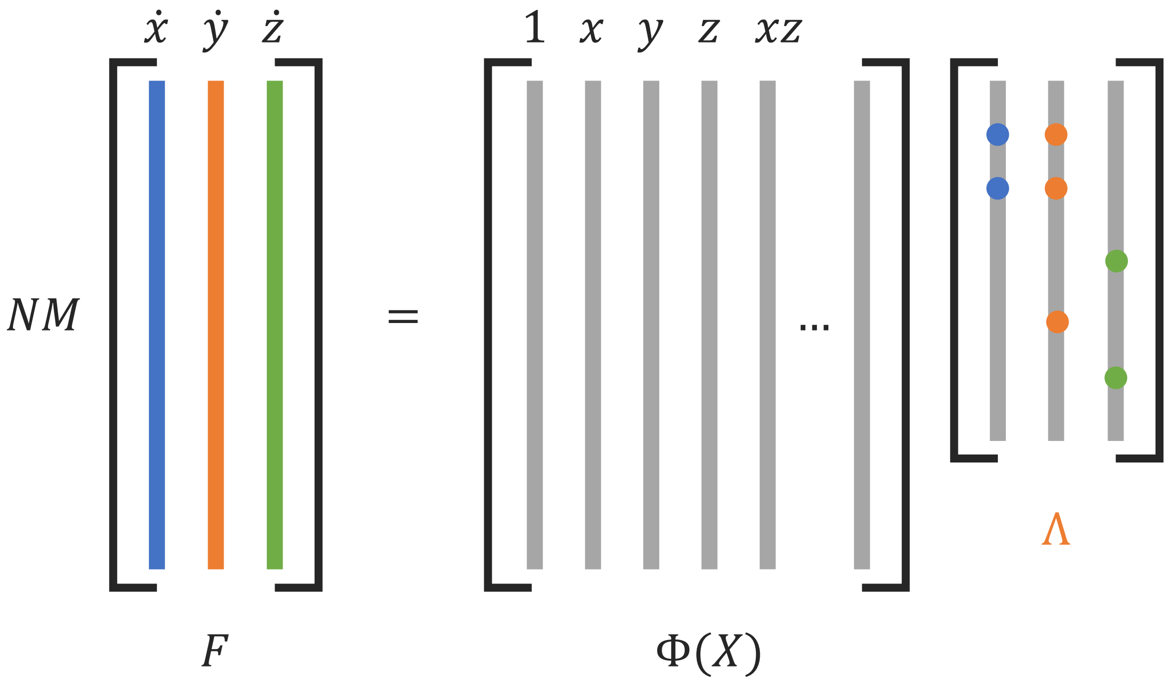
E.g.

$$\begin{array}{cc} & \boldsymbol{\phi}^T(\boldsymbol{x}) \qquad\qquad \Lambda \\ \boldsymbol{f}^T(\boldsymbol{x}) = & (1 \quad x \quad y \quad z \quad xz \quad \ldots)\begin{pmatrix} 0 & 0 & 0 \\ -\sigma & \rho & 0 \\ \sigma & -1 & 0 \\ 0 & 0 & -\beta \\ 0 & -1 & 0 \\ \ldots & \ldots & \ldots \end{pmatrix} \end{array}$$

$$= (\sigma(y-x) \quad x(\rho - z) - y \quad xy - \beta z)$$

Then for all our training data

$$D = \{(\boldsymbol{x}_1, \boldsymbol{f}_1), \ldots, (\boldsymbol{x}_{NM}, \boldsymbol{f}_{NM})\}$$



SINDy

Assume that $\boldsymbol{f}(\boldsymbol{x})$ can be written as

$$\boldsymbol{f}^T(\boldsymbol{x}) = \boldsymbol{\phi}^T(\boldsymbol{x})\Lambda$$

