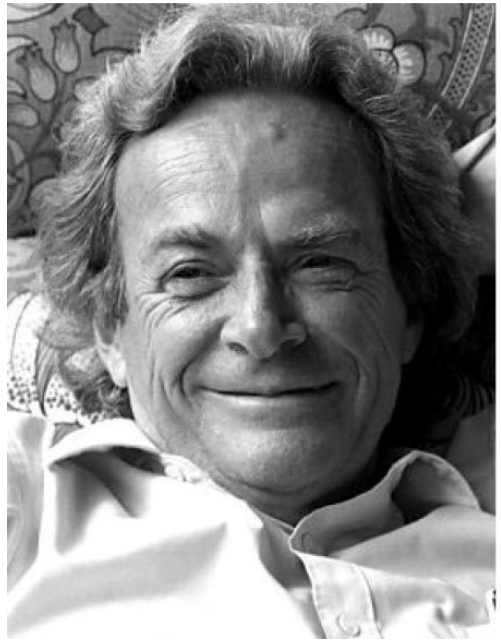
Steps toward building an AI physicist

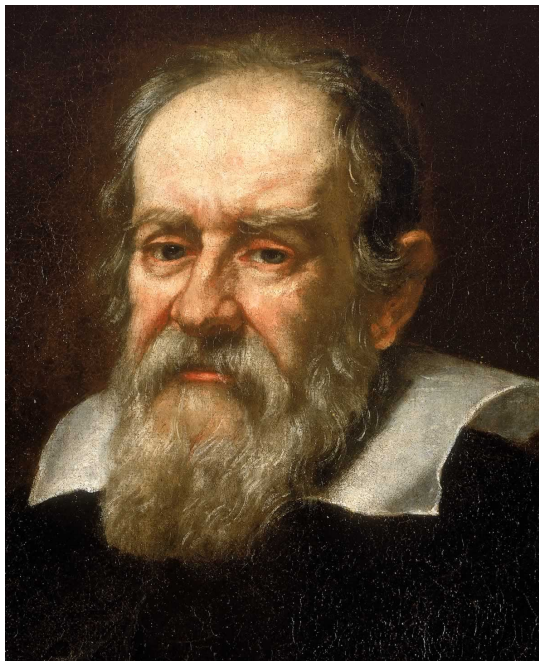


$$L = \frac{\hbar\omega^3}{\pi^2 c^2 (e^{\hbar\omega/k_b T} - 1)}$$

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

$$r = \frac{a(1 - e^2)}{1 + e\cos(\theta_1 - \theta_2)}$$

$$\frac{d\sigma}{d\cos\theta} = \frac{\pi\alpha^2\hbar^2}{m^2c^2}\left(\frac{\omega'}{\omega}\right)^2\left(\frac{\omega'}{\omega} + \frac{\omega}{\omega'} - \sin^2\theta\right)$$



Max Tegmark

Massachusetts Institute of Technology

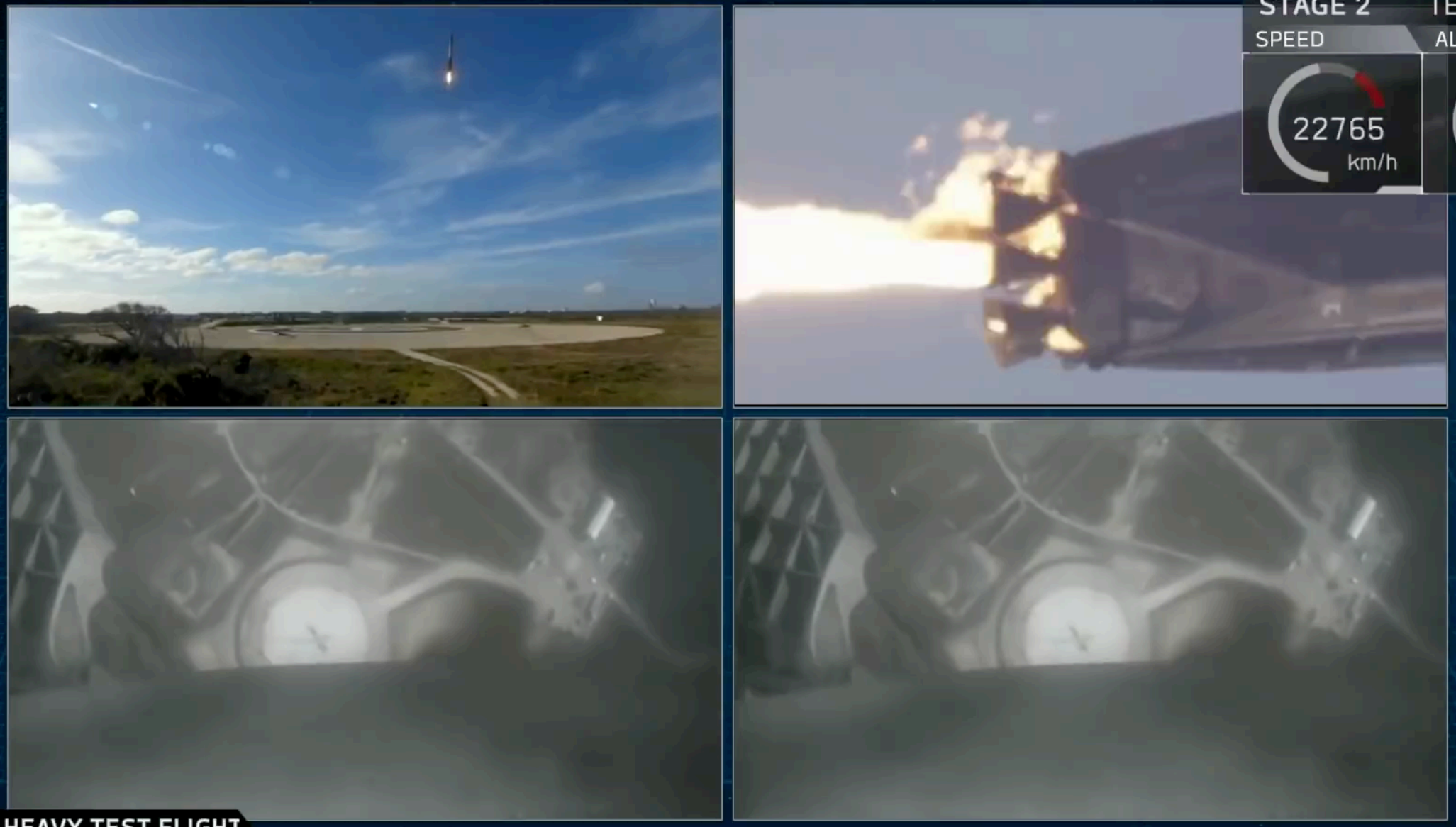
CENTER FOR Brains Minds+ Machines

AI for physics

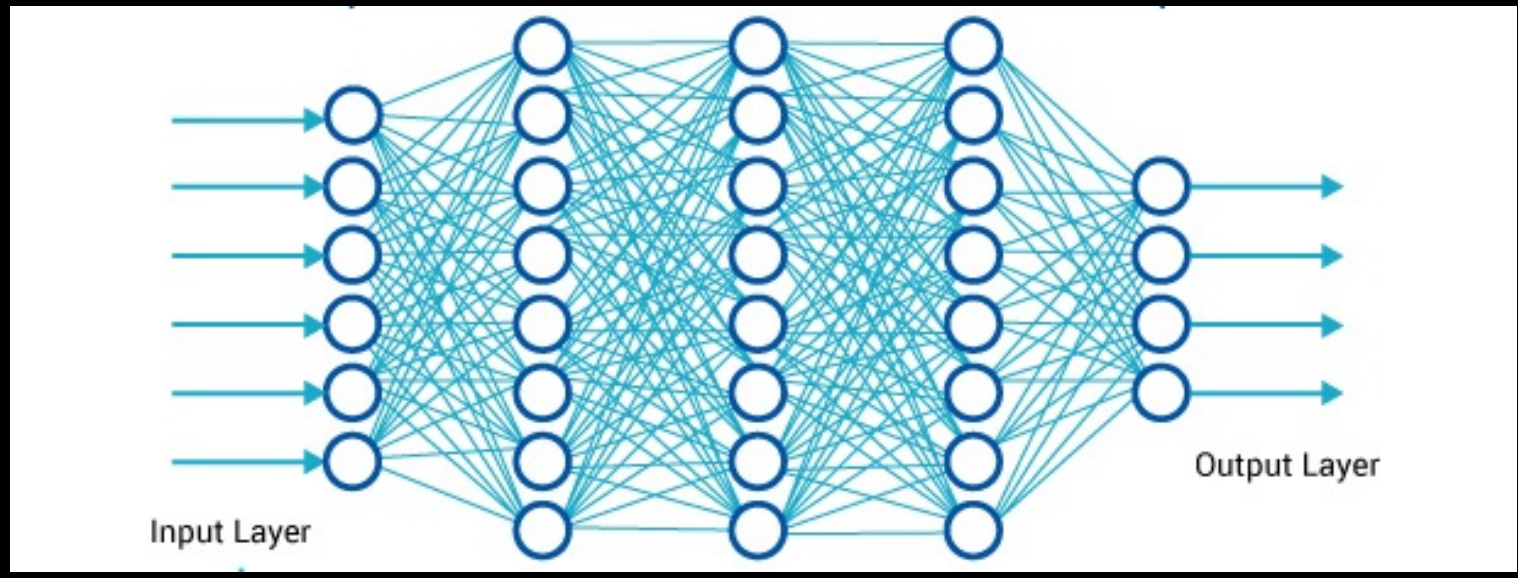
Physics for AI

AI OPPORTUNITIES

Boston Dynamics: Atlas


T+ 00:07:58
STAGE 2 TELEMETRY
SPEED
ALTITUDE
22765
km/h
178
km
FALCON HEAVY TEST FLIGHT
SpaceX

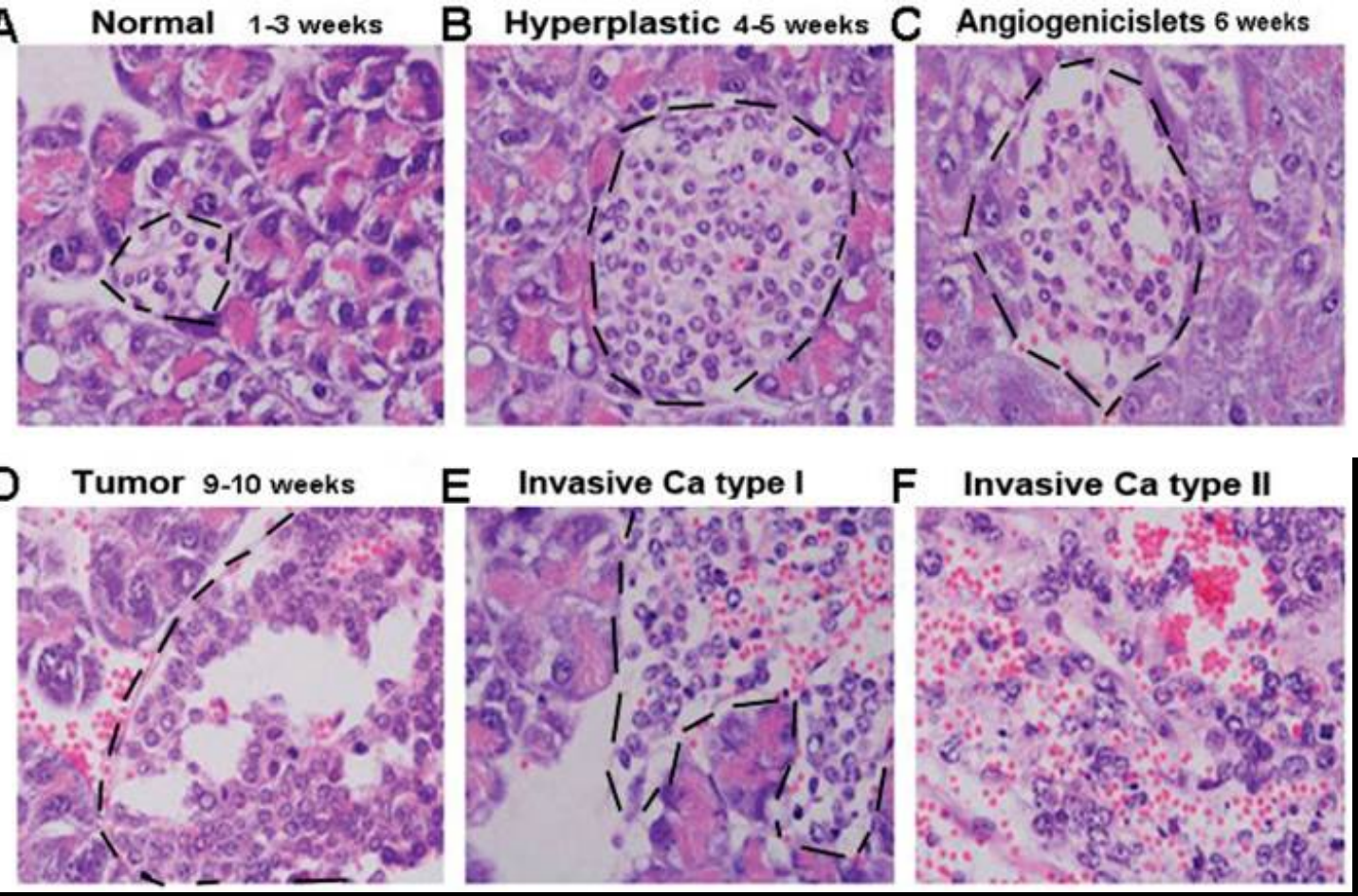
AI diagnosis - prostate cancer

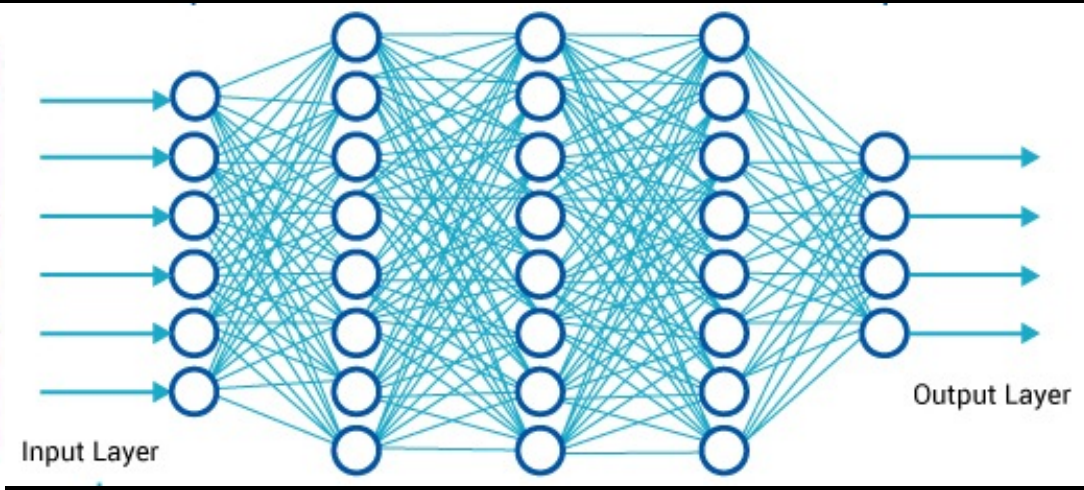


A recent Dutch study showed that AI diagnosis of prostate cancer using MRI was as good as that of human radiologists

Source: Litjens et al 2015, Eur Radiol. **25,** 3187–3199, http://tinyurl.com/prostate-ai

AI diagnosis - lung cancer

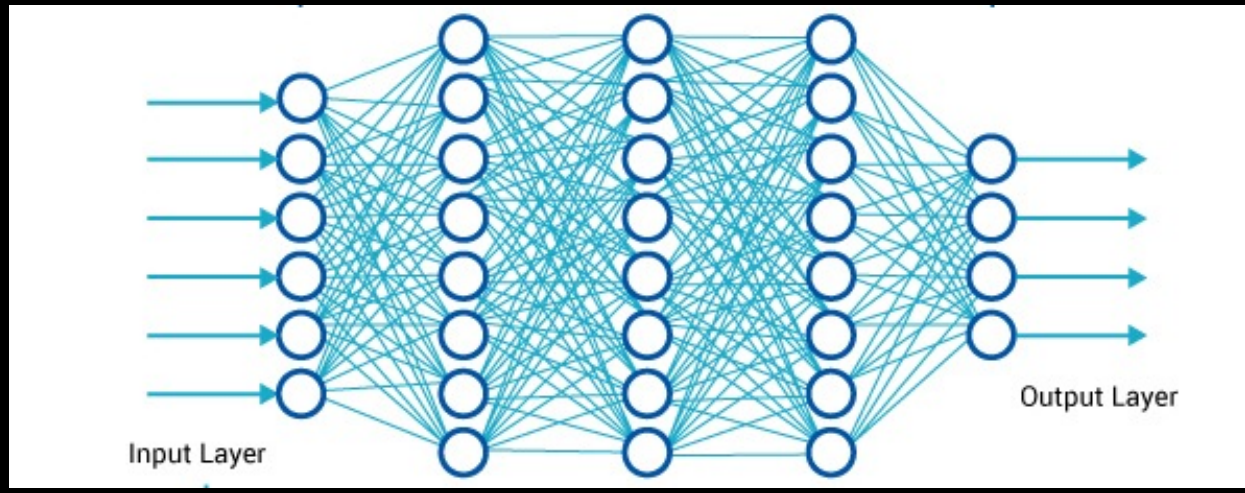




A recent Stanford study showed that AI could diagnose lung cancer using microscope images even better than human pathologists

Source: Snyder et al 2016, http://tinyurl.com/lungcancer-ai

AI diagnosis - blindness

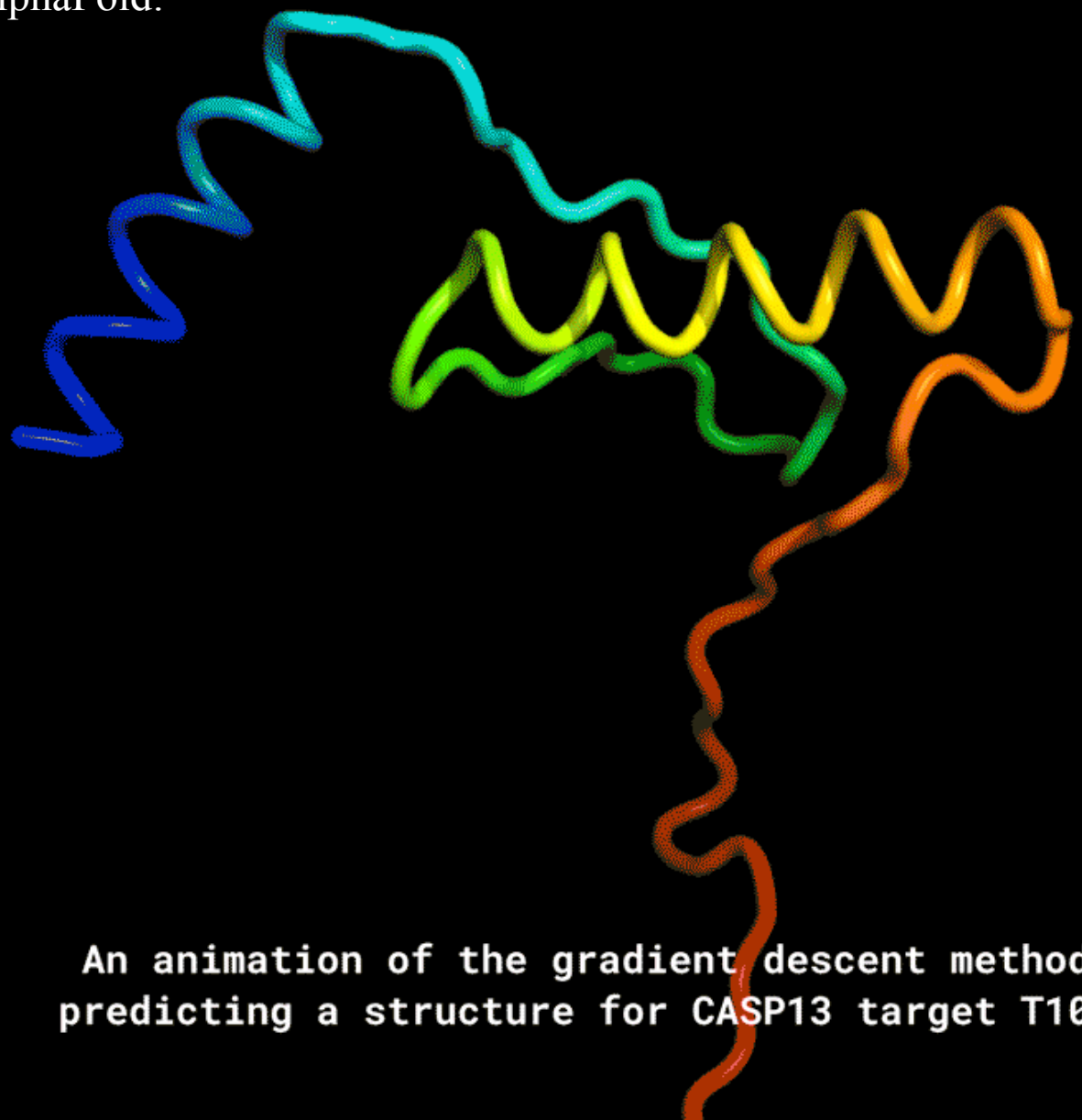


A recent Google DeepMind study showed that AI could diagnose 50 retinal diseases from optical coherence tomography as human optalmologists

Source: De Fauw et al 2018, *Nature Medicine*, **24**, 1342–1350

Input amino acid sequence:
TDELLERLRQLFEELHERGTEIVVEVHINGERDEIRVRNISKEELKKLLERIREKIEREGSSEVEVNVHSGGQTWTFNEK

Protein shape predicted by AlphaFold:

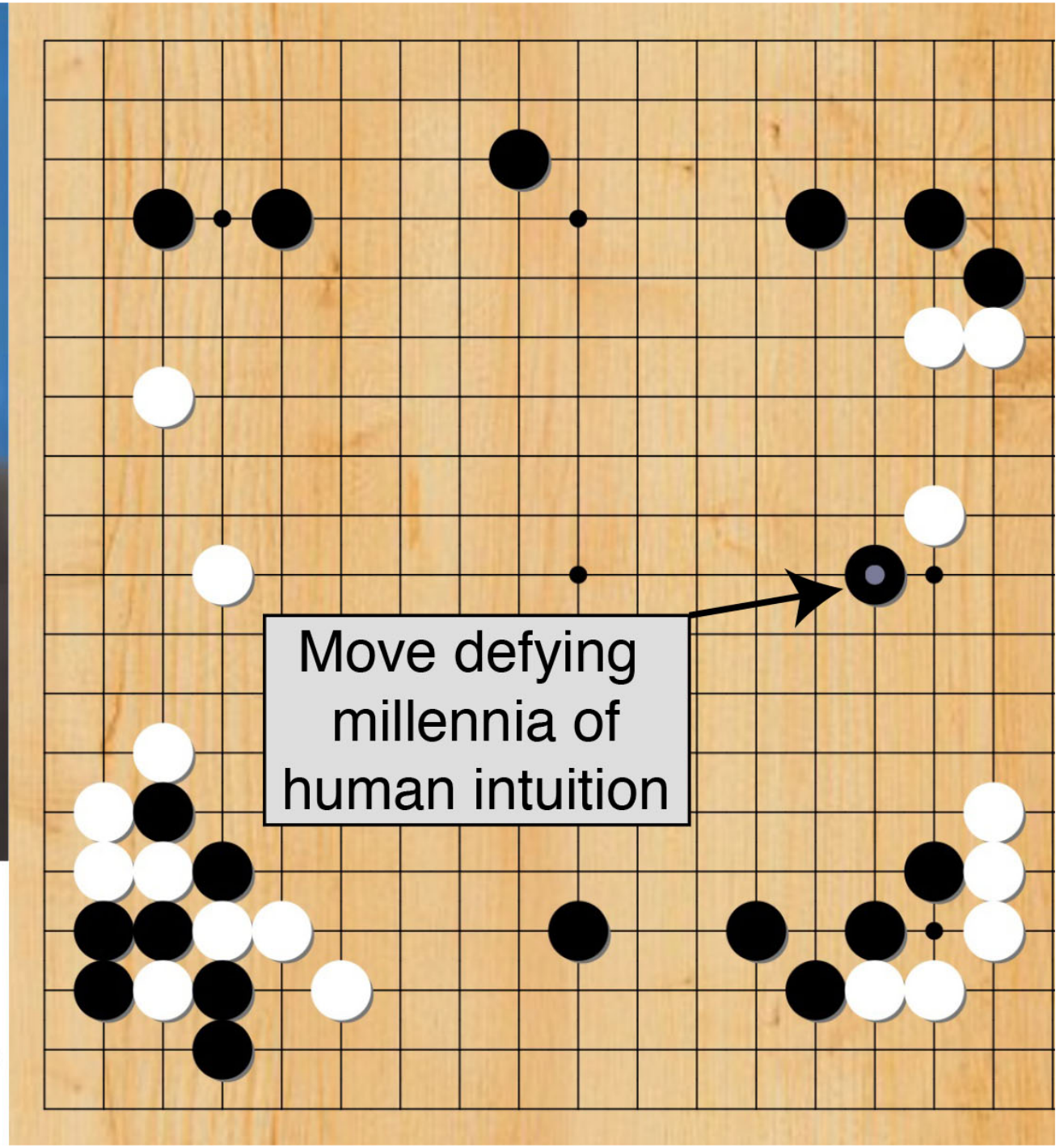


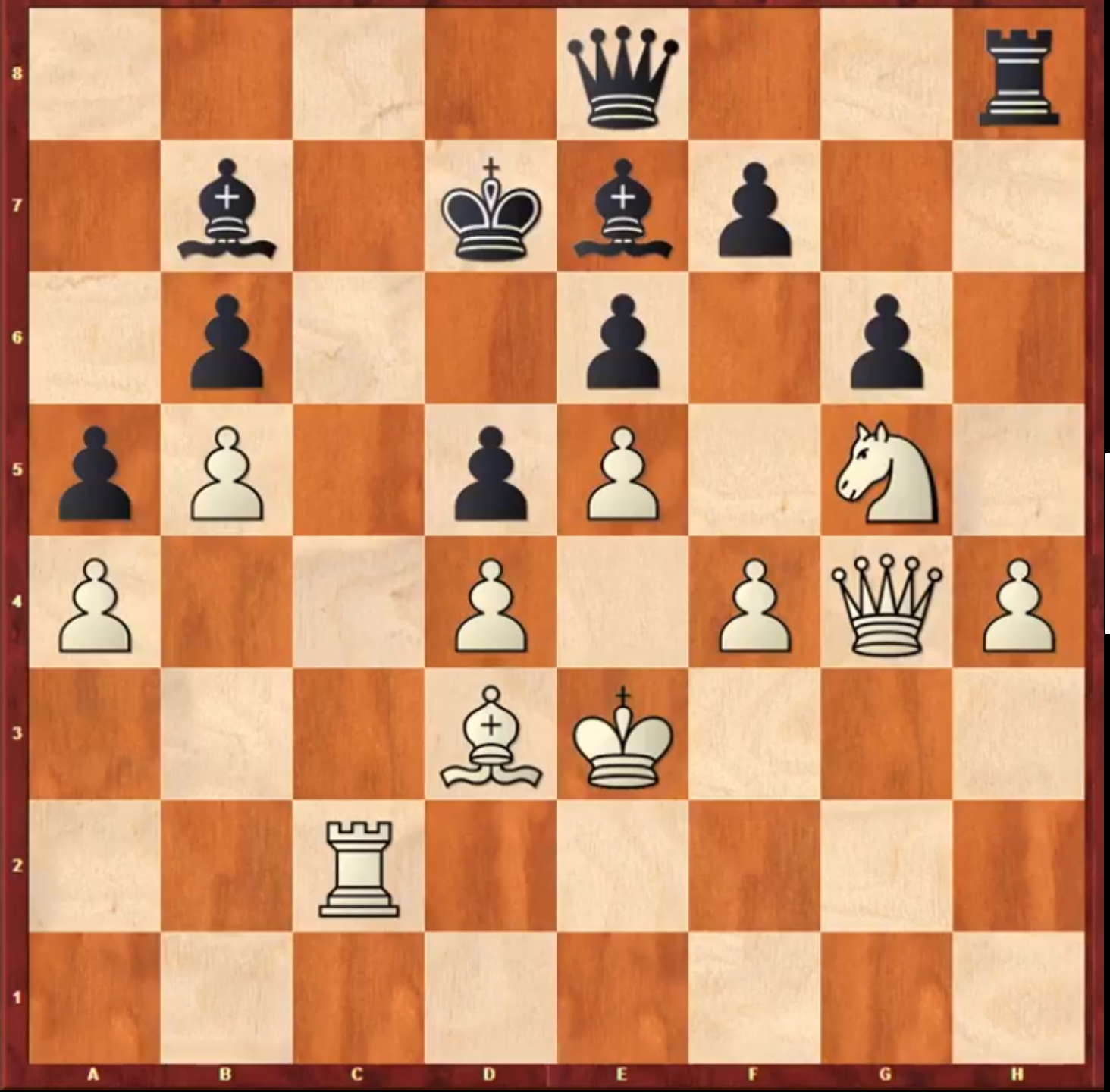
An animation of the gradient descent method
predicting a structure for CASP13 target T1008

DeepMind

vs.

AlphaGo





Google's AlphaZero Destroys Stockfish In 100-Game Match

FM MikeKlein
Dec 6, 2017, 12:50 PM | 349 | Chess Event Coverage

English

Chess changed forever today. And maybe the rest of the world did, too.

AI CHALLENGES

Bernard Parker, left, was rated high risk; Dylan Fugett was rated low risk. (Josh Ritchie for ProPublica)

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica

May 23, 2016

How a ‘confused’ AI may have fought pilots attempting to save Boeing 737 MAX8s

Two Boeing 737 MAX airliners crashed, killing everyone aboard. Now investigations are zeroing in on one single faulty component.

Jamie Seidel

News Corp Australia Network MARCH 19, 2019 2:24PM

Video Image

The New York Times

Knight Capital Says Trading Glitch Cost It $440 Million

BY NATHANIEL POPPER AUGUST 2, 2012 9:07 AM 356

Runaway Trades Spread Turmoil Across Wall St.

Errant trades from the Knight Capital Group began hitting the New York Stock Exchange almost as soon as the opening bell rang on Wednesday. Brendan McDermid/Reuters

1 of 4

4:01 p.m. | Updated

$10 million a minute.

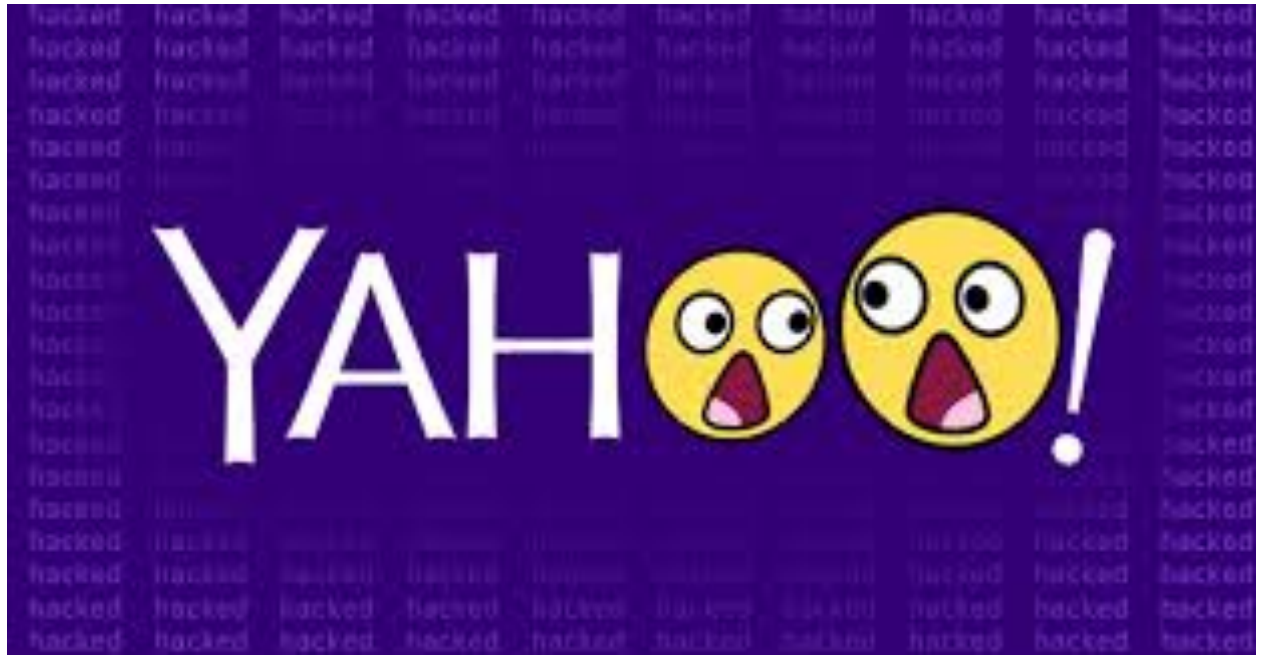
CNN BUSINESS

Every single Yahoo account was hacked - 3 billion in all

by Selena Larson @selenalarson

October 4, 2017: 6:36 AM ET

Timeline: The rise and fall of Yahoo



Sitting down? An epic and historic data breach at Yahoo in August 2013 affected every single customer account that existed at the time, Yahoo parent company Verizon said on Tuesday.

Forbes

3,516 views | May 28, 2019, 08:30am

Equifax Becomes First Firm To See Its Outlook Downgraded Due To A Cyber-Attack

Kate O'Flaherty Contributor ⓘ
Cybersecurity
I'm a freelance cyber security journalist.

KIEV, UKRAINE - 2019/01/17: In this photo illustration, the Equifax Consumer reporting agency company logo seen displayed on a smartphone. (Photo Illustration by Igor Golovniov/SOPA Images/LightRocket via Getty Images) GETTY

GIZMODO

PRIVACY AND SECURITY

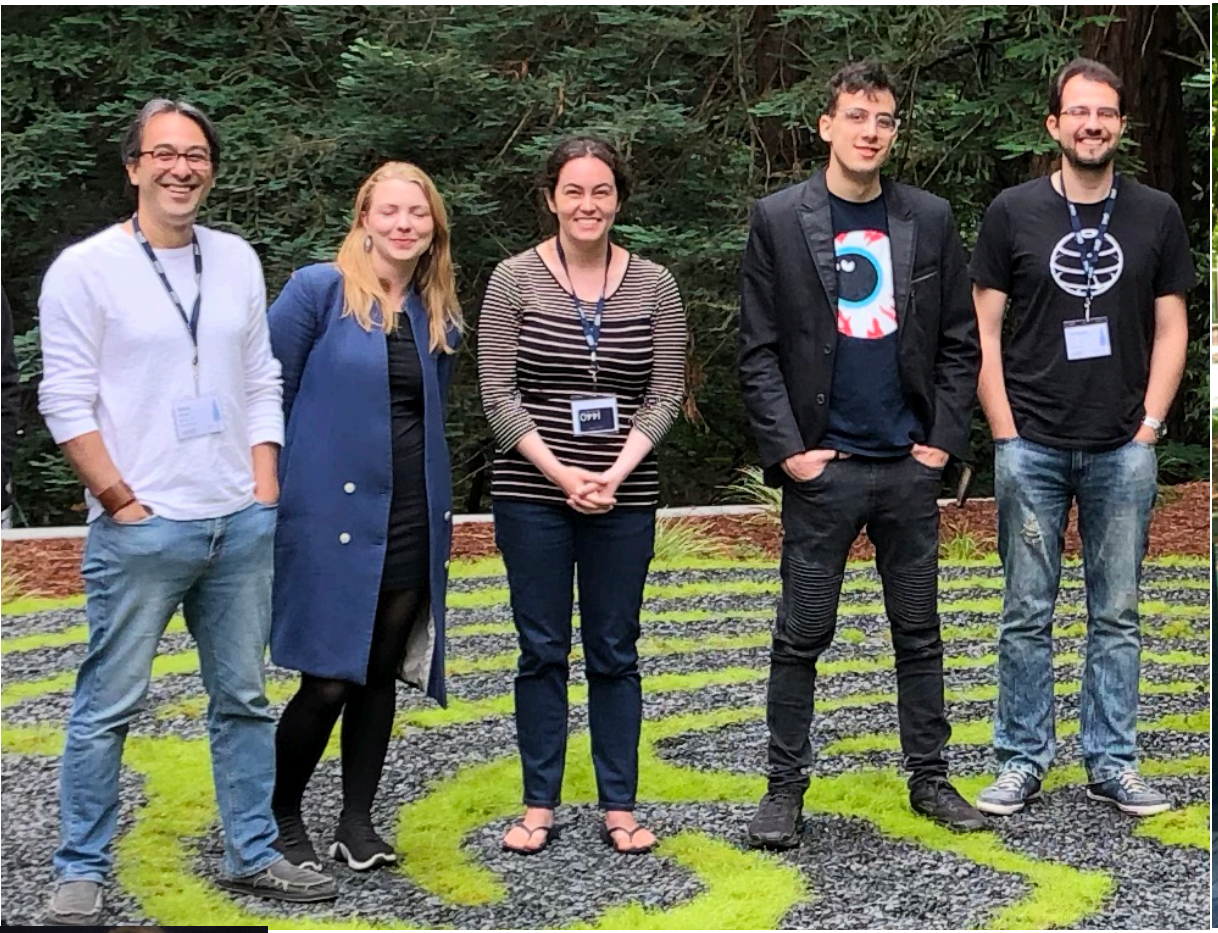
Equifax Is Finally Getting Kicked in the Money Bags Due to Its Disastrous 2017 Hack

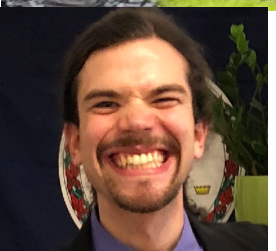
Patrick Howell O'Neill
5/23/19 12:30pm • Filed to: HACKING

20.5K 54 2

Mark Begor, CEO of Equifax, is sworn in during a Senate Homeland Security and Governmental Affairs Committee hearing on Capitol Hill, March 7, 2019 in Washington, DC. The committee heard testimony on investigations examining private sector data breaches.
Photo: Mark Wilson (Getty)

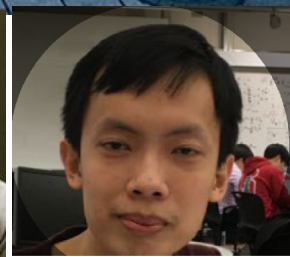
Our focus: Intelligible Intelligence



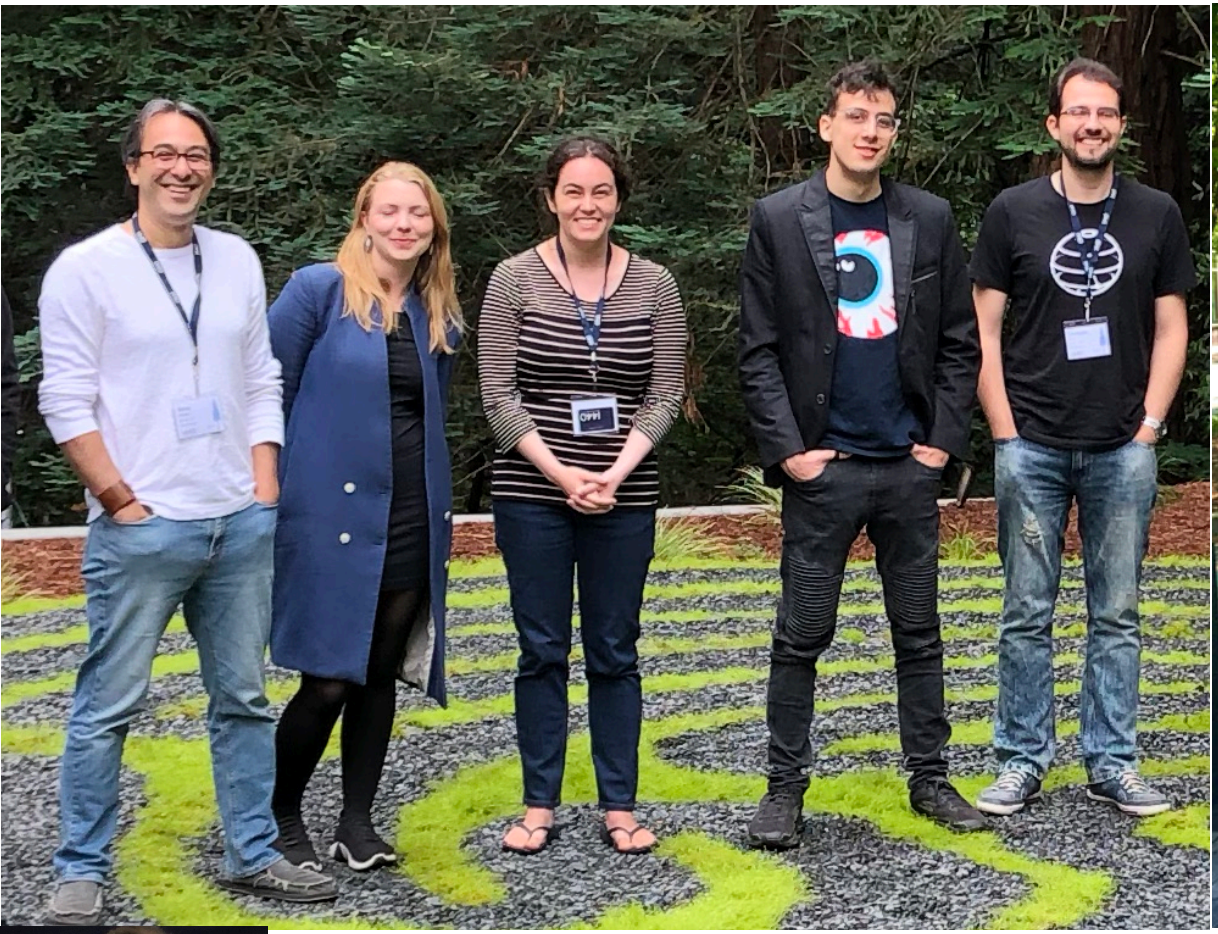


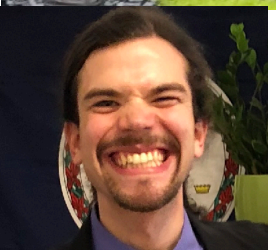
Massachusetts Institute of Technology

CENTER FOR Brains Minds+ Machines

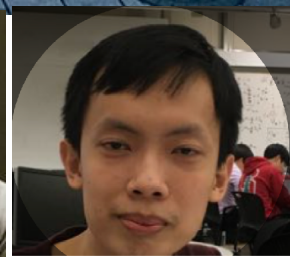


Our focus: Intelligible Intelligence





Risky to install impactful
AI systems that we don't understand!



Toward an AI Physicist for Unsupervised Learning

Tailin Wu, Max Tegmark
Dept. of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139; tailin@mit.edu

(Dated: October 25, 2018)

We investigate opportunities and challenges for improving unsupervised machine learning using four common strategies with a long history in physics: divide-and-conquer, Occam's Razor, unification, and lifelong learning. Instead of using one model to learn everything, we propose a novel paradigm centered around the learning and manipulation of *theories*, which parsimoniously predict both aspects of the future (from past observations) and the domain in which these predictions are accurate. Specifically, we propose a novel generalized-mean-loss to encourage each theory to specialize in its comparatively advantageous domain, and a differentiable description length objective to downweight bad data and "snap" learned theories into simple symbolic formulas. Theories are stored in a "theory hub", which continuously unifies learned theories and can propose theories when encountering new environments. We test our implementation, the "AI Physicist" learning agent, on a suite of increasingly complex physics environments. From unsupervised observation of trajectories through worlds involving random combinations of gravity, electromagnetism, harmonic motion and elastic bounces, our agent typically learns faster and produces mean-squared prediction errors about a billion times smaller than a standard feedforward neural net of comparable complexity, typically recovering integer and rational theory parameters exactly. Our agent successfully identifies domains with different laws of motion also for a nonlinear chaotic double pendulum in a piecewise constant force field.

I. INTRODUCTION

The ability to predict, analyze and parsimoniously model observations is not only central to the scientific endeavor, but also a goal of unsupervised machine learning, which is a key frontier in artificial intelligence (AI) research [1]. Despite impressive recent progress with artificial neural nets, they still get frequently outmatched by human researchers at such modeling, suffering from two drawbacks:

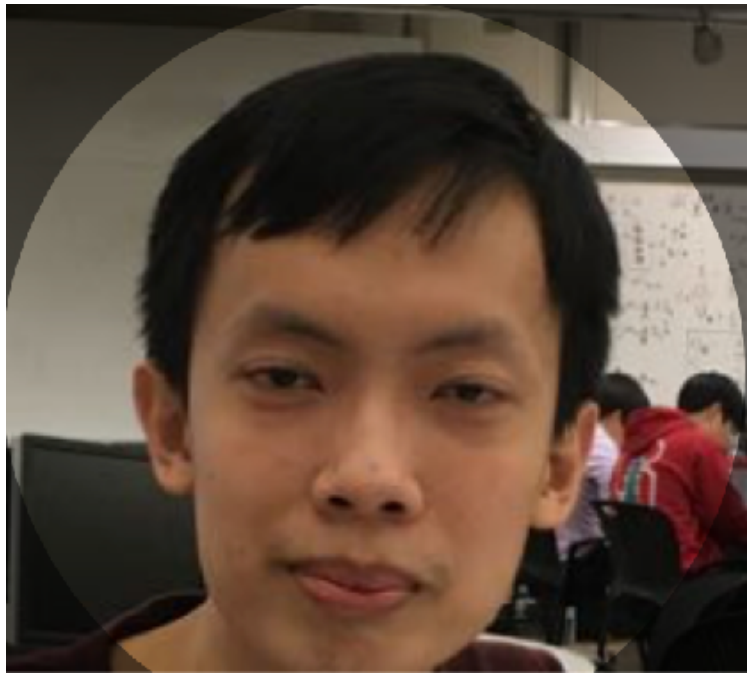
1. Different parts of the data are often generated by different mechanisms in different contexts. A big model that tries to fit all the data in one environment may therefore underperform in a new environment where some mechanisms are replaced by new ones, being inflexible and inefficient at combinatorial generalization [2].