Where $\boldsymbol{\phi}(\boldsymbol{x})$ is a **library** of expressions

And Λ is an (unknown) **sparse** matrix of coefficients

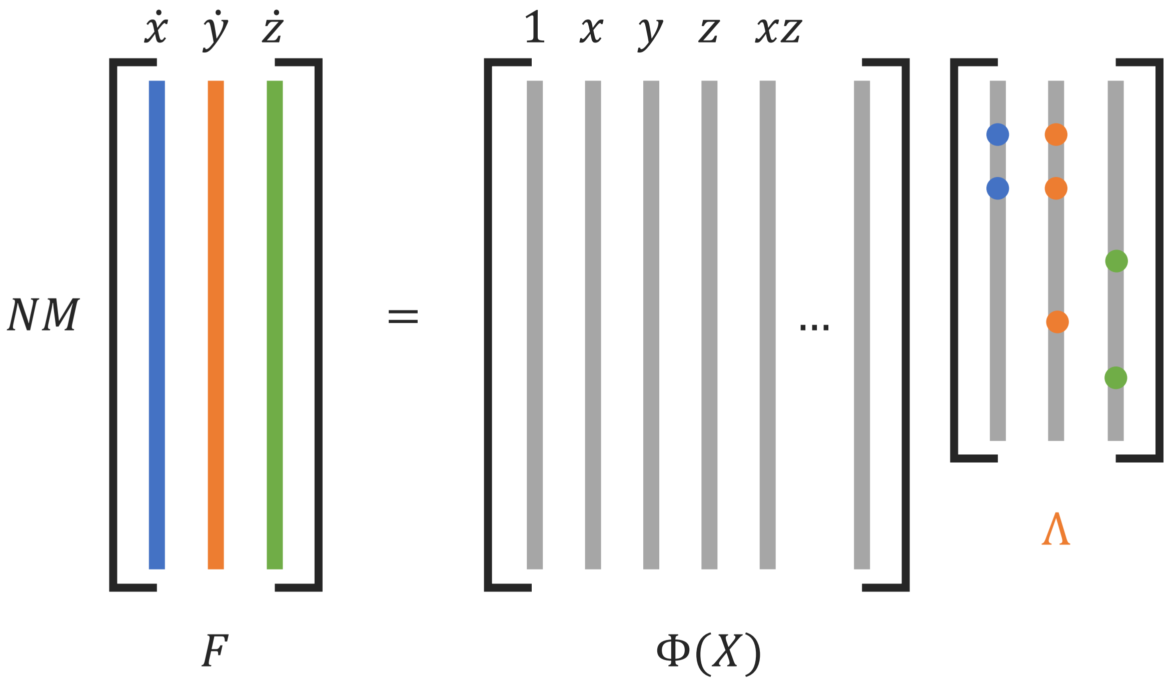
E.g. $\boldsymbol{\phi}^T(\boldsymbol{x})$ Λ

$$\boldsymbol{f}^T(\boldsymbol{x}) = (1 \quad x \quad y \quad z \quad xz \quad \ldots)\begin{pmatrix} 0 & 0 & 0 \\ -\sigma & \rho & 0 \\ \sigma & -1 & 0 \\ 0 & 0 & -\beta \\ 0 & -1 & 0 \\ \ldots & \ldots & \ldots \end{pmatrix}$$

$$= (\sigma(y-x) \quad x(\rho - z) - y \quad xy - \beta z)$$

Then for all our training data

$$D = \{(\boldsymbol{x}_1, \boldsymbol{f}_1), \ldots, (\boldsymbol{x}_{NM}, \boldsymbol{f}_{NM})\}$$



This is just (sparse) **linear regression**

Requirements - SINDy

To successfully solve a symbolic regression problem, we need:

1. An **assumption** (prior) on the structure of the expression

2. A **search** algorithm

Requirements - SINDy

To successfully solve a symbolic regression problem, we need:

1. An **assumption** (prior) on the structure of the expression

2. A **search** algorithm

Expressions must have the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}) = \Lambda^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$$

Limited set of operators, e.g.

$$\boldsymbol{\phi}^T = (1, x, y, z, xy, x^2, \dots)$$

Requirements - SINDy

To successfully solve a symbolic regression problem, we need:

1. An **assumption** (prior) on the structure of the expression
2. A **search** algorithm

Expressions must have the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}) = \Lambda^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$$

Limited set of operators, e.g.

$$\boldsymbol{\phi}^{T} = (1, x, y, z, xy, x^2, \dots)$$

Sparse linear regression (e.g. LASSO) to find Λ

Requirements - SINDy

To successfully solve a symbolic regression problem, we need:

1. An **assumption** (prior) on the structure of the expression
2. A **search** algorithm

Sparse linear regression (e.g. LASSO) to find Λ

Expressions must have the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}) = \Lambda^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$$

Limited set of operators, e.g.

$$\boldsymbol{\phi}^T = (1, x, y, z, xy, x^2, \dots)$$

What are the limitations of SINDy?

Requirements - SINDy

To successfully solve a symbolic regression problem, we need:

1. An **assumption** (prior) on the structure of the expression

Expressions must have the form

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}) = \Lambda^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$$

Limited set of operators, e.g.

$$\boldsymbol{\phi}^{T} = (1, x, y, z, xy, x^{2}, \dots)$$

2. A **search** algorithm

Sparse linear regression (e.g. LASSO) to find Λ

What are the limitations of SINDy?

- Requires measurements of $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$
- Only learns a first-order ODE

SINDy Autoencoders

Assume an unknown dynamical system has the form

$$\frac{d^2\boldsymbol{z}}{dt^2} = \boldsymbol{f}(\boldsymbol{z})$$

Task:

Given many **transformed** observations of $\boldsymbol{z}$

$$D = \{$$
$$[X(\boldsymbol{z}_1(t_1)), \dots, X(\boldsymbol{z}_1(t_M))],$$
$$\dots$$
$$[X(\boldsymbol{z}_N(t_1)), \dots, X(\boldsymbol{z}_N(t_M))]$$
$$\}$$

Find $\boldsymbol{f}(\boldsymbol{z})$

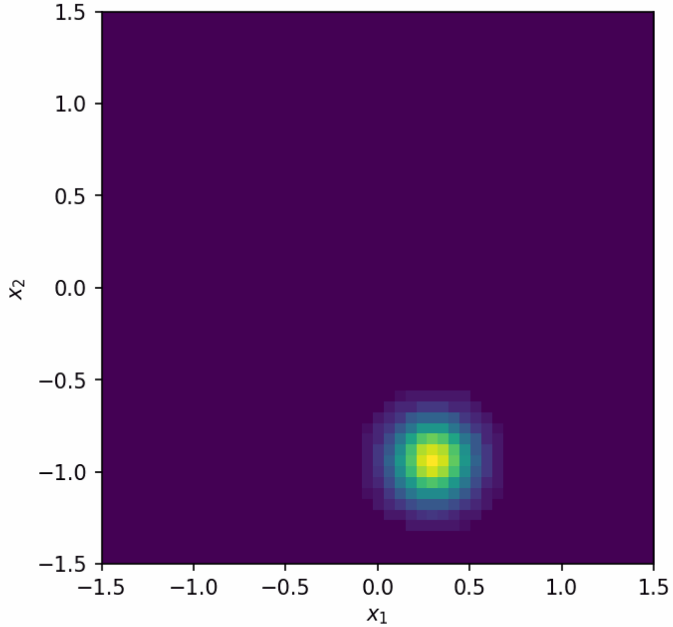
Champion et al, Data-driven discovery of coordinates and governing equations, PNAS (2019)