2. Big models are generally hard to interpret, and may

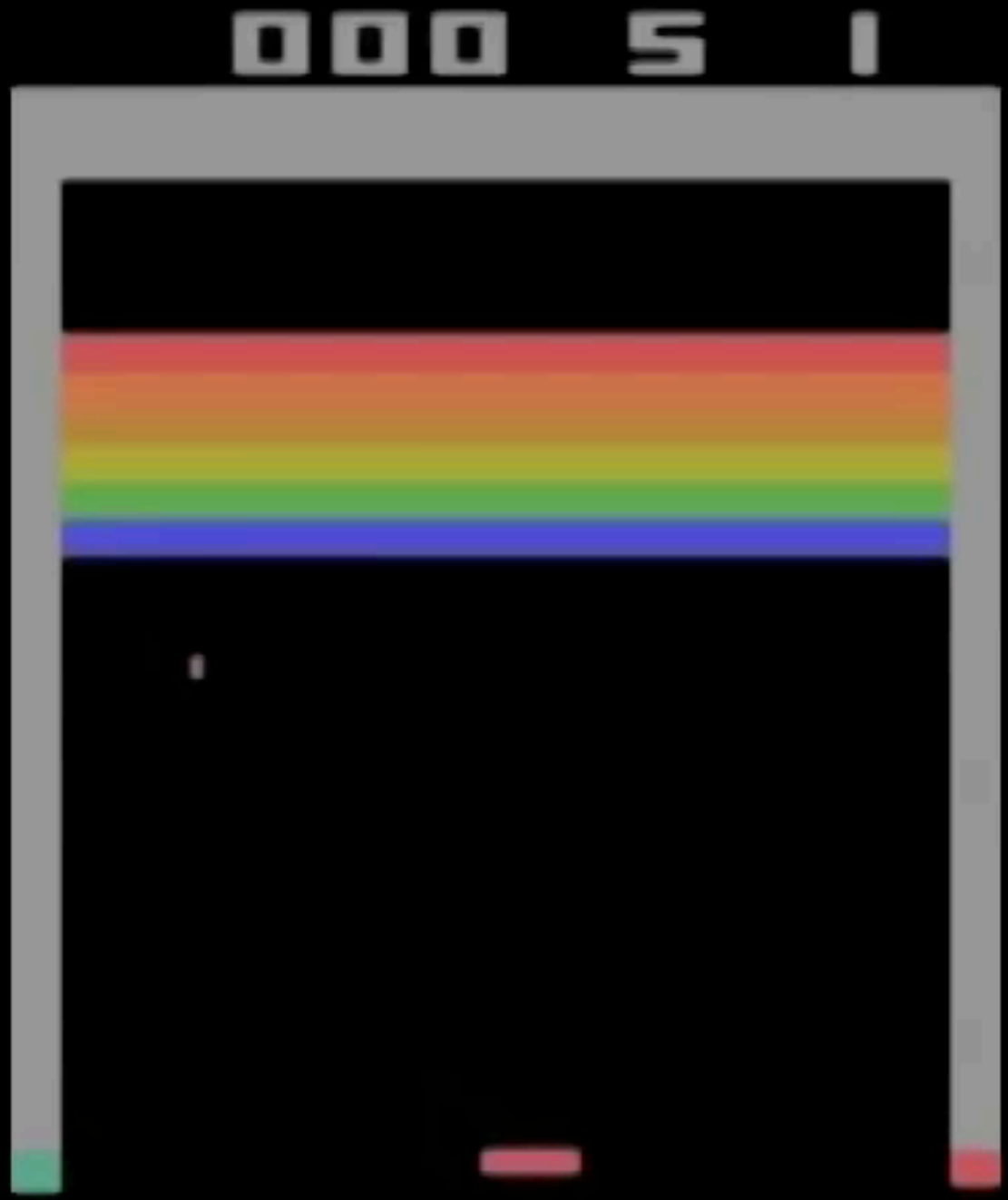
Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit *part* of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems

TABLE I: AI Physicist strategies tested.

are accurate. This suggests an alternative to the standard machine-learning paradigm of fitting a single big model to all the data: instead, learning small theories one by one, and gradually accumulating and organizing them. This paradigm suggests the four specific approaches summarized in Table I, which we combine into a simple "AI Physicist" learning agent: To find individual

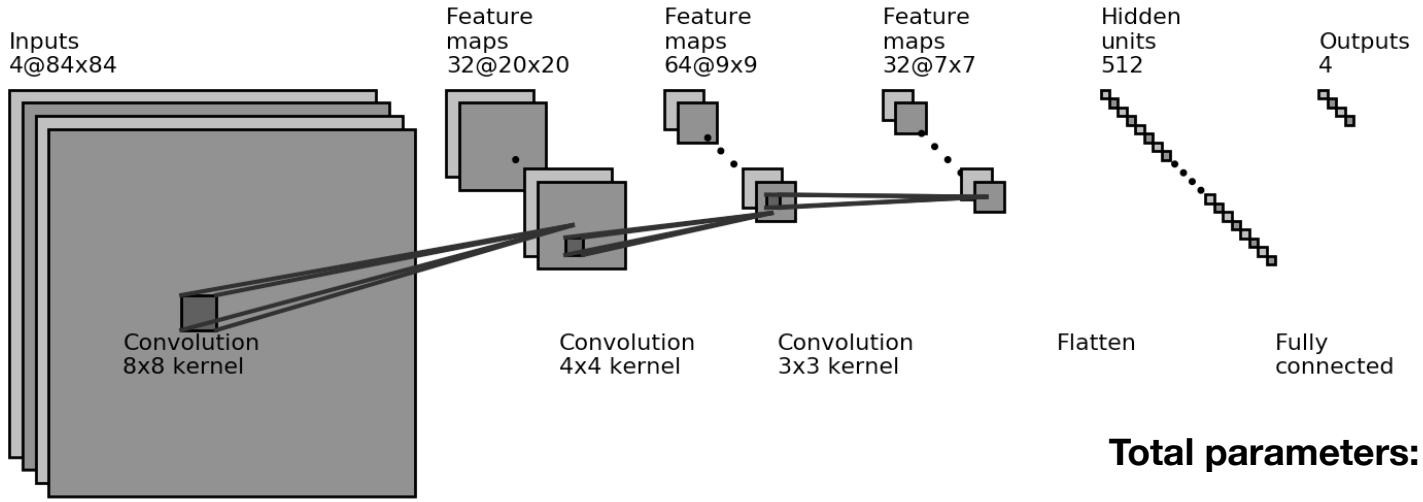


TailinWu



DeepMind

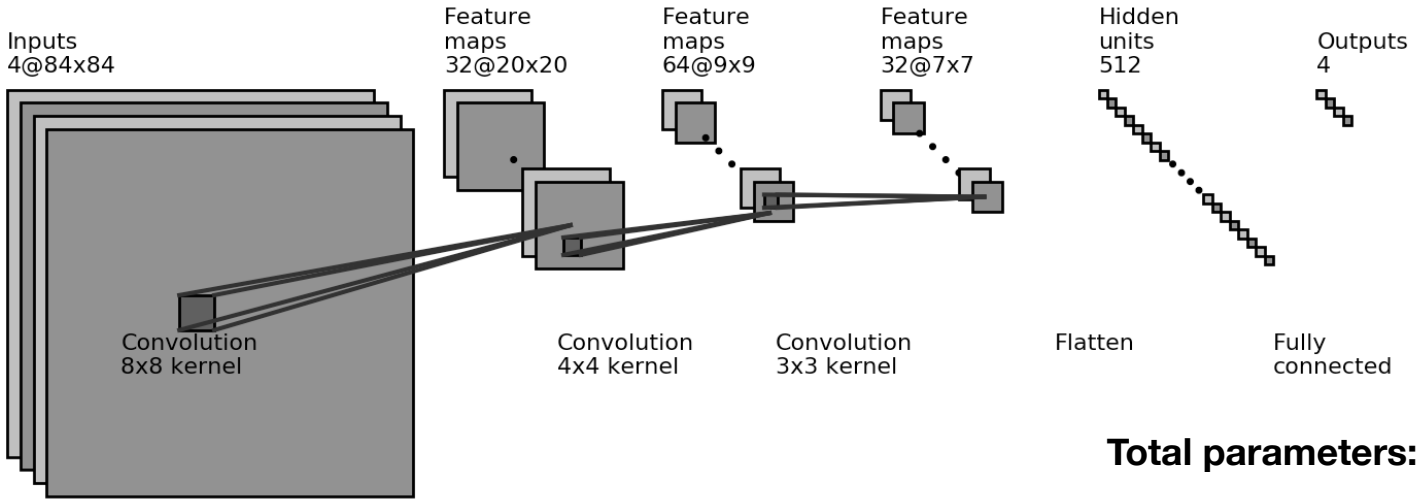
DeepMind



Total parameters: 867488

First kernel on the first layer (0.03% parameters):

```
[[[-0.094, -0.008, -0.037,  0.141,  0.088, -0.095,  0.022, -0.101],
  [-0.085, -0.121, -0.552, -0.538, -0.435,  0.103,  0.109,  0.129],
  [ 0.208, -0.103, -0.6  , -0.743, -0.409,  0.089,  0.025,  0.044],
  [ 0.048,  0.019,  0.074, -0.066, -0.044,  0.023,  0.006, -0.002],
  [ 0.132,  0.089, -0.008,  0.063,  0.019, -0.023,  0.102,  0.11 ],
  [ 0.077,  0.047,  0.098,  0.132, -0.157, -0.072, -0.016,  0.09 ],
  [ 0.242,  0.028, -0.114, -0.041,  0.086, -0.099,  0.093, -0.027],
  [ 0.107, -0.092, -0.094, -0.11 ,  0.045, -0.073, -0.065, -0.017]],

 [[ 0.093, -0.129,  0.126, -0.226,  0.012, -0.359, -0.253, -0.185],
  [ 0.211, -0.199,  0.035,  0.066, -0.146,  0.009, -0.068,  0.021],
  [ 0.135, -0.092, -0.021,  0.048, -0.117, -0.238, -0.073,  0.056],
  [ 0.008,  0.043,  0.187, -0.057, -0.114, -0.142, -0.115,  0.003],
  [ 0.121, -0.258, -0.114, -0.091, -0.131, -0.138, -0.165, -0.134],
  [-0.104, -0.209, -0.309, -0.209, -0.209, -0.168, -0.084, -0.076],
  [ 0.029,  0.07 ,  0.054,  0.072,  0.018, -0.126, -0.057,  0.071],
  [-0.039, -0.064,  0.031,  0.016, -0.062, -0.078, -0.167, -0.113]],

 [[ 0.265,  0.268,  0.451,  0.291,  0.352,  0.279,  0.012,  0.053],
  [ 0.172,  0.268,  0.296,  0.223,  0.399,  0.2  ,  0.022, -0.342],
  [ 0.164,  0.345,  0.273,  0.29 ,  0.2  , -0.094, -0.146, -0.087],
  [ 0.064,  0.094,  0.101,  0.108, -0.004, -0.151, -0.135, -0.169],
  [-0.076,  0.081, -0.087, -0.107, -0.584, -0.284, -0.036, -0.169],
  [ 0.167, -0.025,  0.078, -0.307, -0.457, -0.327, -0.037, -0.091],
  [ 0.088,  0.099,  0.1  , -0.013,  0.126, -0.183,  0.072,  0.007],
  [-0.004,  0.101,  0.262, -0.032, -0.066, -0.19 , -0.278, -0.296]],

 [[-0.217, -0.265, -0.347, -0.321, -0.151,  0.027,  0.164,  0.339],
  [-0.277, -0.093, -0.435, -0.606, -0.858, -0.043,  0.004,  0.069],
  [-0.614,  0.153,  0.162,  0.185,  0.041,  0.075,  0.122,  0.101],
  [-0.477,  0.081,  0.096,  0.019,  0.078, -0.025,  0.071, -0.091],
  [-0.777,  0.118,  0.365,  0.189,  0.149,  0.209,  0.171, -0.019],
  [-0.461,  0.053,  0.399,  0.473,  0.17 ,  0.28 ,  0.132, -0.124],
  [-0.191,  0.073,  0.055,  0.071, -0.092,  0.06 ,  0.082, -0.291],
  [-0.629, -0.416,  0.111,  0.338, -0.111, -0.902, -0.442,  0.023]]]
```



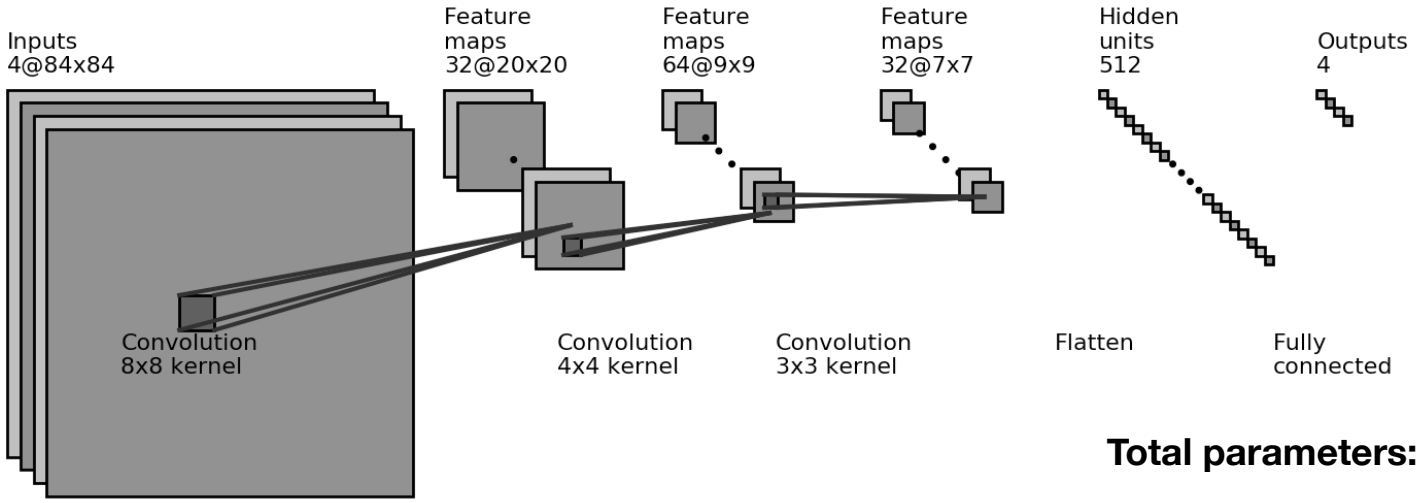
Total parameters: 867488

First kernel on the first layer (0.03% parameters):

```
[[[-0.094, -0.008, -0.037,  0.141,  0.088, -0.095,  0.022, -0.101],
  [-0.085, -0.121, -0.552, -0.538, -0.435,  0.103,  0.109,  0.129],
  [ 0.208, -0.103, -0.6  , -0.743, -0.409,  0.089,  0.025,  0.044],
  [ 0.048,  0.019,  0.074, -0.066, -0.044,  0.023,  0.006, -0.002],
  [ 0.132,  0.089, -0.008,  0.063,  0.019, -0.023,  0.102,  0.11 ],
  [ 0.077,  0.047,  0.098,  0.132, -0.157, -0.072, -0.016,  0.09 ],
  [ 0.242,  0.028, -0.114, -0.041,  0.086, -0.099,  0.093, -0.027],
  [ 0.107, -0.092, -0.094, -0.11 ,  0.045, -0.073, -0.065, -0.017]],

 [[ 0.093, -0.129,  0.126, -0.226,  0.012, -0.359, -0.253, -0.185],
  [ 0.211, -0.199,  0.035,  0.066, -0.146,  0.009, -0.068,  0.021],
  [ 0.135, -0.092, -0.021,  0.048, -0.117, -0.238, -0.073,  0.056],
  [ 0.008,  0.043,  0.187, -0.057, -0.114, -0.142, -0.115,  0.003],
  [ 0.121, -0.258, -0.114, -0.091, -0.131, -0.138, -0.165, -0.134],
  [-0.104, -0.209, -0.309, -0.209, -0.209, -0.168, -0.084, -0.076],
  [ 0.029,  0.07 ,  0.054,  0.072,  0.018, -0.126, -0.057,  0.071],
  [-0.039, -0.064,  0.031,  0.016, -0.062, -0.078, -0.167, -0.113]],

 [[ 0.265,  0.268,  0.451,  0.291,  0.352,  0.279,  0.012,  0.053],
  [ 0.172,  0.268,  0.296,  0.223,  0.399,  0.2  ,  0.022, -0.342],
  [ 0.164,  0.345,  0.273,  0.29 ,  0.2  , -0.094, -0.146, -0.087],
  [ 0.064,  0.094,  0.101,  0.108, -0.004, -0.151, -0.135, -0.169],
  [-0.076,  0.081, -0.087, -0.107, -0.584, -0.284, -0.036, -0.169],
  [ 0.167, -0.025,  0.078, -0.307, -0.457, -0.327, -0.037, -0.091],
  [ 0.088,  0.099,  0.1  , -0.013,  0.126, -0.183,  0.072,  0.007],
  [-0.004,  0.101,  0.262, -0.032, -0.066, -0.19 , -0.278, -0.296]],

 [[-0.217, -0.265, -0.347, -0.321, -0.151,  0.027,  0.164,  0.339],
  [-0.277, -0.093, -0.435, -0.606, -0.858, -0.043,  0.004,  0.069],
  [-0.614,  0.153,  0.162,  0.185,  0.041,  0.075,  0.122,  0.101],
  [-0.477,  0.081,  0.096,  0.019,  0.078, -0.025,  0.071, -0.091],
  [-0.777,  0.118,  0.365,  0.189,  0.149,  0.209,  0.171, -0.019],
  [-0.461,  0.053,  0.399,  0.473,  0.17 ,  0.28 ,  0.132, -0.124],
  [-0.191,  0.073,  0.055,  0.071, -0.092,  0.06 ,  0.082, -0.291],
  [-0.629, -0.416,  0.111,  0.338, -0.111, -0.902, -0.442,  0.023]]]
```

Preferable:

INTELLIGIBLE INTELLIGENCE



Total parameters: 867488

First kernel on the first layer (0.03% parameters):

```
[[[-0.094, -0.008, -0.037,  0.141,  0.088, -0.095,  0.022, -0.101],
  [-0.085, -0.121, -0.552, -0.538, -0.435,  0.103,  0.109,  0.129],
  [ 0.208, -0.103, -0.6  , -0.743, -0.409,  0.089,  0.025,  0.044],
  [ 0.048,  0.019,  0.074, -0.066, -0.044,  0.023,  0.006, -0.002],
  [ 0.132,  0.089, -0.008,  0.063,  0.019, -0.023,  0.102,  0.11 ],
  [ 0.077,  0.047,  0.098,  0.132, -0.157, -0.072, -0.016,  0.09 ],
  [ 0.242,  0.028, -0.114, -0.041,  0.086, -0.099,  0.093, -0.027],
  [ 0.107, -0.092, -0.094, -0.11 ,  0.045, -0.073, -0.065, -0.017]],

 [[ 0.093, -0.129,  0.126, -0.226,  0.012, -0.359, -0.253, -0.185],
  [ 0.211, -0.199,  0.035,  0.066, -0.146,  0.009, -0.068,  0.021],
  [ 0.135, -0.092, -0.021,  0.048, -0.117, -0.238, -0.073,  0.056],
  [ 0.008,  0.043,  0.187, -0.057, -0.114, -0.142, -0.115,  0.003],
  [ 0.121, -0.258, -0.114, -0.091, -0.131, -0.138, -0.165, -0.134],
  [-0.104, -0.209, -0.309, -0.209, -0.209, -0.168, -0.084, -0.076],
  [ 0.029,  0.07 ,  0.054,  0.072,  0.018, -0.126, -0.057,  0.071],
  [-0.039, -0.064,  0.031,  0.016, -0.062, -0.078, -0.167, -0.113]],

 [[ 0.265,  0.268,  0.451,  0.291,  0.352,  0.279,  0.012,  0.053],
  [ 0.172,  0.268,  0.296,  0.223,  0.399,  0.2  ,  0.022, -0.342],
  [ 0.164,  0.345,  0.273,  0.29 ,  0.2  , -0.094, -0.146, -0.087],
  [ 0.064,  0.094,  0.101,  0.108, -0.004, -0.151, -0.135, -0.169],
  [-0.076,  0.081, -0.087, -0.107, -0.584, -0.284, -0.036, -0.169],
  [ 0.167, -0.025,  0.078, -0.307, -0.457, -0.327, -0.037, -0.091],
  [ 0.088,  0.099,  0.1  , -0.013,  0.126, -0.183,  0.072,  0.007],
  [-0.004,  0.101,  0.262, -0.032, -0.066, -0.19 , -0.278, -0.296]],

 [[-0.217, -0.265, -0.347, -0.321, -0.151,  0.027,  0.164,  0.339],
  [-0.277, -0.093, -0.435, -0.606, -0.858, -0.043,  0.004,  0.069],
  [-0.614,  0.153,  0.162,  0.185,  0.041,  0.075,  0.122,  0.101],
  [-0.477,  0.081,  0.096,  0.019,  0.078, -0.025,  0.071, -0.091],
  [-0.777,  0.118,  0.365,  0.189,  0.149,  0.209,  0.171, -0.019],
  [-0.461,  0.053,  0.399,  0.473,  0.17 ,  0.28 ,  0.132, -0.124],
  [-0.191,  0.073,  0.055,  0.071, -0.092,  0.06 ,  0.082, -0.291],
  [-0.629, -0.416,  0.111,  0.338, -0.111, -0.902, -0.442,  0.023]]]
```