For example, **nonlinear pendulum**

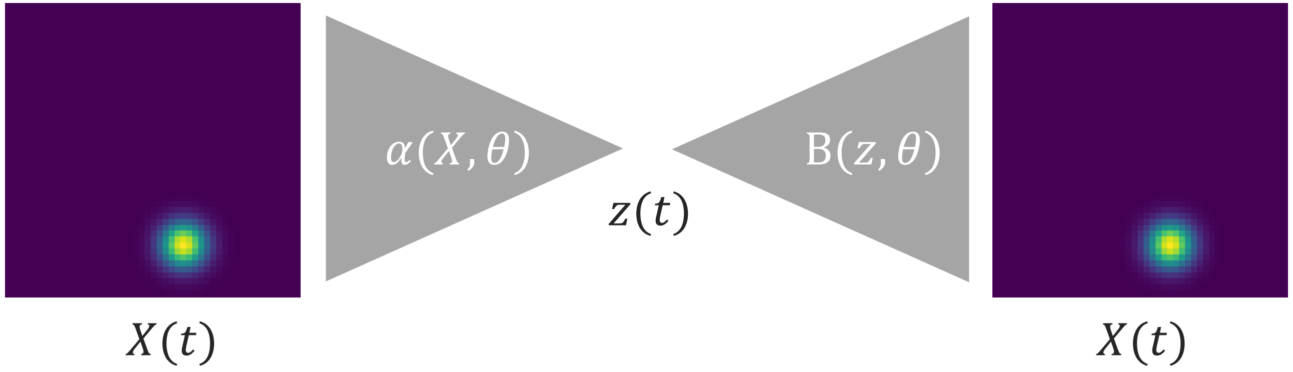
$$\frac{d^2z}{dt^2} = -\sin(z)$$

Where z is the **angle** of the pendulum and X is an **image** of the pendulum

$$X\big(z(t)\big): \mathbb{R}^1 \rightarrow \mathbb{R}^{n\times n}$$



SINDy Autoencoders

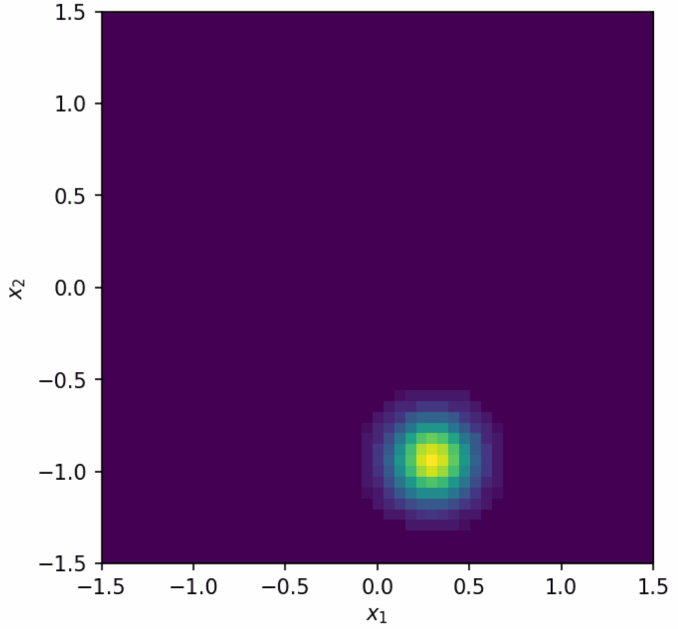


For example, **nonlinear pendulum**

$$\frac{d^2z}{dt^2} = -\sin(z)$$

Where z is the **angle** of the pendulum and X is an **image** of the pendulum

$$X\big(z(t)\big): \mathbb{R}^1 \rightarrow \mathbb{R}^{n\times n}$$



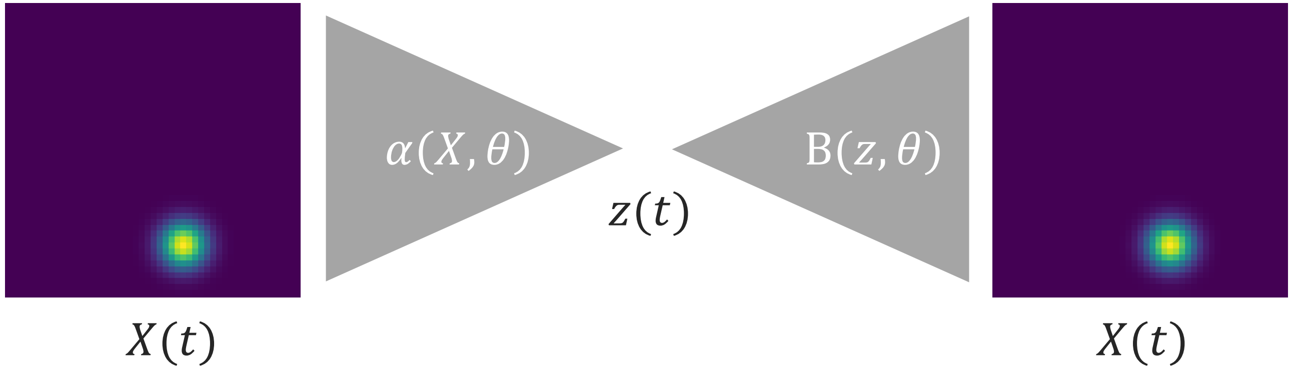
Champion et al, Data-driven discovery of coordinates and governing equations, PNAS (2019)

SINDy Autoencoders

For example, **nonlinear pendulum**

$$\frac{d^2z}{dt^2} = -\sin(z)$$

Where z is the **angle** of the pendulum and X is an **image** of the pendulum

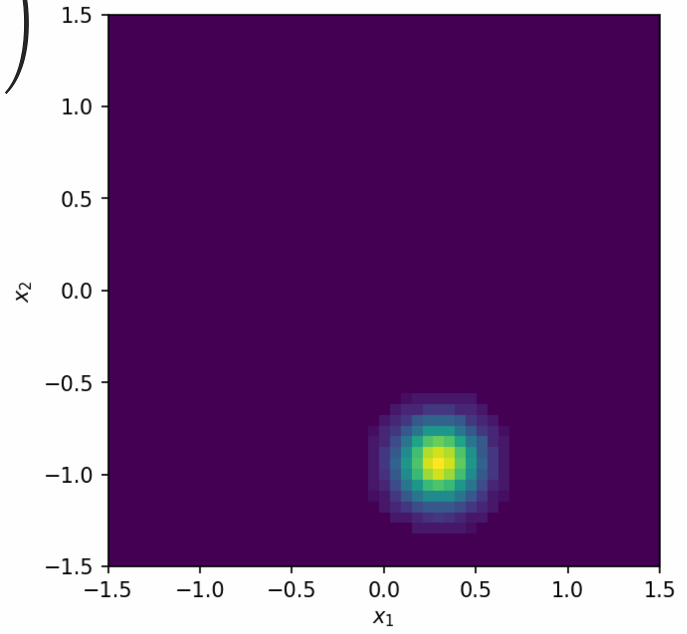


$$L(\boldsymbol{\theta},\Lambda) = \sum_D \left(\|X - \mathrm{B}(\alpha(X,\boldsymbol{\theta}),\boldsymbol{\theta})\|^2 + \left\| \frac{d^2z}{dt^2} - \boldsymbol{\phi}^T\big(\alpha(X,\boldsymbol{\theta})\big)\Lambda \right\|^2 + \|\Lambda\|^1 \right)$$