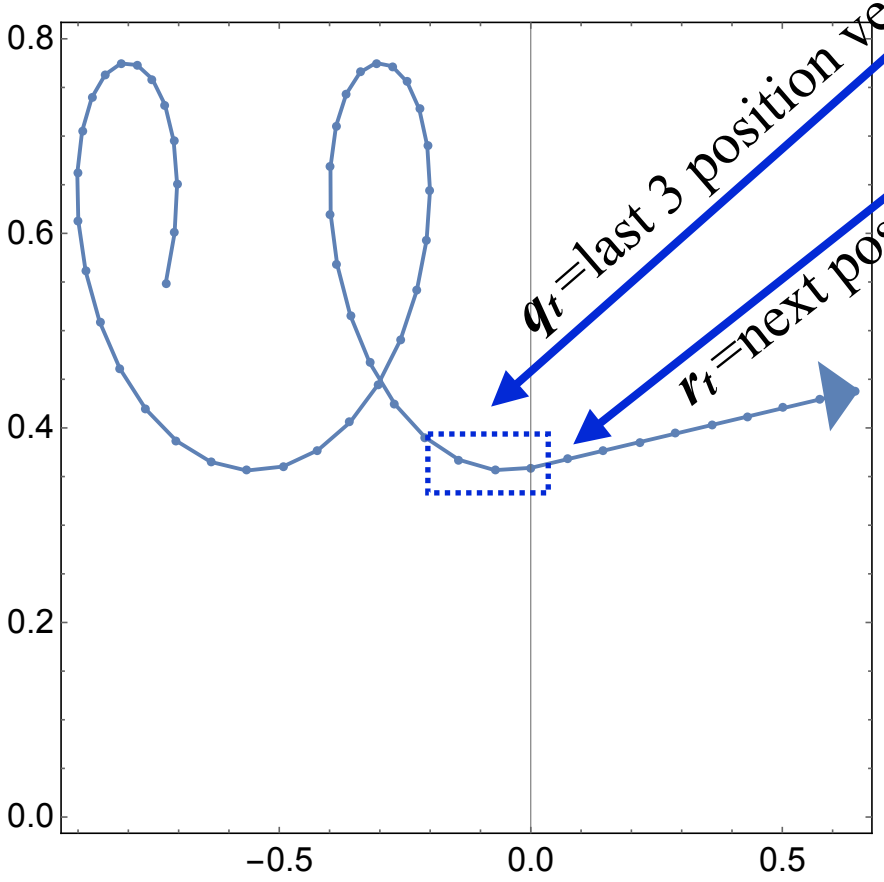
Claim:

The power of deep learning comes not from inscrutability but from differentiability

Problem: Predict next point r_t from past points

Standard ML solution: Train feed-forward neural network

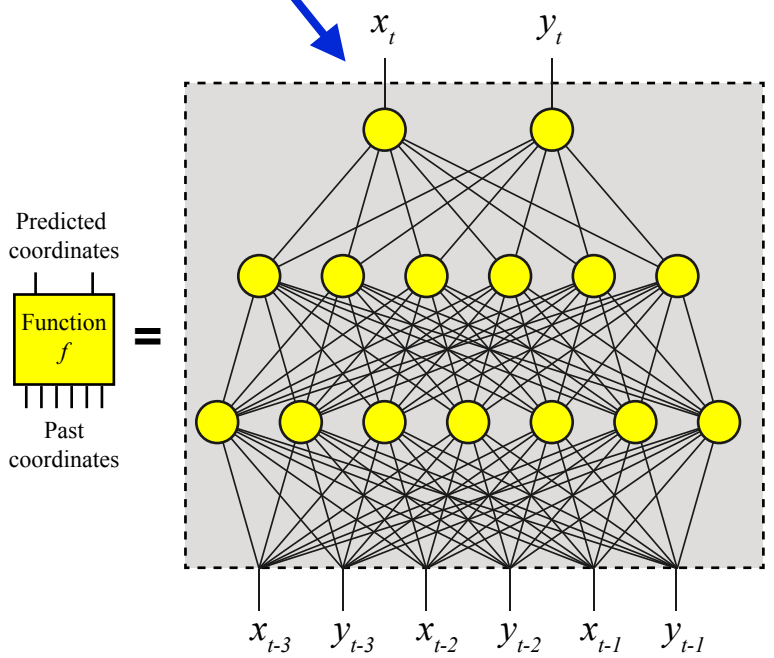
Minimize loss $L = \sum_t |f(\mathbf{q}_t) - \mathbf{r}_t|^2$



q_t=last 3 position vectors

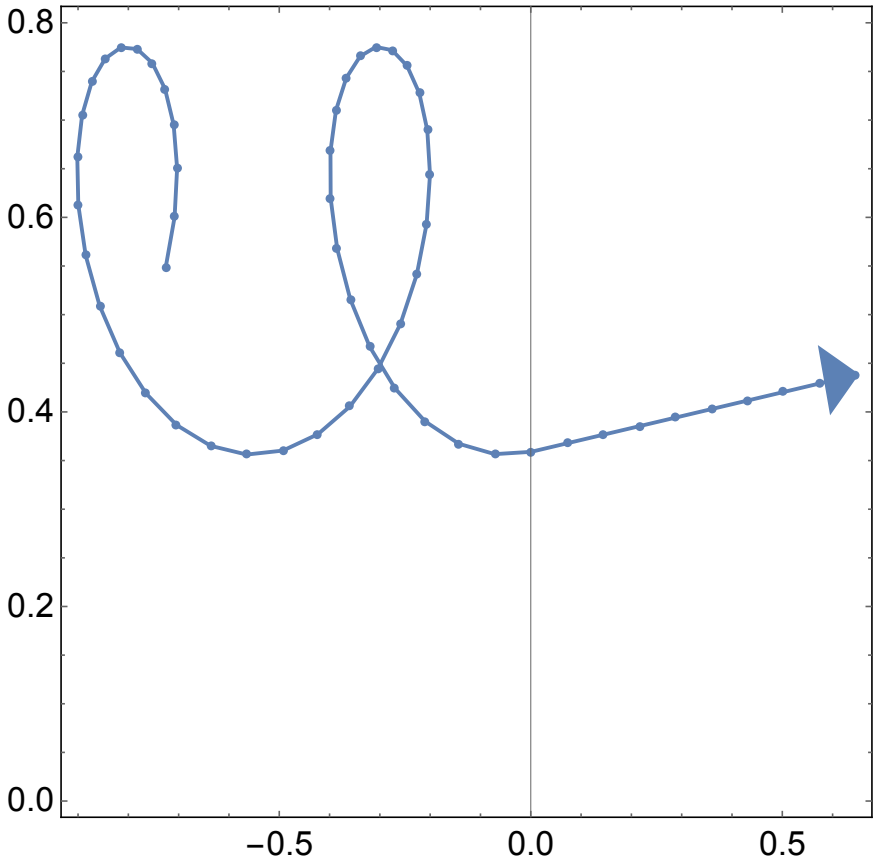
r_t=next position vector

f implemented by neural network with parameters p



Problem: Predict next point r_t from past points
Standard ML solution: Train feed-forward neural network
Result: Works OK, but would be nice if

- more accurate

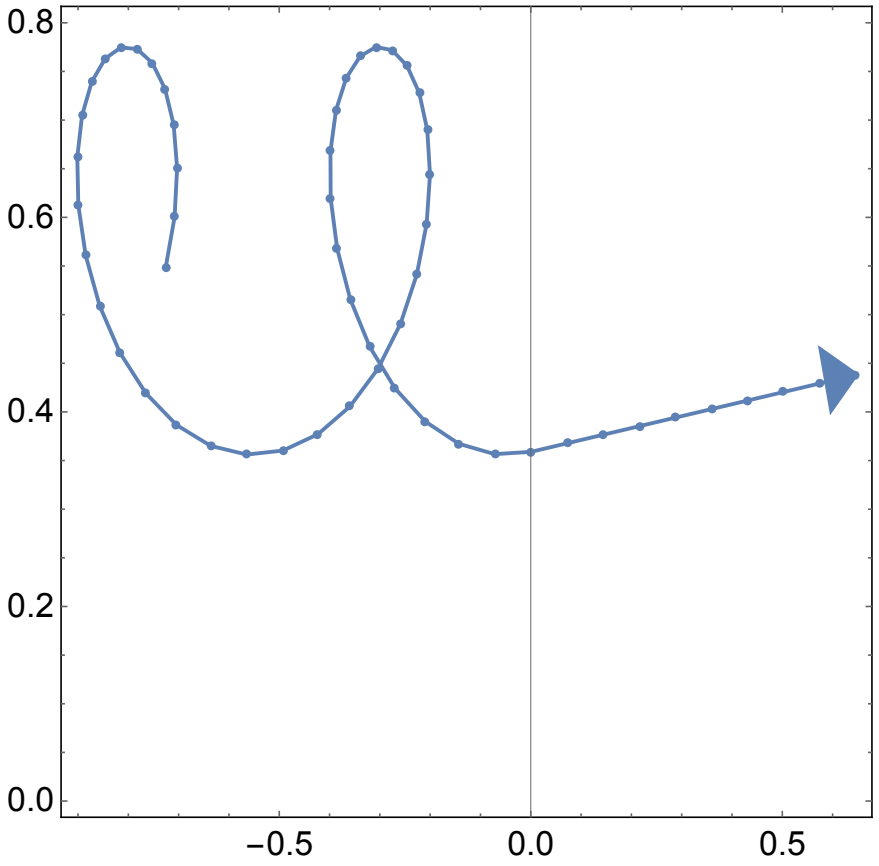


Problem: Predict next point r_t from past points

Standard ML solution: Train feed-forward neural network

Result: Works OK, but would be nice if

- more accurate
- more intelligible

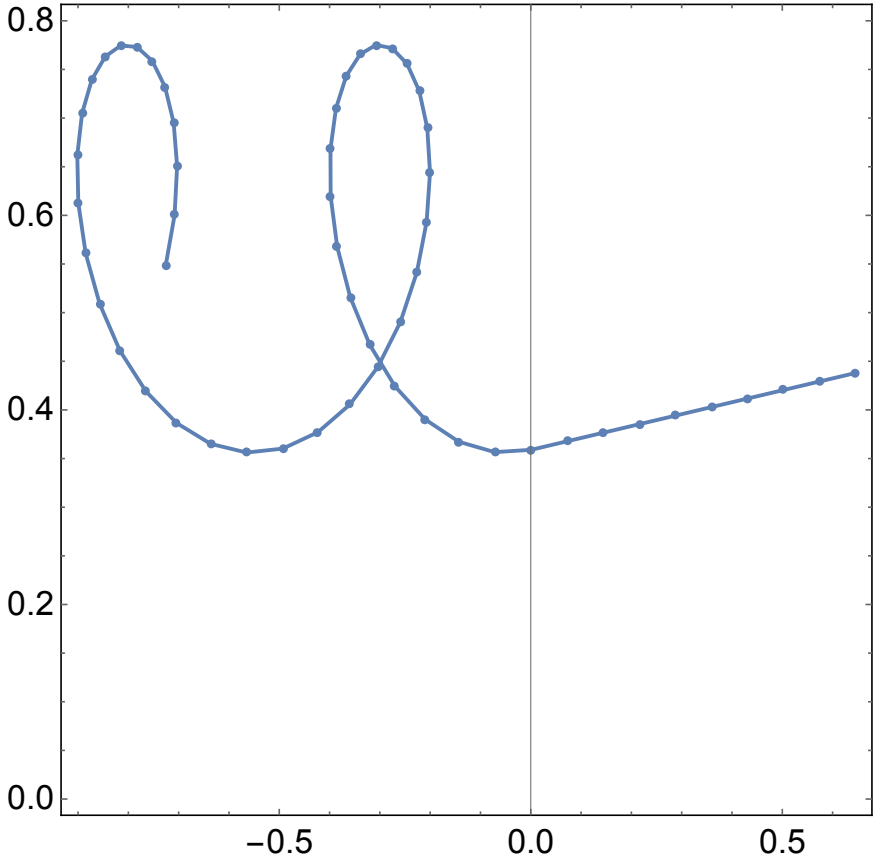


Here's the "explanation":

```
[[[-0.094, -0.008, -0.037,  0.141,  0.088, -0.095,  0.022, -0.101],
  [-0.085, -0.121, -0.552, -0.538, -0.435,  0.103,  0.109,  0.129],
  [ 0.208, -0.103, -0.6  , -0.743, -0.409,  0.089,  0.025,  0.044],
  [ 0.048,  0.019,  0.074, -0.066, -0.044,  0.023,  0.006, -0.002],
  [ 0.132,  0.089, -0.008,  0.063,  0.019, -0.023,  0.102,  0.11 ],
  [ 0.077,  0.047,  0.098,  0.132, -0.157, -0.072, -0.016,  0.09 ],
  [ 0.242,  0.028, -0.114, -0.041,  0.086, -0.099,  0.093, -0.027],
  [ 0.107, -0.092, -0.094, -0.11 ,  0.045, -0.073, -0.065, -0.017]],

 [[ 0.093, -0.129,  0.126, -0.226,  0.012, -0.359, -0.253, -0.185],
  [ 0.211, -0.199,  0.035,  0.066, -0.146,  0.009, -0.068,  0.021],
  [ 0.135, -0.092, -0.021,  0.048, -0.117, -0.238, -0.073,  0.056],
  [ 0.008,  0.043,  0.187, -0.057, -0.114, -0.142, -0.115,  0.003],
  [ 0.121, -0.258, -0.114, -0.091, -0.131, -0.138, -0.165, -0.134],
  [-0.104, -0.209, -0.309, -0.209, -0.209, -0.168, -0.084, -0.076],
  [ 0.029,  0.07 ,  0.054,  0.072,  0.018, -0.126, -0.057,  0.071],
  [-0.039, -0.064,  0.031,  0.016, -0.062, -0.078, -0.167, -0.113]],

 [[ 0.265,  0.268,  0.451,  0.291,  0.352,  0.279,  0.012,  0.053],
  [ 0.172,  0.268,  0.296,  0.223,  0.399,  0.2  ,  0.022, -0.342],
  [ 0.164,  0.345,  0.273,  0.29 ,  0.2  , -0.094, -0.146, -0.087],
  [ 0.064,  0.094,  0.101,  0.108, -0.004, -0.151, -0.135, -0.169],
  [-0.076,  0.081, -0.087, -0.107, -0.584, -0.284, -0.036, -0.169],
  [ 0.167, -0.025,  0.078, -0.307, -0.457, -0.327, -0.037, -0.091],
  [ 0.088,  0.099,  0.1  , -0.013,  0.126, -0.183,  0.072,  0.007],
  [-0.004,  0.101,  0.262, -0.032, -0.066, -0.19 , -0.278, -0.296]],

 [[-0.217, -0.265, -0.347, -0.321, -0.151,  0.027,  0.164,  0.339],
  [-0.277, -0.093, -0.435, -0.606, -0.858, -0.043,  0.004,  0.069],
  [-0.614,  0.153,  0.162,  0.185,  0.041,  0.075,  0.122,  0.101],
  [-0.477,  0.081,  0.096,  0.019,  0.078, -0.025,  0.071, -0.091],
  [-0.777,  0.118,  0.365,  0.189,  0.149,  0.209,  0.171, -0.019],
  [-0.461,  0.053,  0.399,  0.473,  0.17 ,  0.28 ,  0.132, -0.124],
  [-0.191,  0.073,  0.055,  0.071, -0.092,  0.06 ,  0.082, -0.291],
  [-0.629, -0.416,  0.111,  0.338, -0.111, -0.902, -0.442,  0.023]]]
```

Problem: Predict next point y_t from past points
Standard ML solution: Train feed-forward neural network
Result: Works OK, but would be nice if

- more accurate
- more intelligible
- less data-needy



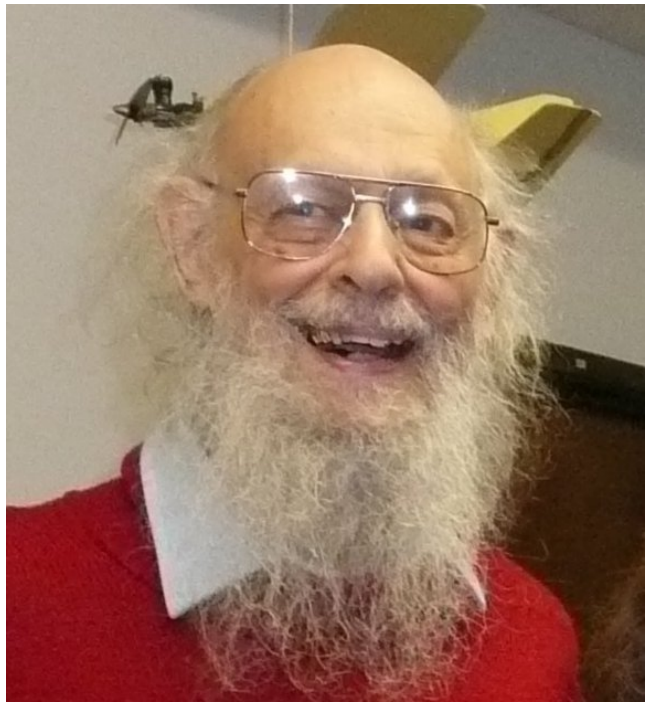
Q. Can we do better?

A. Yes:

Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit *part* of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems

Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit *part* of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems

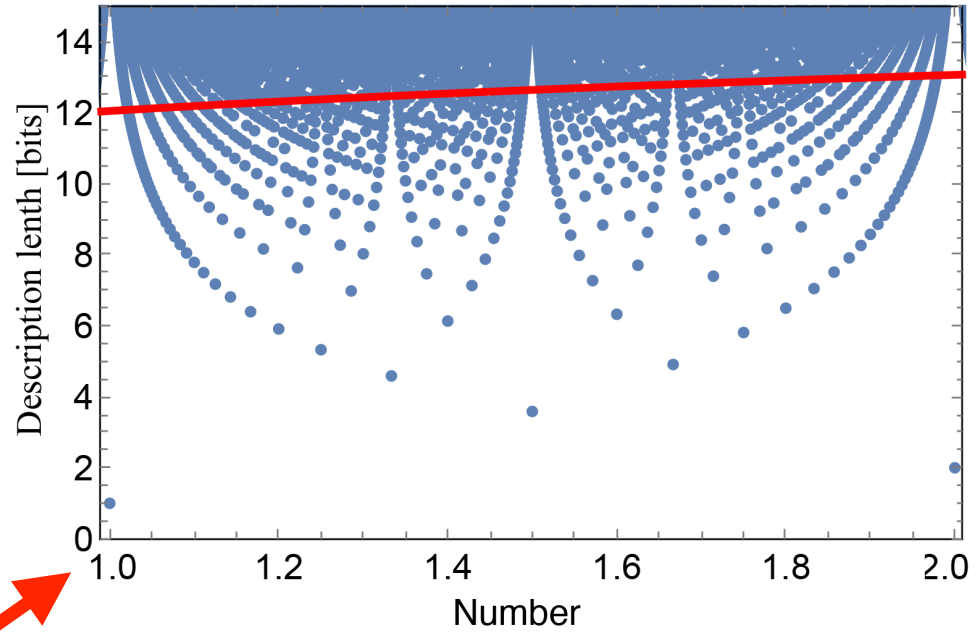
William of Occam

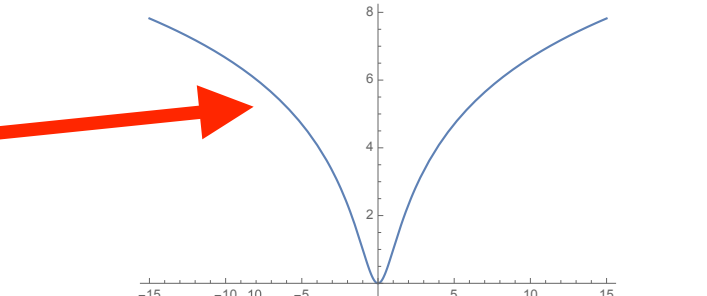


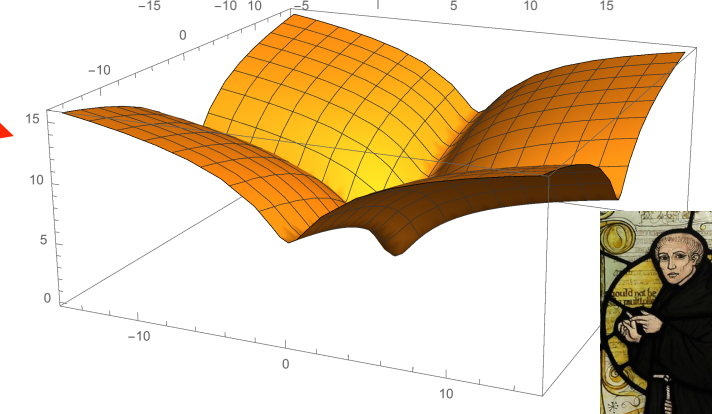
Ray Solomonoff

Description length (bits):

- Natural numbers: $\mathrm{DL}(n) \equiv \log_2 n$
- Integers $\mathrm{DL}(m) \equiv \log_2(1 + |m|)$
- Rational numbers $\mathrm{DL}\left(\frac{m}{n}\right) = \log_2[(1 + |m|)n]$
- Real numbers $\mathrm{DL}(u) \equiv \frac{1}{2}\log_2\left[1 + \left(\frac{u}{\epsilon}\right)^2\right]$
- Vectors $\mathrm{DL}(\mathbf{u}) \equiv \frac{1}{2}\sum_i \log_2\left[1 + \left(\frac{u_i}{\epsilon}\right)^2\right]$
- Model+data $\mathrm{DL}(\mathcal{T}, D) = \mathrm{DL}(\mathcal{T}) + \sum_t \mathrm{DL}(\mathbf{u}_t)$

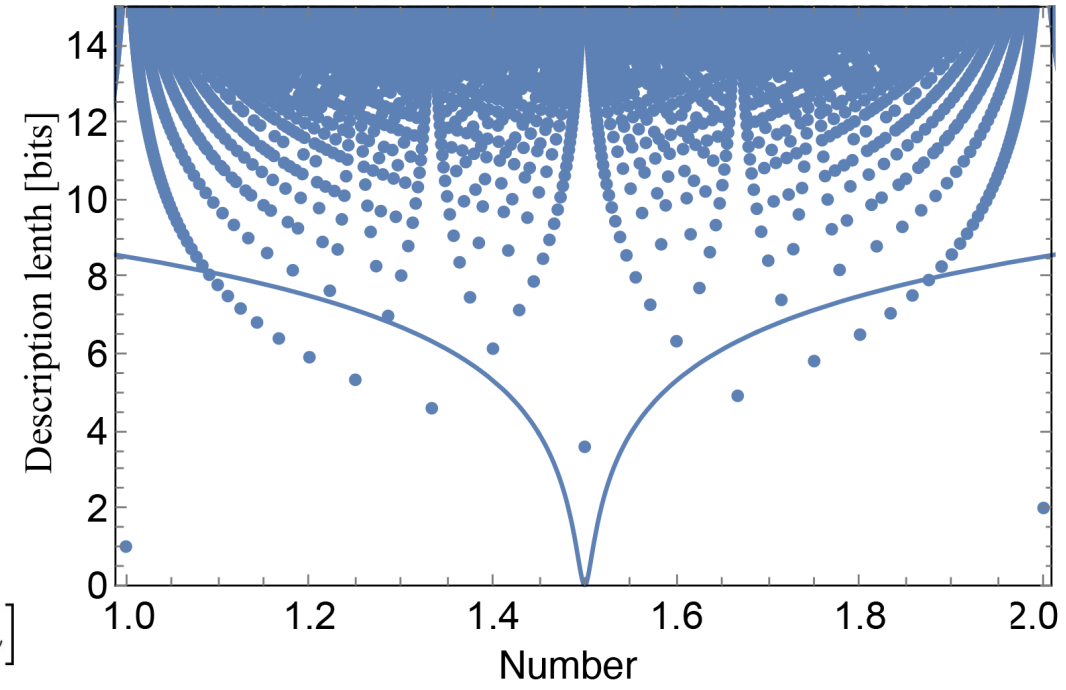






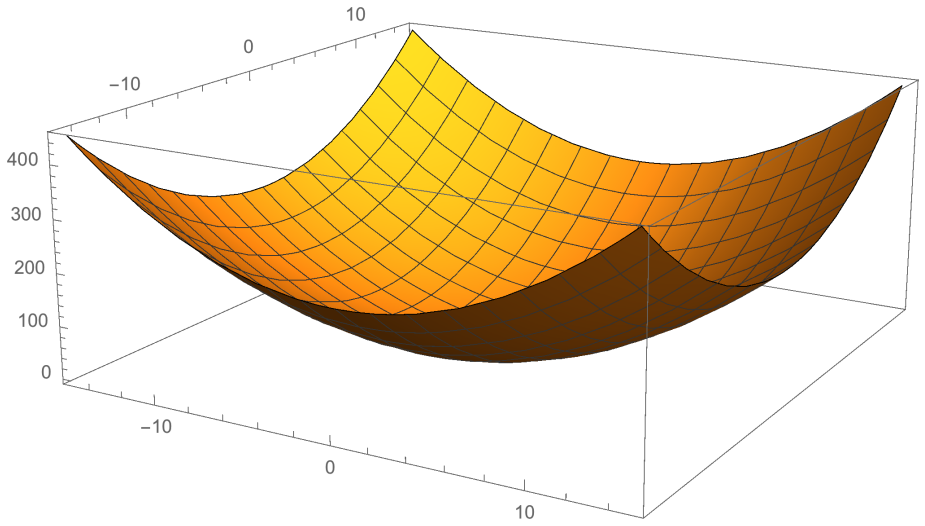
Description length (bits):

- Natural numbers: $\mathrm{DL}(n) \equiv \log_2 n$
- Integers $\mathrm{DL}(m) \equiv \log_2(1 + |m|)$
- Rational numbers $\mathrm{DL}\left(\frac{m}{n}\right) = \log_2[(1 + |m|)n]$
- Real numbers $\mathrm{DL}(u) \equiv \frac{1}{2}\log_2\left[1 + \left(\frac{u}{\epsilon}\right)^2\right]$
- Vectors $\mathrm{DL}(\mathbf{u}) \equiv \frac{1}{2}\sum_i \log_2\left[1 + \left(\frac{u_i}{\epsilon}\right)^2\right]$
- Model+data $\mathrm{DL}(\boldsymbol{\mathcal{T}}, D) = \mathrm{DL}(\boldsymbol{\mathcal{T}}) + \sum_t \mathrm{DL}(\mathbf{u}_t)$

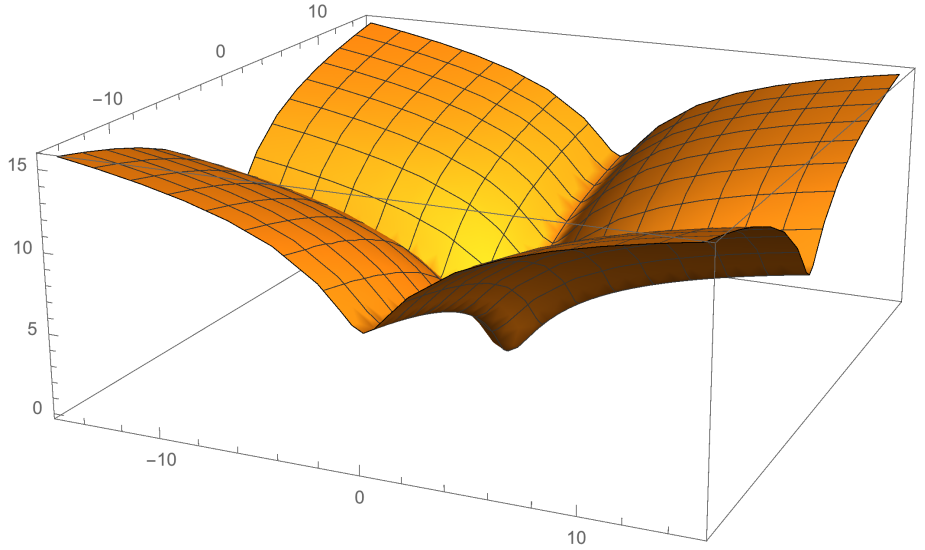


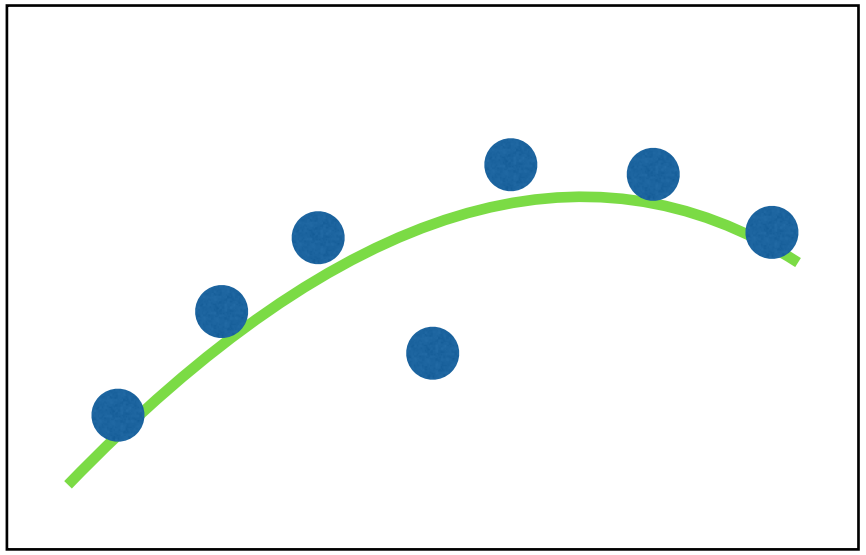
Two universal loss functions:

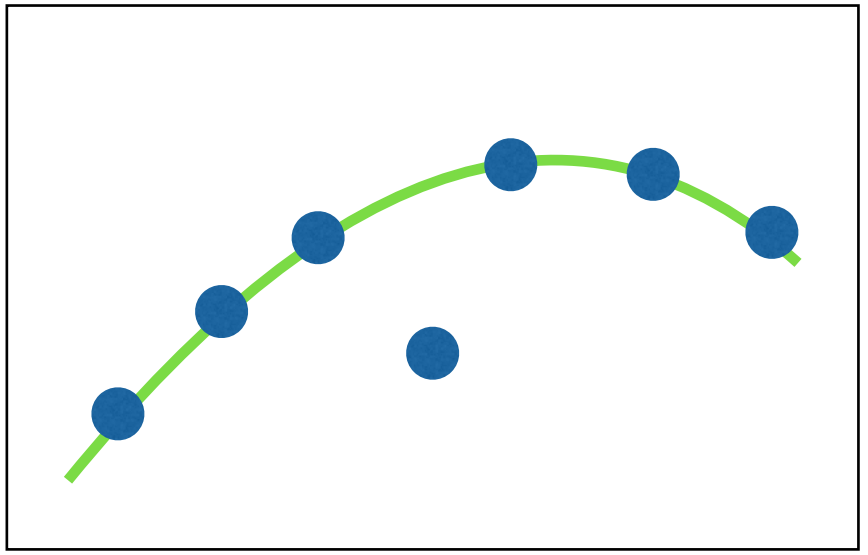
MSE: $L = r^2$

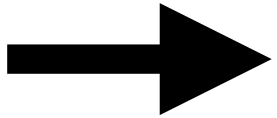


MDL: $L = \max[0, \log r]$



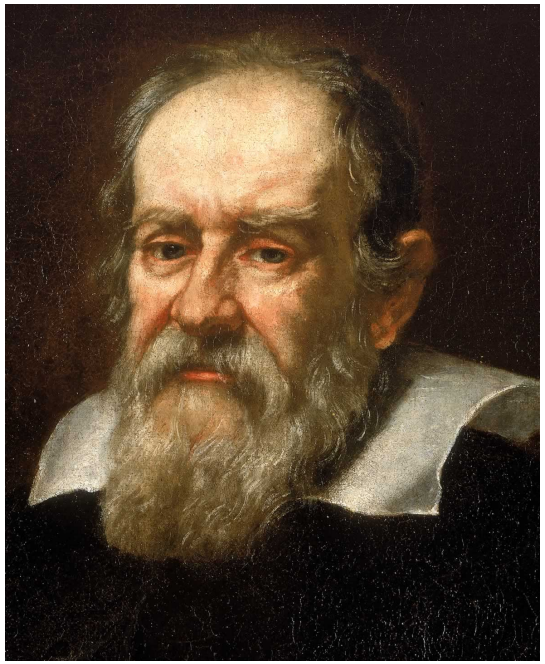






Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit *part* of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems

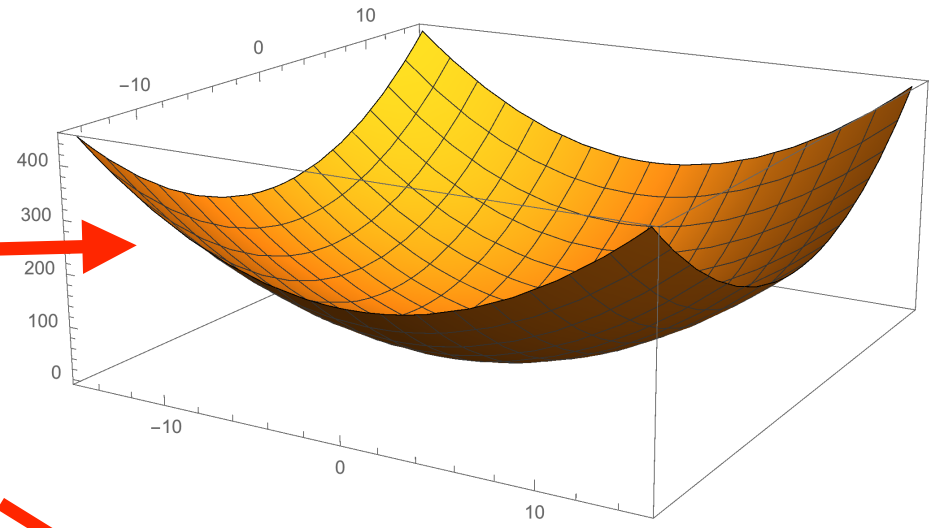
Julius Caesar, arXiv:49BC.0001



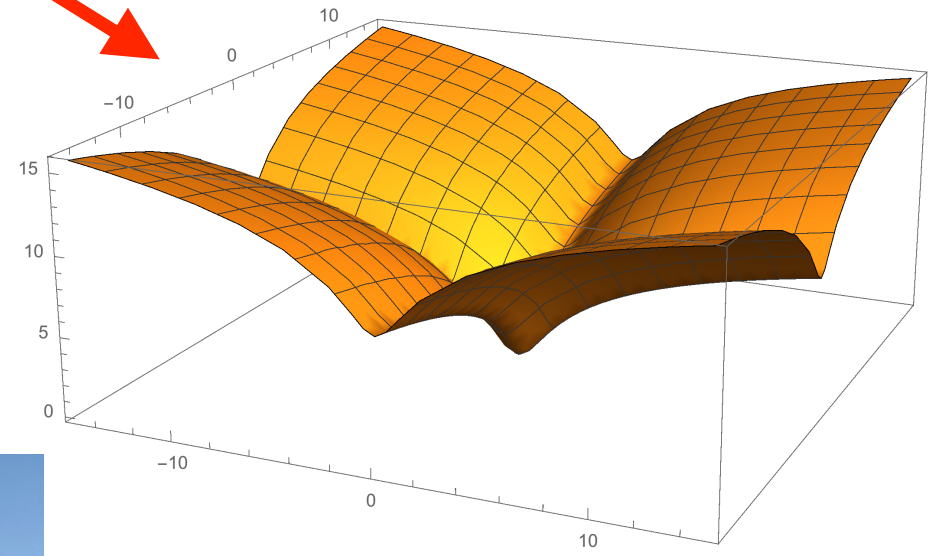
Galileo Galilei, arXiv:1581AD.0001

LOSS FUNCTIONS:

$$\ell(\mathbf{u}) = |\mathbf{u}|^2$$



$$\ell(\mathbf{u}) \equiv \mathrm{DL}(\mathbf{u}) \equiv \frac{1}{2}\sum_i \log_2\left[1+\left(\frac{u_i}{\epsilon}\right)^2\right]$$



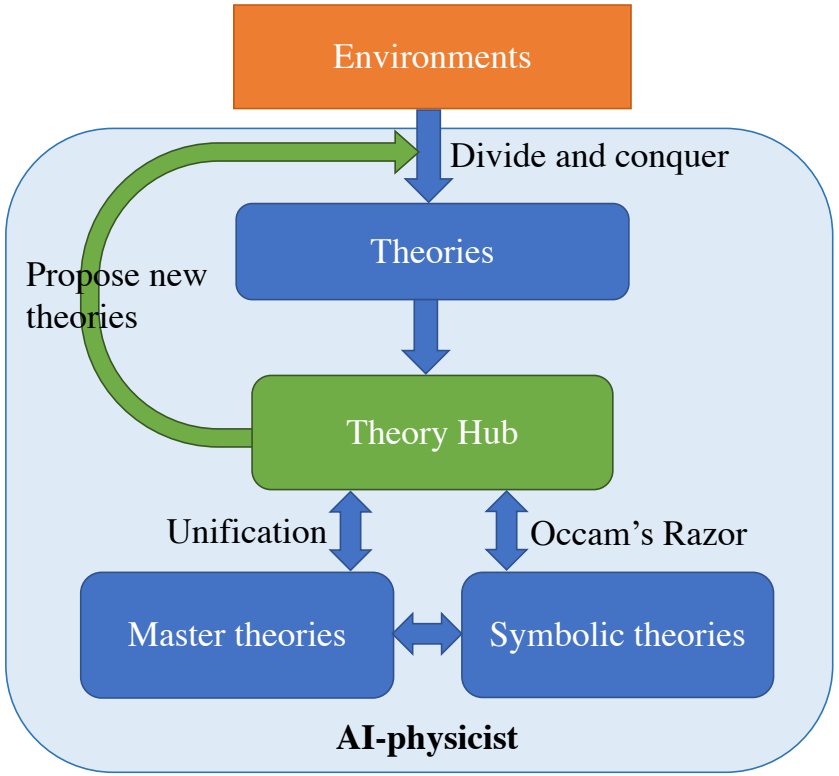
$$\mathcal{L} \equiv \sum_t \ell[\mathbf{y}_t - \mathbf{f}(\mathbf{x}_t)]$$

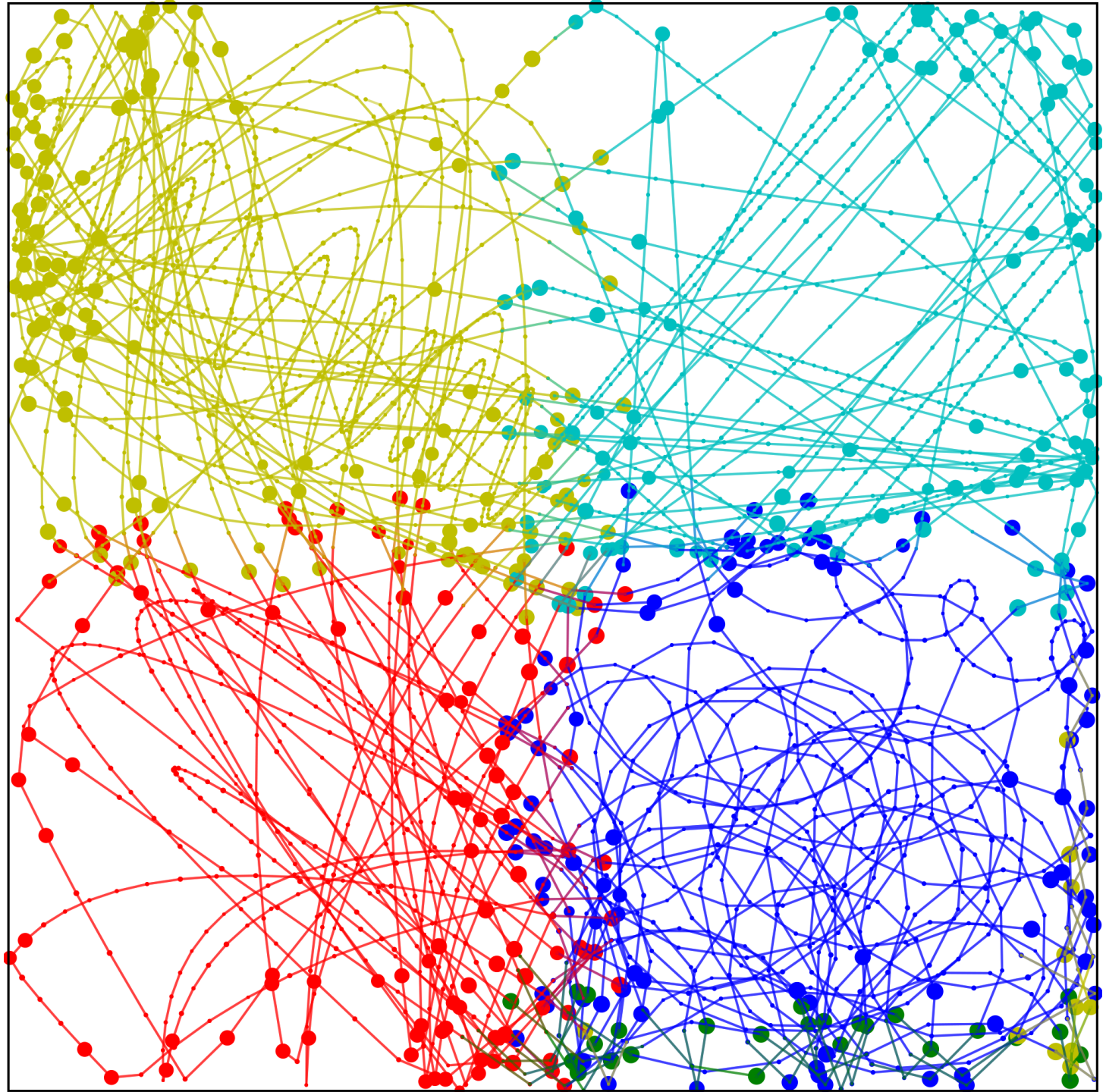
$$\mathcal{L}_\gamma \equiv \sum_t \left(\frac{1}{M}\sum_{i=1} \ell[\mathbf{y}_t - \mathbf{f}_i(\mathbf{x}_t)]^\gamma\right)^{1/\gamma}$$

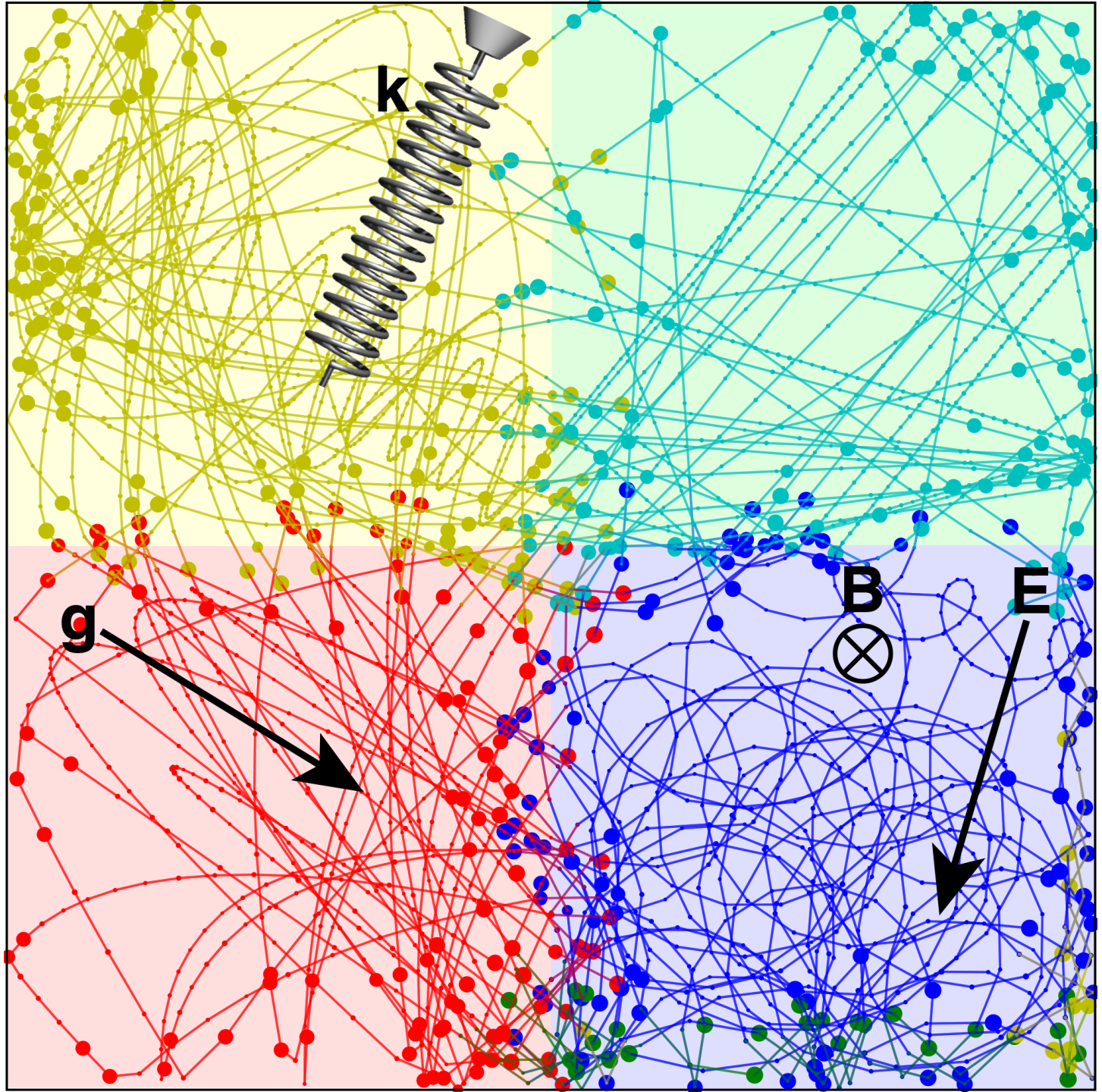
$$\mathcal{L} \equiv \sum_t \left(\frac{1}{M}\sum_{i=1}^{M} \ell[\mathbf{y}_t - \mathbf{f}_i(\mathbf{x}_t)]^{-1}\right)^{-1}$$

“Harmonic loss”

Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit *part* of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems






k
g
B
E

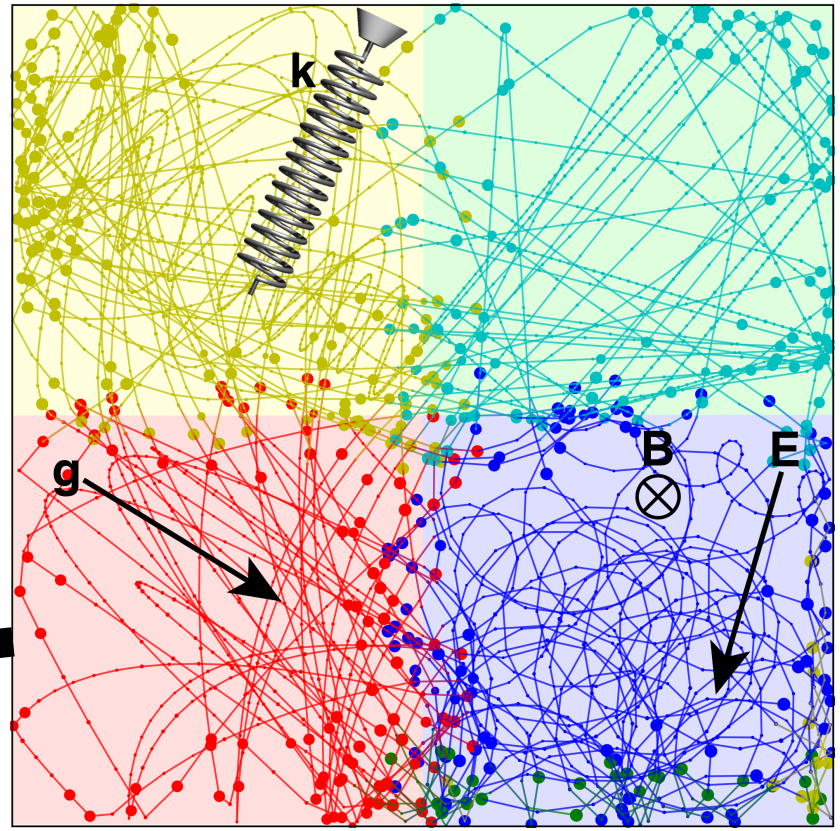
$$\hat{\mathbf{y}}_t = \begin{pmatrix} \text{-0.99999994} & \text{0.00000006} & \text{1.99999990} & \text{-0.00000012} \\ \text{-0.00000004} & \text{-1.0000000} & \text{0.00000004} & \text{2.00000000} \end{pmatrix} \mathbf{x}_t + \begin{pmatrix} \text{0.01088213} \\ \text{-0.00776199} \end{pmatrix}$$

$$\hat{\mathbf{y}}_t = \begin{pmatrix} \text{-1} & \text{0} & \text{2} & \text{0} \\ \text{0} & \text{-1} & \text{0} & \text{2} \end{pmatrix} \mathbf{x}_t + \begin{pmatrix} \text{0.010882} \\ \text{-0.007762} \end{pmatrix}$$

$$\begin{aligned} \hat{x}_{t+2} &= 2x_{t+1} - x_t + 0.010882 \\ \hat{y}_{t+2} &= 2y_{t+1} - y_t - 0.007762 \end{aligned}$$

$$\begin{aligned} \hat{x}_{t+2} &= 2x_{t+1} - x_t + q_1 \\ \hat{y}_{t+2} &= 2y_{t+1} - y_t + q_2 \end{aligned}$$

$$\begin{aligned} \ddot{x} &= g_x \\ \ddot{y} &= g_y \end{aligned} \qquad g_i \equiv q_i(\Delta t)^2$$



Strategy	Definition
Divide-and-conquer	Learn multiple theories each of which specializes to fit *part* of the data very well
Occam's Razor	Avoid overfitting by minimizing description length, which can include replacing fitted constants by simple integers or fractions.
Unification	Try unifying learned theories by introducing parameters
Lifelong Learning	Remember learned solutions and try them on future problems

Regions	$\log_{10}$ MSE			Classification accuracy			Unsolved domains			Description length		
	Base-line	New-born	AI phys	Base-line	New-born	AI phys	Base-line	New-born	AI phys	Base-line	New-born	AI phys
Free + gravity	-4.21	-14.02	-14.04	88.59%	100.00%	100.00%	2	0	0	11271.5	60.3	60.3
Free + gravity	-3.69	-14.03	-14.03	67.65%	100.00%	100.00%	2	0	0	11364.2	60.2	41.9
Free + gravity	-4.18	-13.98	-13.98	80.98%	100.00%	100.00%	2	0	0	11341.7	60.6	57.6
Free + gravity	-4.51	-14.06	-14.07	87.66%	100.00%	100.00%	2	0	0	11289.3	5.2	59.8
Free + harmonic	-3.77	-13.99	-13.94	73.54%	100.00%	100.00%	2	0	0	11333.8	94.4	139.9
Free + harmonic	-3.60	-14.05	-13.89	66.92%	100.00%	100.00%	2	0	0	11337.4	173.0	172.8
Free + harmonic	-3.77	-14.04	-13.95	59.46%	100.00%	100.00%	2	0	0	11317.5	156.0	173.8
Free + harmonic	-5.32	-10.48	-13.14	80.29%	100.00%	100.00%	2	1	0	11219.5	91.6	90.5
Free + harmonic	-3.64	-14.00	-13.89	71.70%	100.00%	100.00%	2	0	0	11369.6	143.7	136.6
Free + EM	-3.62	-13.95	-13.96	82.77%	100.00%	100.00%	2	0	0	11397.5	142.8	284.9
Free + EM	-4.13	-13.82	-13.67	76.55%	100.00%	100.00%	2	0	0	11283.0	306.2	306.2
Free + EM	-4.03	-13.45	-13.47	74.56%	99.97%	99.97%	2	0	0	11388.1	305.9	307.9
Free + EM	-4.31	-13.77	-13.62	86.68%	99.91%	99.91%	2	0	0	11257.7	152.0	133.5
Free + EM	-4.32	-14.00	-14.05	84.55%	100.00%	100.00%	2	0	0	11258.9	303.7	303.8
Free + EM rational	-3.45	-13.96	-13.95	77.88%	99.96%	99.93%	2	0	0	11414.9	194.2	195.8
Free + EM rational	-3.90	-13.96	-13.91	71.13%	100.00%	100.00%	2	0	0	11340.0	199.0	199.0
Free + EM rational	-4.12	-13.97	-13.90	72.78%	100.00%	100.00%	2	0	0	11330.7	198.8	198.8
Free + EM rational	-4.02	-14.07	-14.00	77.92%	100.00%	100.00%	2	0	0	11323.5	197.8	197.8
Free + EM rational	-4.83	-13.87	-13.86	91.14%	100.00%	100.00%	2	0	0	11247.1	10.3	13.9
Free + gravity + harmonic	-4.08	-14.03	-13.95	60.08%	100.00%	100.00%	3	0	0	11269.0	191.8	191.9
Free + gravity + harmonic	-4.31	-14.02	-13.66	63.01%	100.00%	100.00%	3	0	0	11334.2	170.4	83.1
Free + gravity + harmonic	-4.01	-14.01	-13.99	67.48%	100.00%	100.00%	3	0	0	11351.0	168.7	198.9
Free + gravity + harmonic	-3.64	-13.97	-13.88	60.02%	99.97%	99.93%	3	0	0	11374.6	225.7	225.7
Free + gravity + harmonic	-4.11	-7.42	-7.43	51.63%	100.00%	99.97%	3	1	1	11313.7	193.5	179.2
Free + gravity + EM	-3.79	-13.93	-13.47	57.89%	100.00%	100.00%	3	0	0	11334.0	323.9	346.8
Free + gravity + EM	-4.18	-14.00	-14.00	77.16%	100.00%	100.00%	3	1	1	11301.0	277.9	96.2
Free + gravity + EM	-3.38	-13.58	-13.87	53.33%	100.00%	99.96%	3	0	0	11381.4	360.4	364.0
Free + gravity + EM	-3.46	-13.87	-13.89	49.08%	100.00%	100.00%	3	0	0	11370.1	354.0	350.4
Free + gravity + EM	-3.54	-13.69	-13.83	51.28%	100.00%	100.00%	3	0	0	11370.3	331.1	320.7
Free + harmonic + EM	-3.87	-13.82	-13.55	67.27%	100.00%	100.00%	3	0	0	11404.0	267.1	275.4
Free + harmonic + EM	-3.69	-13.87	-10.93	56.02%	99.97%	99.94%	3	0	0	11413.4	468.5	464.9
Free + harmonic + EM	-4.06	-13.39	-13.56	70.87%	100.00%	100.00%	3	0	0	11340.0	452.3	452.3
Free + harmonic + EM	-3.46	-13.94	-10.51	59.02%	99.97%	99.93%	3	0	0	11416.0	475.5	471.9
Free + harmonic + EM	-3.70	-13.75	-13.82	61.67%	100.00%	100.00%	3	0	0	11354.9	466.8	466.8
Free + gravity + harmonic + EM	-3.76	-13.82	-9.48	27.93%	100.00%	99.94%	4	0	0	11358.8	526.9	530.4
Free + gravity + harmonic + EM	-3.74	-13.00	-13.18	40.80%	100.00%	99.97%	4	1	1	11284.8	418.5	389.1
Free + gravity + harmonic + EM	-4.09	-13.97	-13.75	35.69%	100.00%	100.00%	4	0	0	11297.4	504.6	504.6
Free + gravity + harmonic + EM	-3.63	-13.80	-9.99	31.61%	100.00%	99.97%	4	0	0	11407.4	526.3	526.2
Free + gravity + harmonic + EM	-3.51	-6.37	-13.52	32.97%	100.00%	100.00%	4	0	0	11445.8	527.4	527.5
Median	-3.89	-13.95	-13.88	67.56%	100.00%	100.00%	2.5	0.00	0.00	11338.7	198.9	198.9
Mean	-3.94	-13.44	-13.29	65.51%	99.99%	99.99%	2.6	0.10	0.07	11337.9	253.7	252.9

Regions	Epochs to 10^{-2}			Epochs to 10^{-4}			Epochs to 10^{-6}			Epochs to 10^{-8}		
	Base-line	New-born	AI-phys	Base-line	New-born	AI physi	Base-line	New-born	AI phys	Base-line	New-born	AI phys
Free+gravity	100	85	85	8440	120	120	∞	4175	3625	∞	6315	4890
Free+gravity	100	70	10	4680	190	35	∞	2900	4650	∞	2995	6500
Free+gravity	85	100	15	∞	135	30	∞	8205	3815	∞	9620	6455
Free+gravity	95	75	20	7495	140	25	∞	6735	1785	∞	8040	2860
Free+gravity	110	75	0	1770	295	35	∞	3740	3240	∞	7030	3460
Free + harmonic	80	75	20	∞	145	25	∞	2725	4050	∞	2830	6145
Free + harmonic	85	75	20	∞	80	25	∞	7965	1690	∞	10000	3400
Free + harmonic	95	75	30	∞	110	30	∞	1805	3895	∞	1855	3900
Free + harmonic	25	20	5	1285	460	10	∞	5390	1060	∞	7225	6385
Free + harmonic	80	95	5	∞	110	20	∞	4380	3300	∞	4800	4035
Free + EM	90	85	20	∞	1190	115	∞	6305	3380	∞	6590	3435
Free + EM	125	120	0	6240	885	70	∞	7310	1865	∞	7565	1865
Free + EM	115	115	15	15260	600	70	∞	2430	1225	∞	2845	4435
Free + EM	145	90	0	6650	140	0	∞	3000	5205	∞	4530	8735
Free + EM	80	80	10	965	200	25	∞	4635	1970	∞	4690	2870
Free + EM rational	80	75	0	∞	580	70	∞	5415	4150	∞	5445	4175
Free + EM rational	100	100	10	∞	460	45	∞	2560	965	∞	2575	5760
Free + EM rational	140	95	10	11050	455	65	∞	1960	1150	∞	6295	4005
Free + EM rational	120	100	5	13315	325	175	∞	3970	1290	∞	4335	3560
Free + EM rational	35	30	35	1155	335	35	∞	3245	2130	∞	5115	5610
Free + gravity + harmonic	150	75	25	9085	130	30	∞	3870	6145	∞	5555	6185
Free + gravity + harmonic	145	90	5	6915	140	25	∞	4525	3720	∞	10275	4430
Free + gravity + harmonic	105	100	15	6925	155	40	∞	6665	6560	∞	8915	6845
Free + gravity + harmonic	95	95	5	∞	120	30	∞	5790	10915	∞	18450	13125
Free + gravity + harmonic	135	95	15	7970	190	45	∞	13125	7045	∞	∞	∞
Free + gravity + EM	130	100	20	∞	575	40	∞	3215	5095	∞	3215	5100
Free + gravity + EM	125	110	15	5650	160	30	∞	6085	4720	∞	8025	4980
Free + gravity + EM	80	65	15	∞	630	120	∞	4100	6250	∞	4100	6570
Free + gravity + EM	80	75	5	∞	90	45	∞	5910	5815	∞	7295	6090
Free + gravity + EM	80	85	20	∞	1380	465	∞	2390	11425	∞	7450	11510
Free + harmonic + EM	85	75	25	∞	600	150	∞	3775	4525	∞	4675	5070
Free + harmonic + EM	85	90	25	∞	1245	200	∞	6225	2340	∞	6390	3180
Free + harmonic + EM	115	85	15	16600	190	35	∞	6035	1515	∞	10065	2110
Free + harmonic + EM	80	70	35	∞	720	195	∞	6990	3895	∞	6995	6115
Free + harmonic + EM	85	65	10	∞	985	165	∞	5660	1670	∞	5820	1820
Free + gravity + harmonic + EM	90	75	0	∞	540	255	∞	8320	7390	∞	9770	7590
Free + gravity + harmonic + EM	95	80	15	∞	1265	635	∞	6520	6365	∞	8475	6475
Free + gravity + harmonic + EM	130	85	10	8620	575	105	∞	6320	4035	∞	9705	7685
Free + gravity + harmonic + EM	75	80	0	∞	815	425	∞	7575	8405	∞	10440	8620
Free + gravity + harmonic + EM	80	65	20	∞	735	280	∞	6715	4555	∞	12495	8495
Median	95	83	15	∞	330	45	∞	5403	3895	∞	6590	5100
Mean	98	82	15	∞	455	109	∞	5217	4171	∞	6892	5499

Benchmark	Baseline	Newborn	AI Physicist
$\log_{10}$ mean-squared error	-3.89	-13.95	-13.88
Classification accuracy	67.56%	100.00%	100.00%
Fraction of worlds solved	0.00%	90.00%	92.50%
Description length for $\mathbf{f}$	11,338.7	198.9	198.9
Epochs until 0.01 MSE	95	83	15
Epochs until 0.0001 MSE	6925	330	45
Epochs until 10^{-6} MSE	∞	5403	3895
Epochs until 10^{-8} MSE	∞	6590	5100
$\log_{10}$ MSE error			
using 100% of data	-3.78	-13.89	-13.89
using 50% of data	-3.84	-13.76	-13.81
using 10% of data	-3.16	-7.38	-10.54
using 5% of data	-3.06	-6.06	-6.20
using 1% of data	-2.46	-3.69	-3.95
Epochs until 0.01 MSE			
using 100% of data	95	80	15
using 50% of data	190	152.5	30
using 10% of data	195	162.5	30
using 5% of data	205	165	30
using 1% of data	397.5	235	35

AI Feynman: a Physics-Inspired Method for Symbolic Regression

Silviu-Marian Udrescu, Max Tegmark
Dept. of Physics & Center for Brains, Minds & Machines,
Massachusetts Institute of Technology, Cambridge, MA 02139; sudrescu@mit.edu

(Dated: May 29, 2019)

A core challenge for both physics and artificial intellicence (AI) is *symbolic regression*: finding a symbolic expression that matches data from an unknown function. Although this problem is likely to be NP-hard in principle, functions of practical interest often exhibit symmetries, separability, compositionality and other simplifying properties. In this spirit, we develop a recursive multidimensional symbolic regression algorithm that combines neural network fitting with a suite of physics-inspired techniques. We apply it to 100 equations from the Feynman Lectures on Physics, and it discovers all of them, while previous publicly available software cracks only 71; for a more difficult test set, we improve the state of the art success rate from 15% to 90%.

Silviu-Marian Udrescu

I. INTRODUCTION

In 1601, Johannes Kepler got access to the world's best data tables on planetary orbits, and after 4 years and about 40 failed attempts to fit the Mars data to various ovoid shapes, he launched a scientific revolution by discovering that Mars' orbit was an ellipse [1]. This was an example of *symbolic regression*: discovering a symbolic expression that accurately matches a given data set. More specifically, we are given a table of numbers, whose rows are of the form $\{x_1, ..., x_n, y\}$ where $y = f(x_1, ..., x_n)$, and our task is to discover the correct symbolic expression for the unknown mystery function f, optionally including the complication of noise.

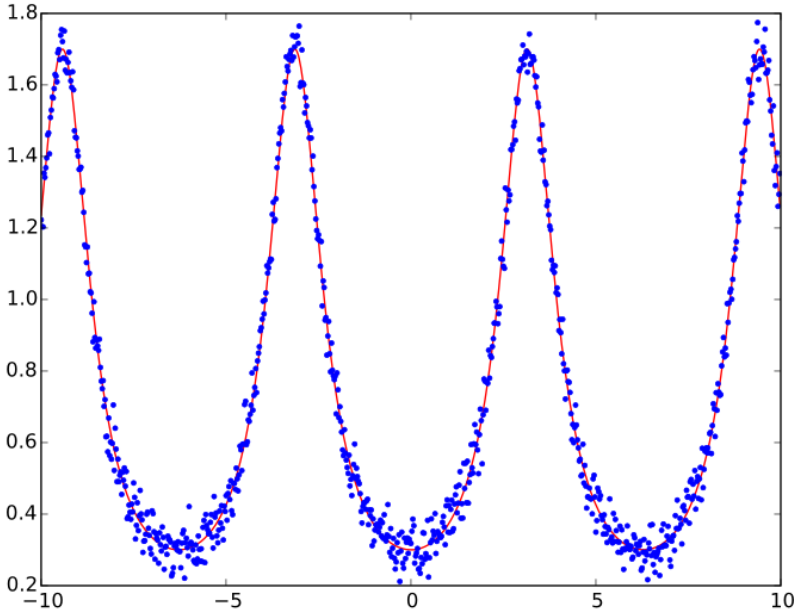
Growing data sets have motivated attempts to automate such regression tasks, with significant success. For the special case where the unknown function f is a linear combination of known functions of $\{x_1, ..., x_n\}$, symbolic regression reduces to simply solving a system of linear equations. *Linear* regression (where f is simply a linear function) is ubiquitous in the scientific literature, from

in exponentially large spaces, which replace the above-mentioned brute-force search by biology-inspired strategies of mutation, selection, inheritance and recombination; crudely speaking, the role of genes is played by useful symbol strings that may form part of the sought-after formula or program. Such algorithms have been successfully applied to areas ranging from design of antennas [4, 5] and vehicles [6] to wireless routing [7], vehicle routing [8], robot navigation [9], code breaking [10], investment strategy [11], marketing [12], classification [13], Rubik's cube [14], program synthesis [15] and metabolic networks [16].

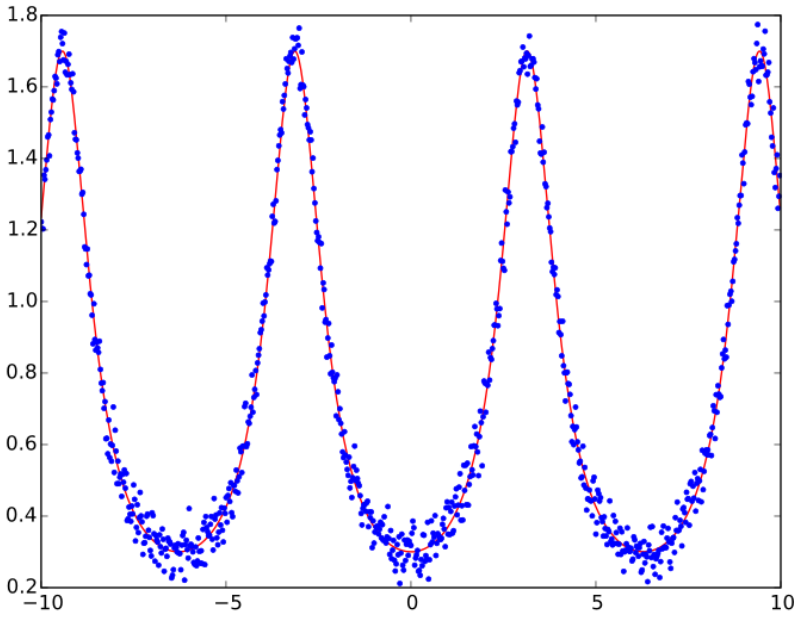
The symbolic regression problem for mathematical functions (the focus of this paper) has been tackled with a variety of methods [17–19], including genetic algorithms [20, 21]. By far the most successful of these is, as we will see in Section III, the genetic algorithm outlined in [22] and implemented in the commercial *Eureqa* software [21].