Reconstruction loss

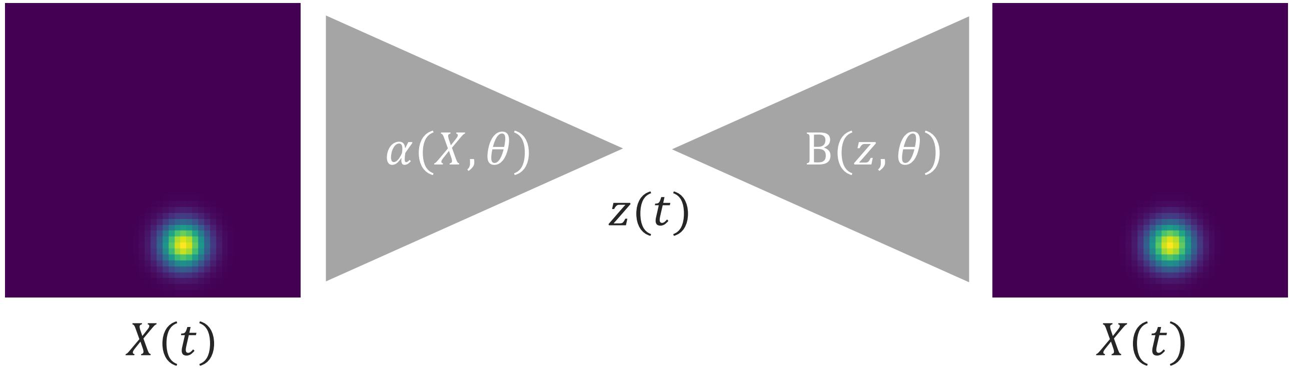
SINDy loss

$X\big(z(t)\big): \mathbb{R}^1 \rightarrow \mathbb{R}^{n\times n}$



Champion et al, Data-driven discovery of coordinates and governing equations, PNAS (2019)

SINDy Autoencoders



For example, **nonlinear pendulum**

$$\frac{d^2z}{dt^2} = -\sin(z)$$

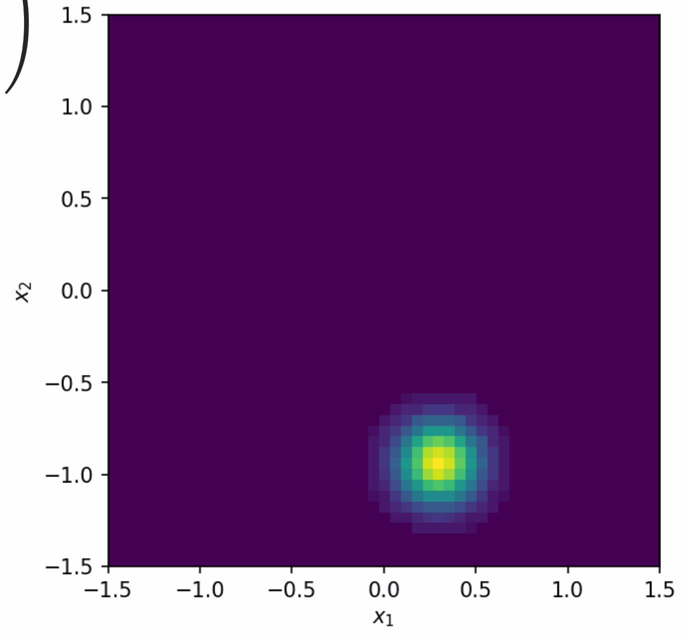
Where z is the **angle** of the pendulum and X is an **image** of the pendulum

$$L(\boldsymbol{\theta},\Lambda) = \sum\nolimits_D \left(\underbrace{\|X - \mathrm{B}(\alpha(X,\boldsymbol{\theta}),\boldsymbol{\theta})\|^2}_{\text{Reconstruction loss}} + \underbrace{\left\| \frac{d^2z}{dt^2} - \boldsymbol{\phi}^T\big(\alpha(X,\boldsymbol{\theta})\big)\Lambda \right\|^2 + \|\Lambda\|^1}_{\text{SINDy loss}} \right)$$

Where $\frac{d^2z}{dt^2}$ can be estimated numerically e.g.

$$\frac{d^2z}{dt^2} \approx \frac{\alpha(X(t+1),\boldsymbol{\theta}) - 2\alpha(X(t),\boldsymbol{\theta}) + \alpha(X(t-1),\boldsymbol{\theta})}{\delta t^2}$$

$X\big(z(t)\big)\colon \mathbb{R}^1 \to \mathbb{R}^{n\times n}$



Champion et al, Data-driven discovery of coordinates and governing equations, PNAS (2019)

Lecture summary

- Function and model discovery is usually extremely challenging because of the **exponential** search space