The purpose of this paper is to further improve on this state-of-the-art, using physics-inspired strategies enabled by neural networks. Our most important contribution is using neural networks to discover hidden simplicity such

- What is symbolic regression?
  - Finding a symbolic expression that matches data from an unknown function



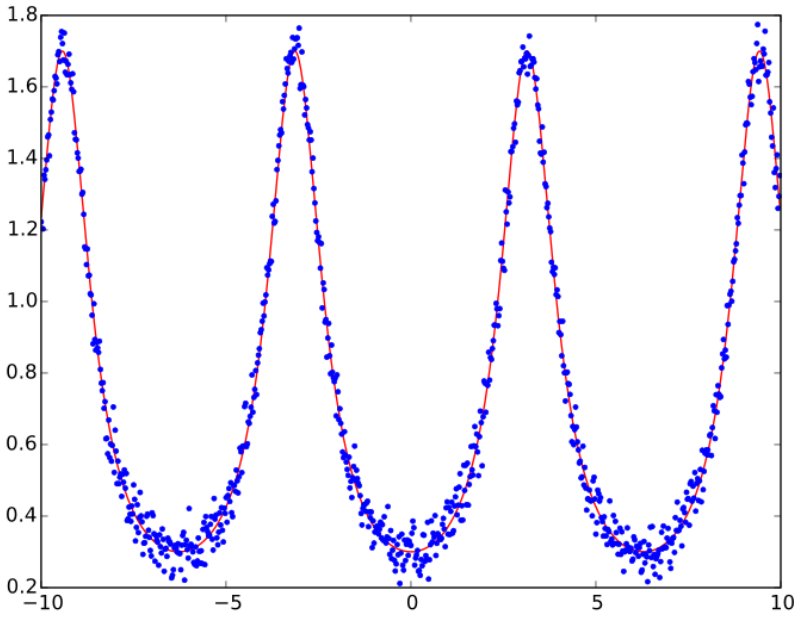
- What is symbolic regression?
  - Finding a symbolic expression that matches data from an unknown function



Took Kepler 4 years to figure out,
but was worth it!

$$r = \frac{a(1-e^2)}{1 + e\cos(\theta_1 - \theta_2)}$$

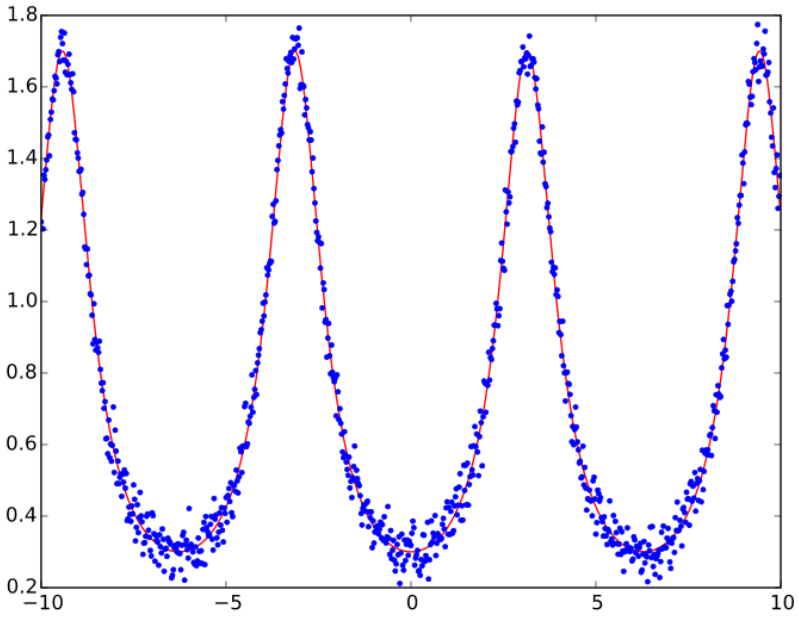
- What is symbolic regression?
  - Finding a symbolic expression that matches data from an unknown function



- One can use brute force to find a solution:
  - Try each combination of allowed mathematical symbols and variables and see if they match the data
  - Use reverse polish notation:

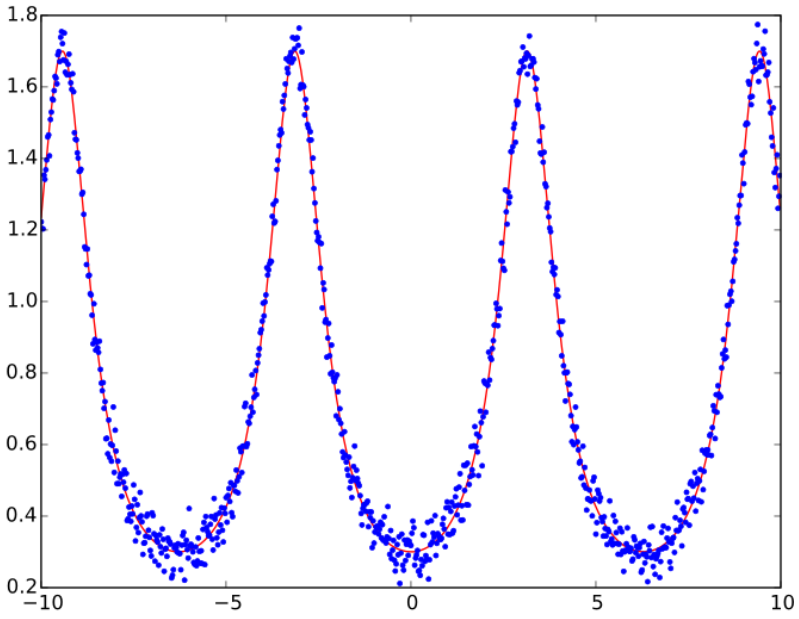
    $a * b \rightarrow ab*$, $a^2 + b \rightarrow aa*b+$

- What is symbolic regression?
  - Finding a symbolic expression that matches data from an unknown function



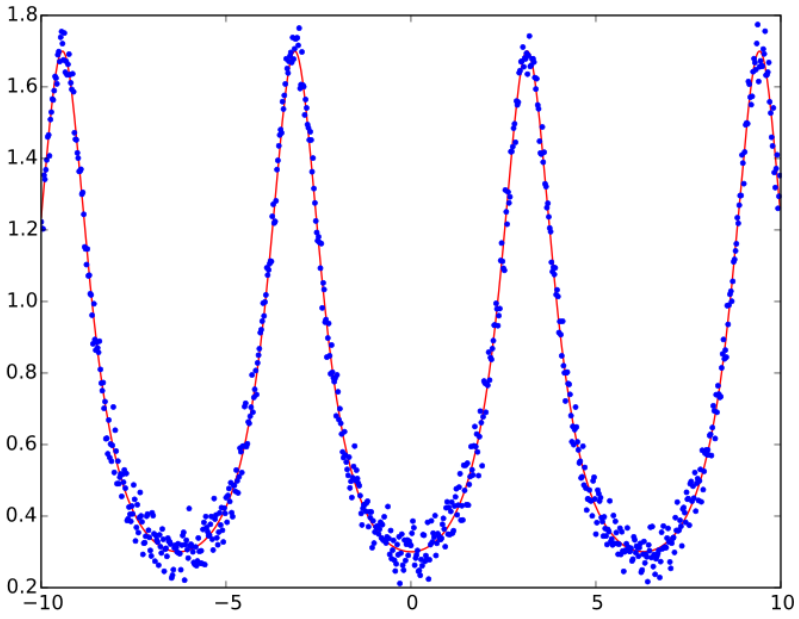
- One can use brute force to find a solution:
  - Try each combination of allowed mathematical symbols and variables and see if they match the data
  - Use reverse polish notation:
    $a * b \rightarrow ab*$, $a^2 + b \rightarrow aa*b+$
  - The search space grows exponentially with the number of symbols
  - $f = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} \rightarrow \boldsymbol{xx*0} \gg /\sim \boldsymbol{EPP} + \boldsymbol{R}/ \rightarrow \sim 30$ years

- What is symbolic regression?
  - Finding a symbolic expression that matches data from an unknown function



- One can use brute force to find a solution:
  - Try each combination of allowed mathematical symbols and variables and see if they match the data
  - Use reverse polish notation:
    $a * b \rightarrow ab*$, $a^2 + b \rightarrow aa*b+$
  - The search space grows exponentially with the number of symbols
  - $f = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} \rightarrow \boldsymbol{xx*0} \gg /\sim \boldsymbol{EPP} + \boldsymbol{R}/ \rightarrow \sim 30$ years
  - $\omega = \frac{1+\frac{v}{c}}{\sqrt{1-\frac{v^2}{c^2}}}\omega_0 \rightarrow \sim 10^6$ years

- What is symbolic regression?
  - Finding a symbolic expression that matches data from an unknown function



- One can use brute force to find a solution:
  - Try each combination of allowed mathematical symbols and variables and see if they match the data
  - Use reverse polish notation:
    $a * b \rightarrow ab*$, $a^2 + b \rightarrow aa*b+$
  - The search space grows exponentially with the number of symbols
  - $f = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} \rightarrow \boldsymbol{xx*0} \gg /\sim \boldsymbol{EPP} + \boldsymbol{R}/ \rightarrow \sim 30$ years
  - $\omega = \frac{1+\frac{v}{c}}{\sqrt{1-\frac{v^2}{c^2}}}\omega_0 \rightarrow \sim 10^6$ years
  - $I = \frac{\hbar\omega^3}{\pi^2 c^2 (e^{\frac{\hbar\omega}{kT}} - 1)} \rightarrow \sim 10^{17}$ years $\sim 10^7$ age of Universe

- Previous approaches used genetic algorithms → Eureqa (M. Schmidt, H. Lipson, 2009)
- Our approach: Take advantage of the properties of functions of practical interest:
  - Symmetry
  - Separability
  - Compositionality
  - Units of the variables
  - Smoothness => NN
- Break the function apart into smaller parts and solve each part independently using brute force.
- Use 100 equations from The Feynman Lectures on Physics to test the performance of out algorithm.

$$f = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}}$$

$$m = \frac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}}$$

$$K = \frac{1}{2}m\left(v^2 + u^2 + w^2\right)$$

$$E = \frac{3}{4\pi\epsilon}\frac{pz}{r^5}\sqrt{x^2 + y^2}$$

$$\mathrm{F} = \frac{Gm_1m_2}{\left(x_1 - x_2\right)^2 + \left(y_1 - y_2\right)^2 + \left(z_1 - z_2\right)^2}$$

$$r = \frac{m_1r_1 + m_2r_2}{m_1 + m_2}$$

$$L = mrv\sin\theta$$

$$x = \sqrt{x_1^2 + x_2^2 - 2x_1x_2\cos(\theta_1 - \theta_2)}$$

$$P = \frac{pE}{\hbar}\frac{\sin\left(\frac{(\omega - \omega_0)t}{2}\right)^2}{\left(\frac{(\omega - \omega_0)}{2}\right)^2}$$

$$E = 2U(1 - \cos kd)$$

$$x = x_1(\cos(\omega t) + \alpha\cos(\omega t)^2)$$

$$P = \frac{nk_bT}{V}$$

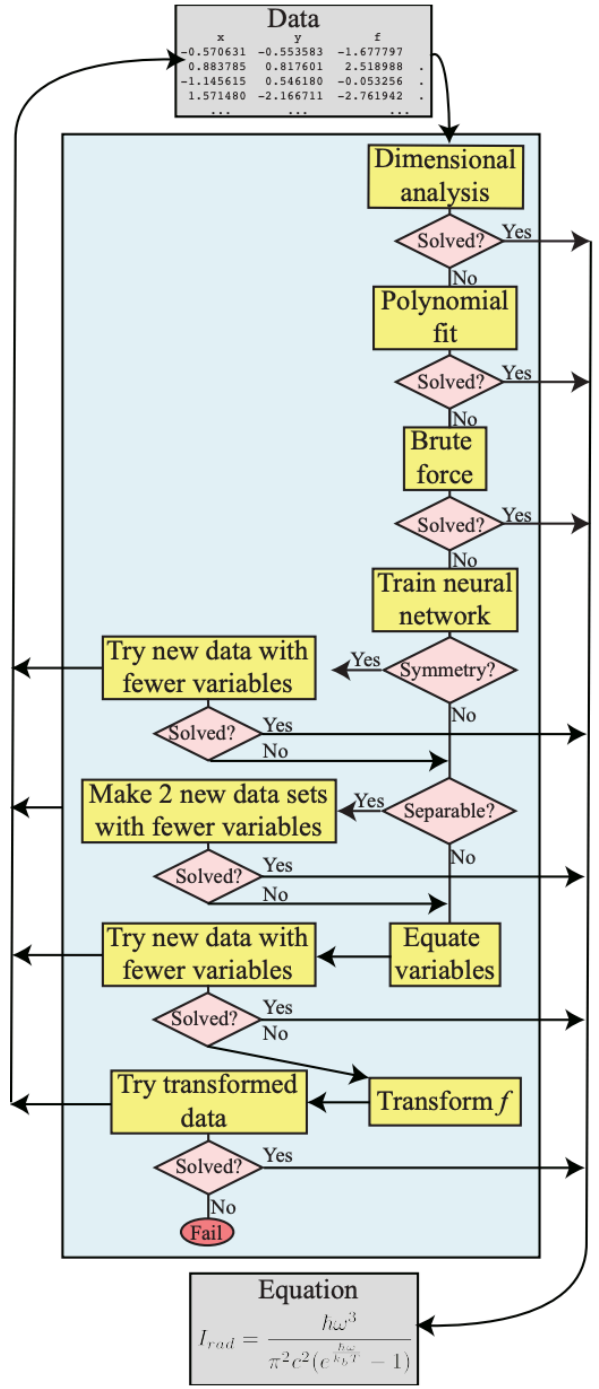
$$L = \frac{\hbar\omega^3}{\pi^2c^2(e^{\hbar\omega/k_bT} - 1)}$$

$$P = \frac{n\alpha}{1 - \frac{n\alpha}{3}}\epsilon E$$


1.1431209959193709 2.7700644791483753 1.7508575540193472 0.23452895063856676
2.4680655653881054 2.2073166947348444 1.7761705838854234 0.15919403345867914
2.7621479700455853 1.4168131204210188 1.5378176974809339 0.14429337334417677
1.9536888384746354 2.7336267945043491 1.2592849110534683 0.15360014539410058
2.1532278876457527 1.5016008010765851 1.4218686278023172 0.18514940761978987
1.9899434091665062 1.4250958039594244 2.5409132056932424 0.17131474581788358
1.2841783534277345 2.5038413591290976 1.2255232096430668 0.18928439532548705
2.3550853261290494 2.2555822345853405 1.4525468706453792 0.15982929556091857
2.7529820467784543 2.6405850369222492 1.6148891450043024 0.13519598787103054
1.2043936184306594 1.4441117081403013 1.4546229392136278 0.33122650381723288
1.3423962980280737 2.0552387587225684 2.8071816262301414 0.25403615776576924
1.4156715177406782 1.0577553334831364 2.5122383795854709 0.16623813635214552
2.0187117984150182 1.0672568295716016 1.4729243386484654 0.19367199411465894
1.5109507435415031 2.5932595628763617 2.9492633493443927 0.25680582940703343
2.5078719313855347 1.0771974670079432 2.7296063077362385 0.12803644285479733
1.4846972064278652 2.7401242687034313 1.3356870260708698 0.17177799061908141
2.1398682204221178 1.5010412666279194 2.0784556152016664 0.17976791449866503
1.7000728811732311 1.5155671571998763 1.5295642030291683 0.23465391617605863
2.3516545980830331 1.0329627134609363 2.4473327905843174 0.14157574334122566
2.6157973581192184 2.0432144044920815 1.4576627239471807 0.14873897153317067
1.7508575540193472 0.1622731522067302
1.7761705838854234 0.2302621797695692
1.5378176974809339 0.1292859714460743
1.2592849110534683 0.1304799510153489
1.4218686278023172 0.1550546569588785
2.5409132056932424 0.2923818062791344
1.2255232096430668 0.2510574942870803
1.4525468706453792 0.1579155509580034
1.6148891450043024 0.1274291905284596
1.4546229392136278 0.1450699418916719
2.8071816262301414 0.2155645614988079
2.5122383795854709 0.2503687300370431
1.4729243386484654 0.2089983354998851
2.9492633493443927 0.1402358941533639
2.7296063077362385 0.2265496300298594
1.3356870260708698 0.1736561350680618
2.0784556152016664 0.1402358941533639
1.5295642030291683 0.1655325251599024
2.4473327905843174 0.2481481255299127
1.4576627239471807 0.1340330042989904
1.1431209959193709 0.3493011816455185
2.4680655653881054 0.3510535854942762
2.7621479700455853 0.1449505804785308
1.9536888384746354 0.2186919801694292
2.1532278876457527 0.3419915928477334
1.9899434091665062 0.2261770389103675
1.2841783534277345 0.1419964531147948
2.3550853261290494 0.2026257866465666
2.7529820467784543 0.1531056963832452
1.2043936184306594 0.1948645845149858
1.3423962980280737 0.1772545424107829
1.4156715177406782 0.1644247259359752
2.0187117984150182 0.2318265474470857
1.5109507435415031 0.3358012970398757
2.5078719313855347 0.1843526096297733
1.4846972064278652 0.1763791748051428
2.1398682204221178 0.2446000792256473
1.7000728811732311 0.1582659295296309
2.3516545980830331 0.2991489091774166
2.6157973581192184 0.2049907486482494
1.1431209959193709 2.7700644791483753 0.1707457911193286
2.4680655653881054 2.2073166947348444 0.1363381395217752
2.7621479700455853 1.4168131204210188 0.1354560901272666
1.9536888384746354 2.7336267945043491 0.2295511633451723
2.1532278876457527 1.5016008010765851 0.1284863026533888
1.9899434091665062 1.4250958039594244 0.2804446939442301
1.2841783534277345 2.5038413591290976 0.1949306373422478
2.3550853261290494 2.2555822345853405 0.1977880809625735
2.7529820467784543 2.6405850369222492 0.2496376891910489
1.2043936184306594 1.4441117081403013 0.3346169706493638
1.3423962980280737 2.0552387587225684 0.2535725867763916
1.4156715177406782 1.0577553334831364 0.2068977937880752
2.0187117984150182 1.0672568295716016 0.0984812028737029
1.5109507435415031 2.5932595628763617 0.2114861012697027
2.5078719313855347 1.0771974670079432 0.2018426884652462
1.4846972064278652 2.7401242687034313 0.1374340789894264
2.1398682204221178 1.5010412666279194 0.2664264122114690
1.7000728811732311 1.5155671571998763 0.1971030576655539
2.3516545980830331 1.0329627134609363 0.1735523440724042
2.6157973581192184 2.0432144044920815 0.2022186112443449
2.7700644791483753 0.1582237525765422
2.2073166947348444 0.1720212618293609
1.4168131204210188 0.1571139260793397
2.7336267945043491 0.1980579273883148
1.5016008010765851 0.1393047496371947
1.4250958039594244 0.1805044305151408
2.5038413591290976 0.1405161772807477
2.2555822345853405 0.1563198546405198
2.6405850369222492 0.1536294037742006
1.4441117081403013 0.1368871892899424
2.0552387587225684 0.2108896577944663
1.0577553334831364 0.1470820012048271
1.0672568295716016 0.1466806601424796
2.5932595628763617 0.1846847412501496
1.0771974670079432 0.1904383471004166
2.7401242687034313 0.1414793863414717
1.5010412666279194 0.1517468022868348
1.5155671571998763 0.1552012167424786
1.0329627134609363 0.1347806847282837
2.0432144044920815 0.2520184739425074

Methods

- Dimensional Analysis
- Brute Force
- Polynomial Fit
- Neural Network
  - Check for symmetry (e.g. $f(x, y) = f(x - y)$)
  - Check for separability (e.g. $f(x, y) = g(x)h(y)$)
  - Make 2 variables equal
- Apply unary transformations on the variables



Dimensional Analysis

- Many physics equations can be simplified by requiring the units of the two sides of an equation to match.
- The number of independent variables can be significantly reduced.
- Represent the units of each variable by a six dimensional vector:

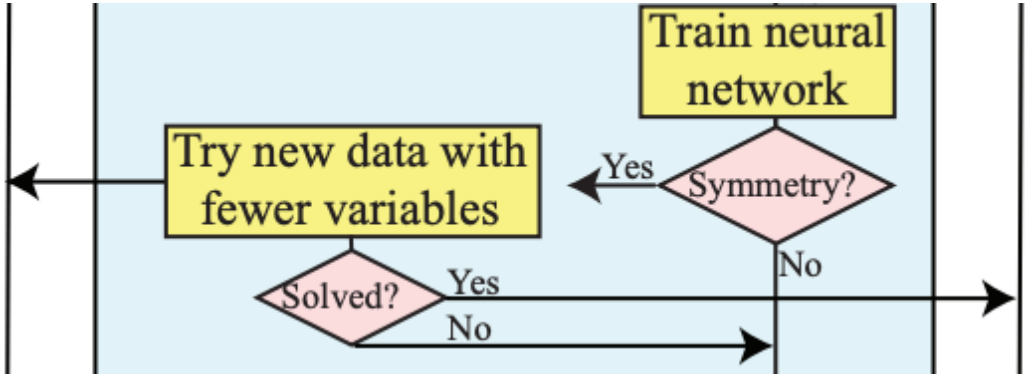
Variables	Units	m	s	kg	T	V
	Acceleration	1	-2	0	0	0
	Angular momentum	2	-1	1	0	0
	Force	1	-2	1	0	0
	Boltzmann constant	2	-2	1	-1	0
	Capacitance	2	-2	1	0	-2
	Electric Charge	2	-2	1	0	-1

Neural Network

- Train a fully connected neural network able to predict the output of an equation for a given input.
- Parameters:
  - 6 layers with 128 neurons
  - Softplus activation function
  - Learning rate = 0.005 (1cycle policy)
  - Batch size = 1024
  - Training/Validation set size = 100,000/20,000
  - Adam optimizer
  - 100 epochs
  - RMSE loss $(10^{-3} - 10^{-5})$

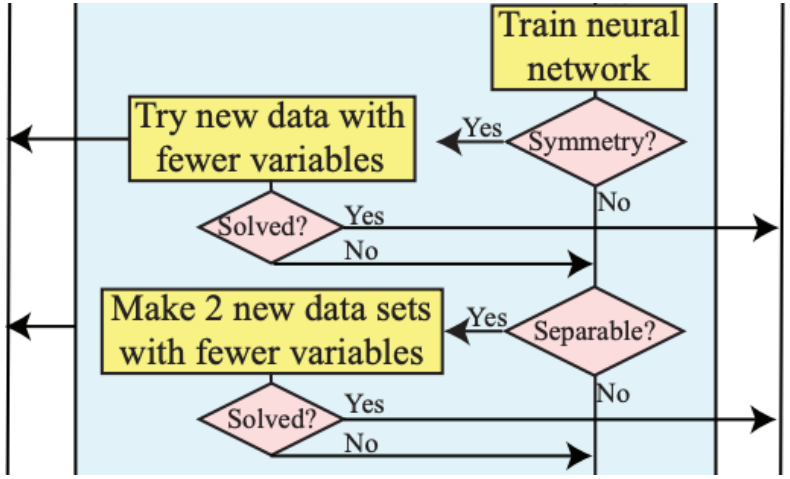
Checking for symmetries using the NN

- For a given function $f(x_1,\ x_2, \ldots, x_n)$, calculate, using the neural network: $f(x_1 + a,\ x_2 + a, \ldots, x_n)$
- If $f(x_1,\ x_2, \ldots, x_n) = f(x_1 + a,\ x_2 + a, \ldots, x_n)$ to within a precision ϵ, given by the neural network error $\Rightarrow$ f depends on x_1 and x_2 only through their difference.
- Replace x_1 and x_2 by $x_1' \equiv x_1 - x_2 \ \Rightarrow f'(x_1',\ x_3, \ldots, x_n)$.
- Do the same for $x_1 + x_2,\ x_1 x_2,$ and $\dfrac{x_1}{x_2}$.
- Example: $f = \dfrac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}} \rightarrow f = \dfrac{e^{-\frac{x'^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}}$