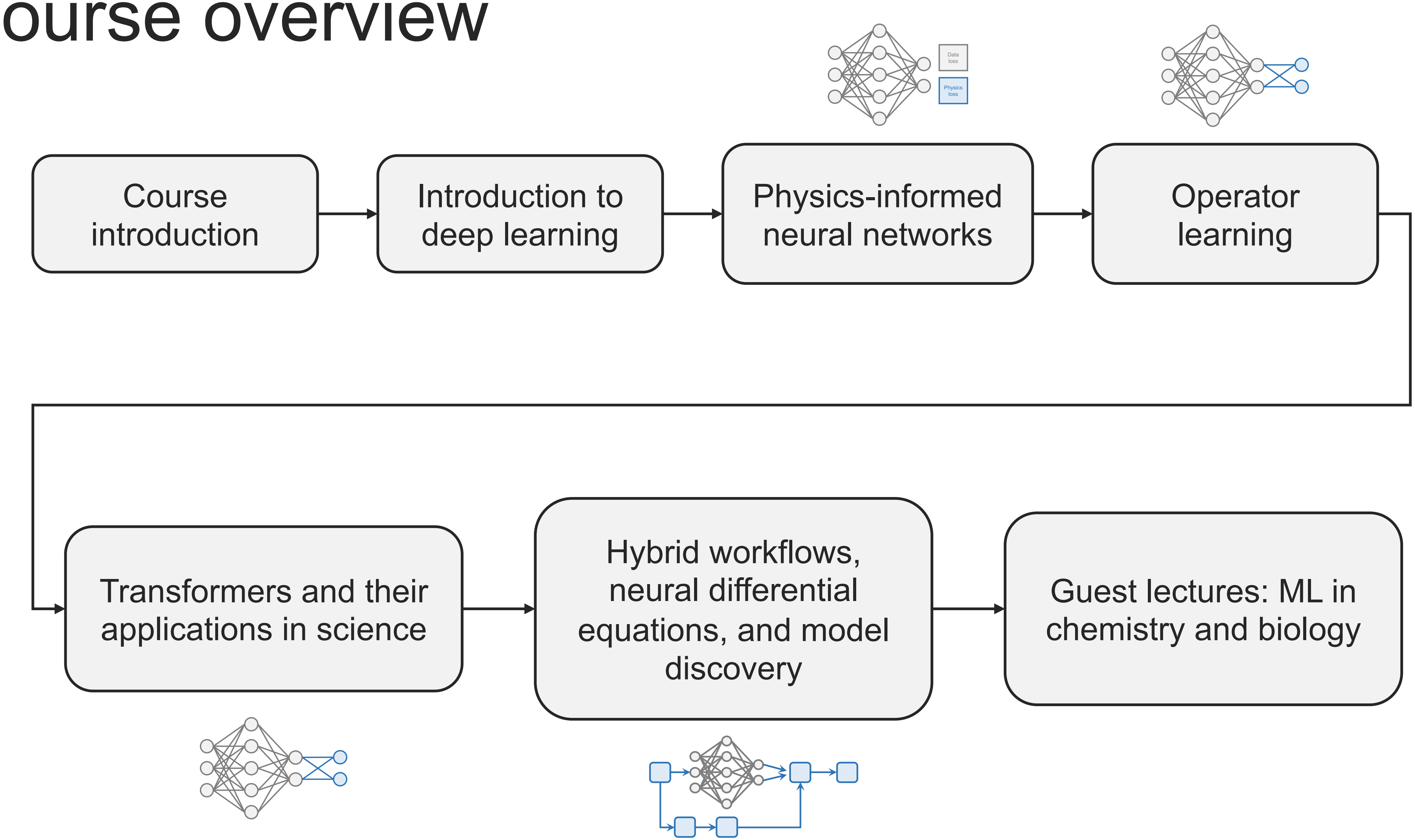
- We can **prune** the search space by using **domain-specific** constraints

- Many different pruning strategies and search algorithms exist

Course learning objectives

- Aware of advanced **applications** of AI in the sciences and engineering
- Familiar with the **design**, **implementation**, and **theory** of these algorithms
- Understand the **pros** and **cons** of using AI and deep learning for science
- Understand key scientific machine learning **concepts** and themes

Course overview



Course introduction

Introduction to deep learning

Physics-informed neural networks

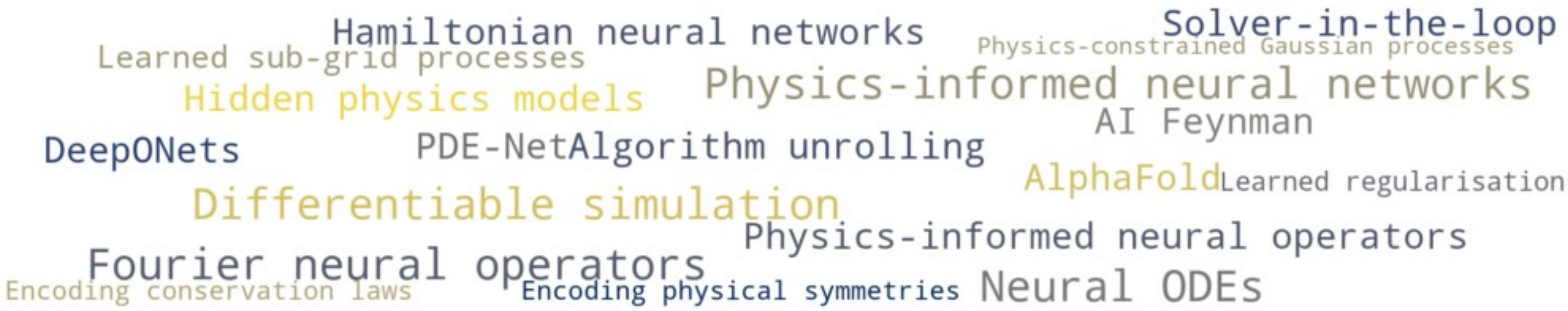
Operator learning

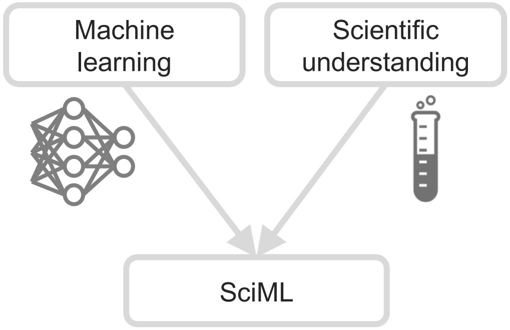
Transformers and their applications in science

Hybrid workflows, neural differential equations, and model discovery

Guest lectures: ML in chemistry and biology

Scientific machine learning


Hamiltonian neural networks
Solver-in-the-loop
Learned sub-grid processes
Physics-constrained Gaussian processes
Hidden physics models
Physics-informed neural networks
AI Feynman
DeepONets
PDE-Net
Algorithm unrolling
AlphaFold
Learned regularisation
Differentiable simulation
Physics-informed neural operators
Fourier neural operators
Encoding conservation laws
Encoding physical symmetries
Neural ODEs



Machine learning

Scientific understanding

SciML

more **powerful, robust, interpretable** models

Some key takeaways

- There are both pros and cons of using deep learning for science
- Incorporating scientific understanding into ML usually **improves** performance
  - There are a plethora of SciML approaches; chose the one which **suits** your problem
  - SciML approaches can be as **flexible** (learnable) or as **inflexible** (unlearnable) as necessary
  - SciML approaches **still** suffer from the limitations of deep neural networks (generalisation, lack of interpretability, optimisation challenges, …)
- AI can be applied to:
  - **many** different problems (simulation, inversion, data assimilation, control, model discovery, …)
  - **many** different fields
- Truly **interdisciplinary** research is required to solve grand challenges in science

Impactful directions

	Search / optimisation	**Representation**
Scientific applications	Inverse problems Model discovery Control …	“Every model is approximate” Finite amount of computing power
AI applications	Planning Reasoning Learning	Hierarchical representations Abstract features and concepts