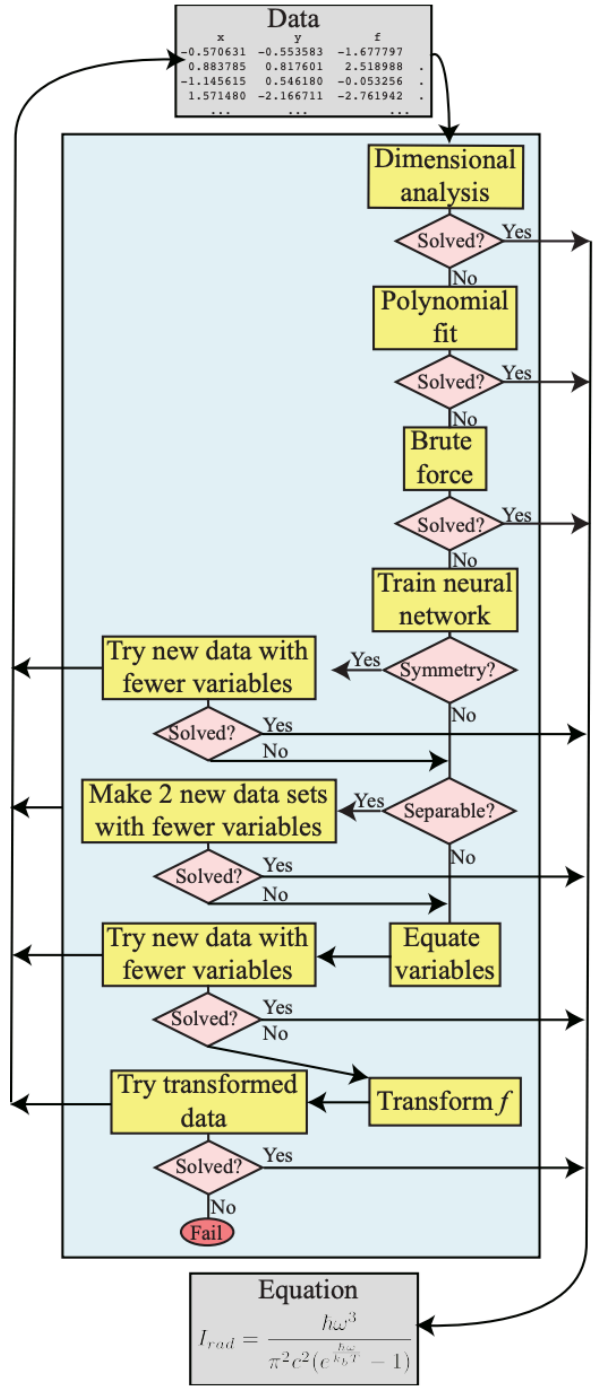
Checking for separability using the NN

- Check if a given equation $f(x_1,\ x_2)$, can be split into two parts with no variables in common: $f(x_1,\ x_2) = g(x_1)h(x_2)$
- If $f(x_1,\ x_2) = \dfrac{f(x_1, c_2)\, f(c_1, x_2)}{f(c_1, c_2)}$ to within a precision ϵ, given by the neural network error $\Rightarrow f$ is multiplicatively separable.
- Define $y' \equiv f(x_1, c_2)$ and $y'' \equiv \dfrac{f(c_1, x_2)}{f(c_1, c_2)}$ and try to find the symbolic expression of each of them.
- Do the same for additive separability: $f(x_1,\ x_2) = g(x_1) + h(x_2)$
- Example: $I = I_0 \left(\dfrac{\sin\left(\frac{\phi}{2}\right)}{\frac{\phi}{2}} \dfrac{\sin\left(\frac{N\delta}{2}\right)}{\sin\left(\frac{\delta}{2}\right)} \right)^2$



Methods

- Dimensional Analysis
- Brute Force
- Polynomial Fit
- Neural Network
  - Check for symmetry (e.g. $f(x, y) = f(x - y)$)
  - Check for separability (e.g. $f(x, y) = g(x)h(y)$)
  - Make 2 variables equal
- Apply unary transformations on the variables



$$\frac{Gm_1m_2}{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}$$

Dimensional Analysis

$$\frac{Gm_1^2}{x_1^2}$$

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

$\Leftrightarrow$

$$\frac{a}{(b-1)^2+(c-d)^2+(e-f)^2}$$

$$\frac{Gm_1m_2}{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}$$

Dimensional Analysis

$$\frac{Gm_1^2}{x_1^2}$$

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

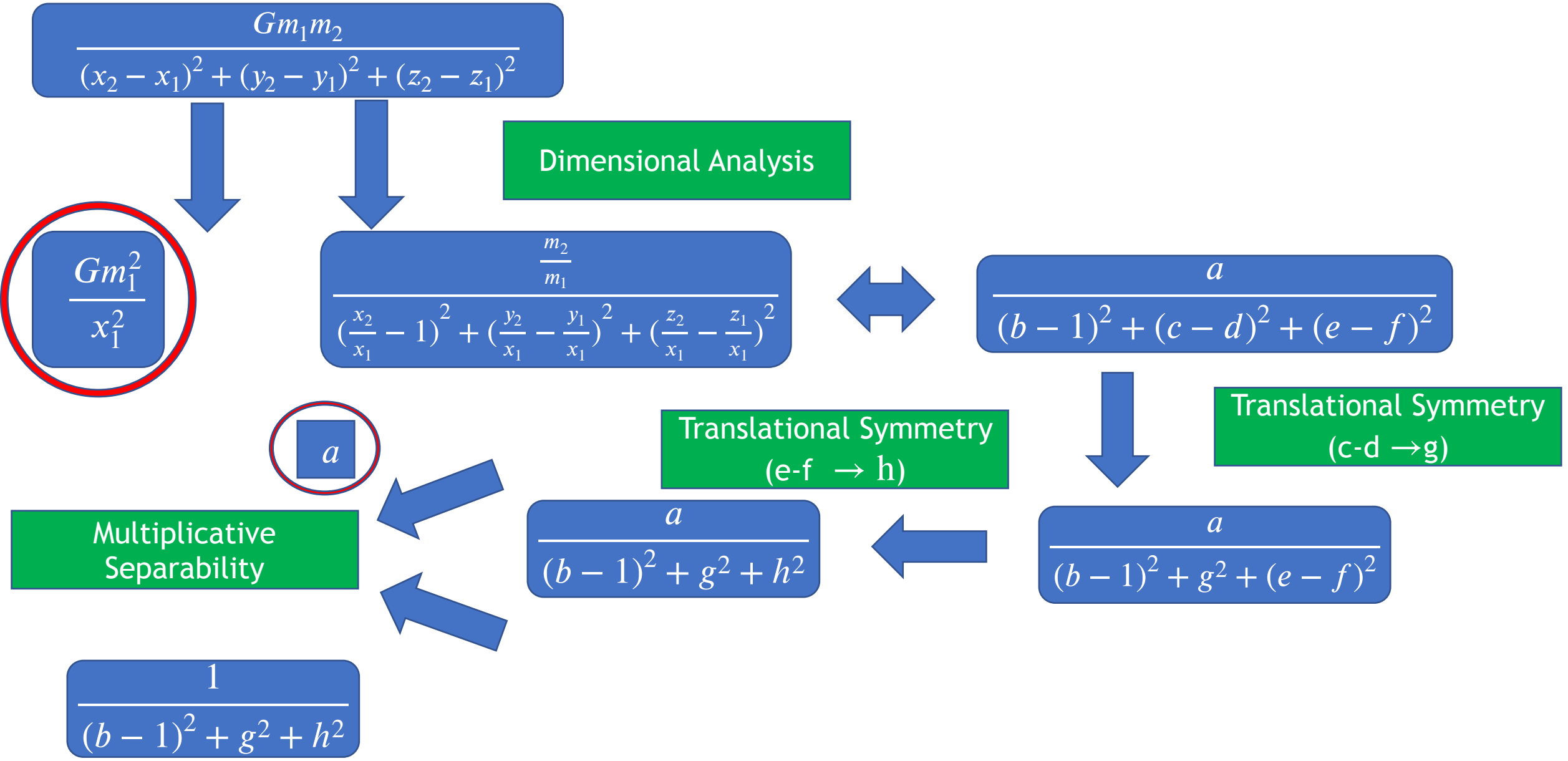
$$\frac{a}{(b-1)^2+(c-d)^2+(e-f)^2}$$

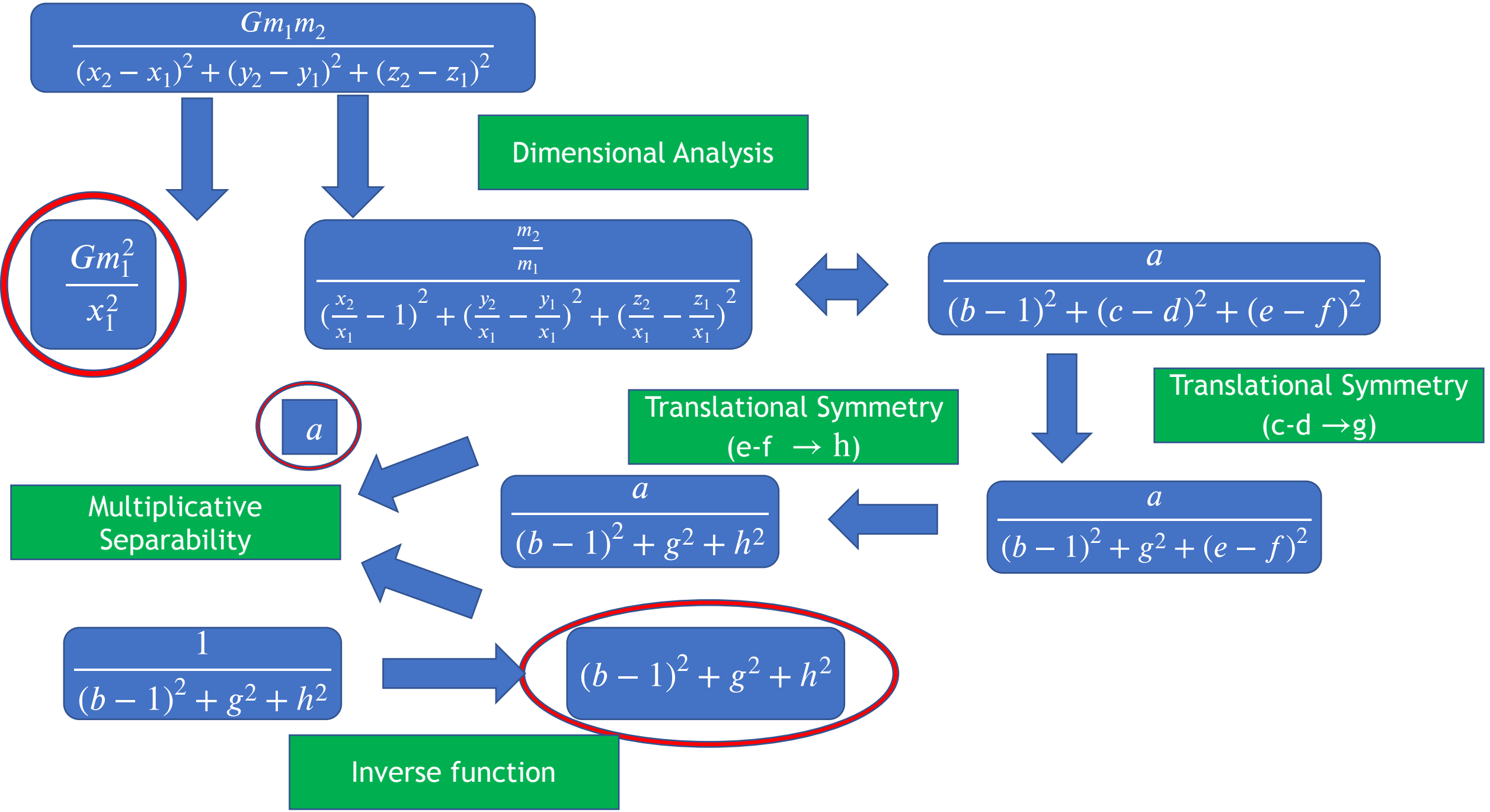
Translational Symmetry
(c-d →g)

$$\frac{a}{(b-1)^2+g^2+(e-f)^2}$$

Translational Symmetry
(e-f → h)

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


$\frac{Gm_1m_2}{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}$
Dimensional Analysis
$\frac{Gm_1^2}{x_1^2}$
$\frac{\frac{m_2}{m_1}}{(\frac{x_2}{x_1}-1)^2+(\frac{y_2}{x_1}-\frac{y_1}{x_1})^2+(\frac{z_2}{x_1}-\frac{z_1}{x_1})^2}$
$\frac{a}{(b-1)^2+(c-d)^2+(e-f)^2}$
Translational Symmetry
(c-d →g)
$\frac{a}{(b-1)^2+g^2+(e-f)^2}$
Translational Symmetry
(e-f → h)
$\frac{a}{(b-1)^2+g^2+h^2}$
a
Multiplicative
Separability
$\frac{1}{(b-1)^2+g^2+h^2}$


$\frac{Gm_1m_2}{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}$
Dimensional Analysis
$\frac{Gm_1^2}{x_1^2}$
$\frac{\frac{m_2}{m_1}}{(\frac{x_2}{x_1}-1)^2+(\frac{y_2}{x_1}-\frac{y_1}{x_1})^2+(\frac{z_2}{x_1}-\frac{z_1}{x_1})^2}$
$\frac{a}{(b-1)^2+(c-d)^2+(e-f)^2}$
Translational Symmetry
(c-d →g)
$\frac{a}{(b-1)^2+g^2+(e-f)^2}$
Translational Symmetry
(e-f → h)
$\frac{a}{(b-1)^2+g^2+h^2}$
a
Multiplicative
Separability
$\frac{1}{(b-1)^2+g^2+h^2}$
$(b-1)^2+g^2+h^2$
Inverse function

Results

- Our algorithm discovered all 100 equations, with a run time between 10 seconds and 1 hour for each equation.
- The Eureqa software discovered 71 equations.

$$f = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}}$$

$$\mathrm{F} = \frac{Gm_1m_2}{(x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2}$$

$$E = \frac{3}{4\pi\epsilon}\frac{pz}{r^5}\sqrt{x^2+y^2}$$

$$x = \sqrt{x_1^2+x_2^2-2x_1x_2\cos(\theta_1-\theta_2)}$$

$$L = \frac{\hbar\omega^3}{\pi^2c^2(e^{\hbar\omega/k_bT}-1)}$$

$$P = \frac{pE}{\hbar}\frac{\sin\left(\frac{(\omega-\omega_0)t}{2}\right)^2}{\left(\frac{(\omega-\omega_0)}{2}\right)^2}$$

Results

- Adding noise
  - Gaussian noise with standard deviation ϵy_{rms}
  - 95% of the equations get solved for $\epsilon = 10^{-4}$
  - 40% of the equations get solved for $\epsilon = 10^{-2}$
- Reduced data
  - 82% of the equations get solved using 10 data points
  - 95% of the equations get solved using 100 data points
  - $10^2 - 10^5$ data points needed for equations involving the neural network

Removing Dimensional Analysis

- 93% of the equations get solved.
- Solved examples:

$$F = \frac{Gm_1m_2}{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2} \qquad f = \frac{\mu_m H}{k_b T} + \frac{\mu_m \alpha M}{\epsilon c^2 k_b T} \qquad E_n = \frac{\hbar\omega}{e^{\frac{\hbar\omega}{k_b T}} - 1}$$

- Unsolved examples:

$$t_1 = \frac{t - \frac{ux}{c^2}}{\sqrt{1 - \frac{v^2}{c^2}}} \qquad I_* = I_1 + I_2 + 2\sqrt{I_1 I_2}\cos(\delta)$$

$$M = n_\rho \mu_M \tanh\left(\frac{\mu_M B}{k_b T}\right) \qquad t_1 = \frac{t - ux/c^2}{\sqrt{1 - u^2/c^2}}$$

Test on harder equations

- Pick a new set of 20 equations and test the performance of the code.

$$\frac{d\sigma}{d\Omega} = \left(\frac{Z_1 Z_2 \alpha \hbar c}{4E \sin^2 \frac{\theta}{2}} \right)^2$$

$$\frac{d\sigma}{d\cos\theta} = \frac{\pi \alpha^2 \hbar^2}{m^2 c^2} \left(\frac{\omega'}{\omega} \right)^2 \left(\frac{\omega'}{\omega} + \frac{\omega}{\omega'} - \sin^2\theta \right)$$

$$\theta_0 = \arccos\left(\frac{\cos\theta_s - \frac{v}{c}}{1 - \frac{v}{c}\cos\theta_s} \right)$$

$$\frac{1}{r} = \frac{mk}{L^2} \left(1 + \sqrt{1 + \frac{2EL^2}{mk^2}} \cos\left(\theta_1 - \theta_2\right) \right)$$

$$t = \frac{2\pi a^{\frac{3}{2}}}{\sqrt{G(m_1 + m_2)}}$$

$$\rho_0 = -\frac{1}{8\pi G} \left[\frac{k}{R_0^2} + H_0^2 \left(1 - 2q_0\right) \right]$$

$$I = I_0 \left(\frac{\sin\left(\frac{\phi}{2}\right)}{\frac{\phi}{2}} \frac{\sin\left(\frac{N\delta}{2}\right)}{\sin\left(\frac{\delta}{2}\right)} \right)^2$$

$$P = -\frac{\frac{32}{5} G^4}{c^5} \frac{\left(m_1 m_2\right)^2 (m_1 + m_2)}{r^5}$$

$$H = \sqrt{\left(p - qA\right)^2 c^2 + m^2 c^4} + qV$$

$$\omega' = \frac{\sqrt{1 - \frac{v^2}{c^2}}\omega}{1 + \frac{v}{c}\cos\theta}$$

$$v = \sqrt{\frac{2}{m}\left(E - V - \frac{L^2}{2mr^2} \right)}$$

$$r = \frac{a(1 - e^2)}{1 + e\cos(\theta_1 - \theta_2)}$$

$$V = E\cos\theta \left(-r + \frac{\epsilon - 1}{\epsilon + 2} \frac{a^3}{r^2} \right)$$

Test on harder equations

- We discover the exact expression for 18 out of 20 equations.

$$P = -\frac{\frac{32}{5}G^4}{c^5}\frac{\left(m_1 m_2\right)^2(m_1+m_2)}{r^5} \qquad F = \frac{q}{4\pi\epsilon y^2}(4\pi\epsilon Va - \frac{qay^3}{\left(y^2-a^2\right)^2})$$

- Eureqa is able to discover only 3 out of the 20 equations.

$$H = \sqrt{\frac{8\pi G}{3}\rho - \frac{kc^2}{a^2}} \qquad H = \frac{1}{2m}\left[p^2 + m^2\omega^2 q^2\left(1+\frac{\epsilon q}{q_0}\right)\right] \qquad \rho = \frac{3}{8\pi G}\left(\frac{k}{R_0^2}+H_0^2\right)$$

Ideas for further improvement

- A fine tuning of the hyperparameters can give significantly better results.
- A genetic algorithm can be combined with our algorithm.
- Using natural units could improve the performance of the algorithm.
- Use the a power symbol “^” or a square symbol “Q” in the brute force code.
- Other properties of the functions can be included (e.g. rotational symmetry).
- Other 2 (or more) variables combinations can be searched for (e.g. xy^3, $\frac{\log(x)}{\cos(y)}$).

Conclusions

- We took advantage of the properties of functions of practical interest to build a new algorithm for symbolic regression.
- Using the dimensions of the variables and breaking the functions apart using a neural network allow us to find the expression for complicated function.
- We obtained significantly better results than previous methods, based on genetic algorithms.
- The code gives good results even with a significant amount of noise or little data.
- We look forward to the day AI discovers a useful new physics equation!

AI for physics

Physics for AI

Thank you!

Backup Slides

Dimensional Analysis - Algorithm

- For each equation $y = f(x_1,\ x_2, \ldots, x_n)$ build a matrix **M** whose i^{th} column is the **u**-vector corresponding to $\boldsymbol{x_i}$.
- Define **b** as the **u**-vector corresponding to y.
- Solve $\boldsymbol{Mp = b}$ for $\boldsymbol{p}$.
- Build the matrix **U**, having as columns a basis for the null space of **M**.
- Define a new mystery: $y' = f'(x_1',\ x_2', \ldots, x_{n'}')$ where:

$$x_i' = \prod_{i=j}^{n} x_j^{U_{ij}},\ y' = \frac{y}{y^*},\ y^* = \prod_{i=1}^{n} x_i^{p_i}$$

- By construction, x_i' and y' are dimensionless and the number of variables is equal to the dimension of the null space.
- E.g. $m = \dfrac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}} \rightarrow \dfrac{m}{m_0} = \dfrac{1}{\sqrt{1 - \frac{v^2}{c^2}}} \rightarrow y = \dfrac{1}{\sqrt{1 - x^2}}$

Polynomial fit

- Try to fit a polynomial to a given equation.
- Polynomials are simple, but require many symbols ⇒ too difficult to use brute force.
- Many functions in physics are represented by low order polynomials:

$$K = \frac{1}{2}m\left(v^2 + u^2 + w^2\right)$$

or contain parts that are polynomials:

$$\mathrm{F} = \frac{Gm_1m_2}{\left(x_1 - x_2\right)^2 + \left(y_1 - y_2\right)^2 + \left(z_1 - z_2\right)^2}$$

- Pick a maximum degree and a maximum value for the MSE of the fit.

Brute force

- For a list of symbols, try all the mathematically allowed combinations of the symbols and variables until the fitting error dropped below a certain threshold ϵ or after a maximum runtime t_{max} has been exceeded.
- Use reverse polish notation (doesn't need parentheses).
  - $a - b * c + 1 \longrightarrow abc * - 0 > +$
- Increase complexity by adding more symbols to the list.
- Ignore multiplicative or additive numerical constant: $\frac{1}{4\pi\epsilon}\frac{Q}{r^2} \rightarrow \frac{Q}{\epsilon r^2}$.
- Avoid overfitting by picking the solution with the smallest description length: $DL \equiv \log_2[N \bullet max(1, \frac{\epsilon}{\epsilon_d})]$, with ϵ - brute force error, $\epsilon_d = 10^{-15}$ and N - rank of the symbols string.

Association for Computing Machinery

Advancing Computing as a Science & Profession

Ubiquity, an ACM publication
May 2017

Cybersecurity Skeptics Now Embracing Formal Methods

An Interview with Gernot Heiser and Jim Morris

by Ted G. Lewis

Editor's Introduction

There is new hope for those who despair securing computer systems from external hackers. The recent DARPA HACMS project demonstrated conclusively that "certain pathways for attackers have all been shut down in a way that's mathematically proven to be unhackable for those pathways." Continuing research at DARPA and IARPA will eventually shut down all the pathways, and the external hackers will be out of business permanently.

Old majority view:

- *100% security impossible*
- *Formal verification methods useless in practice*

New hope:

- *DARPA HACMS project demonstrated conclusively that "certain pathways for attackers have all been shut down in a way that's mathematically proven to be unhackable"*

seL4

Security. Performance. Proof.

Association for Computing Machinery
Advancing Computing as a Science & Profession

Ubiquity, an ACM publication
May 2017

Cybersecurity Skeptics Now Embracing Formal Methods

An Interview with Gernot Heiser and Jim Morris

by Ted G. Lewis

Editor's Introduction

There is new hope for those who despair securing computer systems from external hackers. The recent DARPA HACMS project demonstrated conclusively that "certain pathways for attackers have all been shut down in a way that's mathematically proven to be unhackable for those pathways." Continuing research at DARPA and IARPA will eventually shut down all the pathways, and the external hackers will be out of business permanently.

Old majority view:

- *100% security impossible*
- *Formal verification methods useless in practice*

New hope:

- *DARPA HACMS project demonstrated conclusively that "certain pathways for attackers have all been shut down in a way that's mathematically proven to be unhackable"*

We'll never need to install security patches for sel4. Ever.