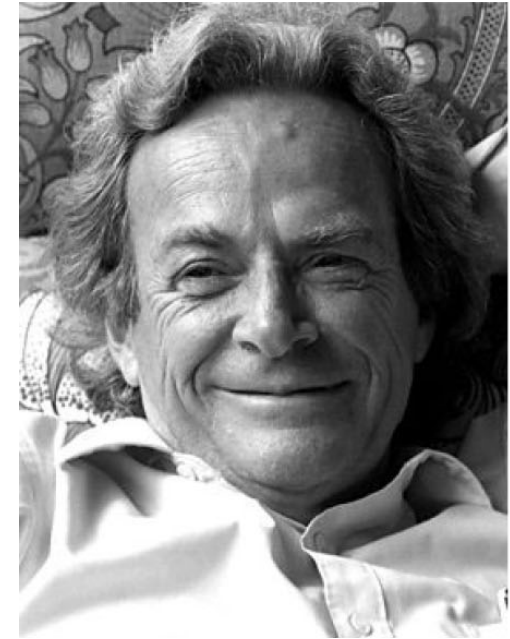


AI for physics & physics for AI



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

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

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

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



Max Tegmark



Massachusetts
Institute of
Technology



Institute for Artificial Intelligence
and Fundamental Interactions



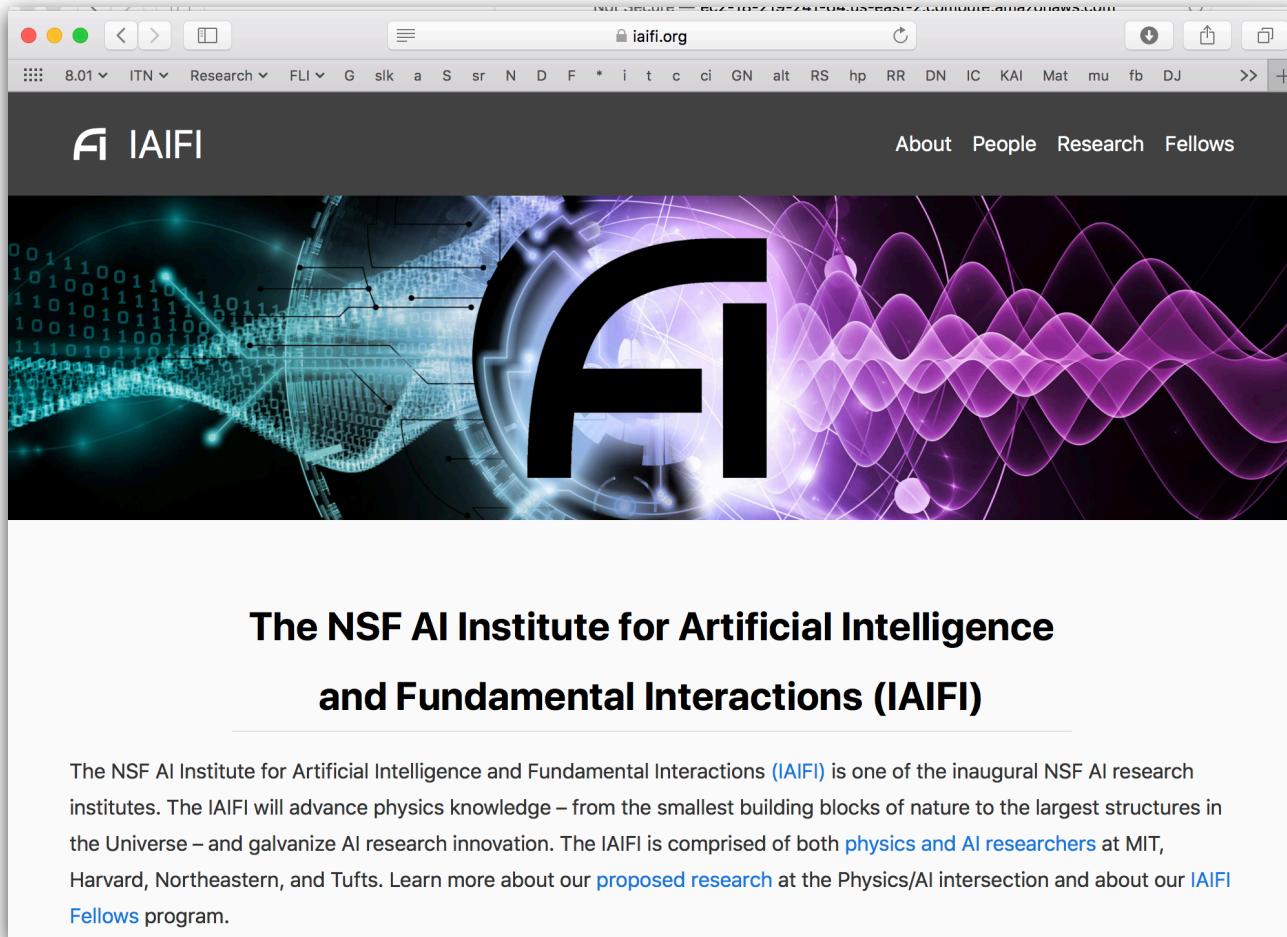
CENTER FOR
Brains
Minds+
Machines

**AI
for
physics**

**Physics
for
AI**

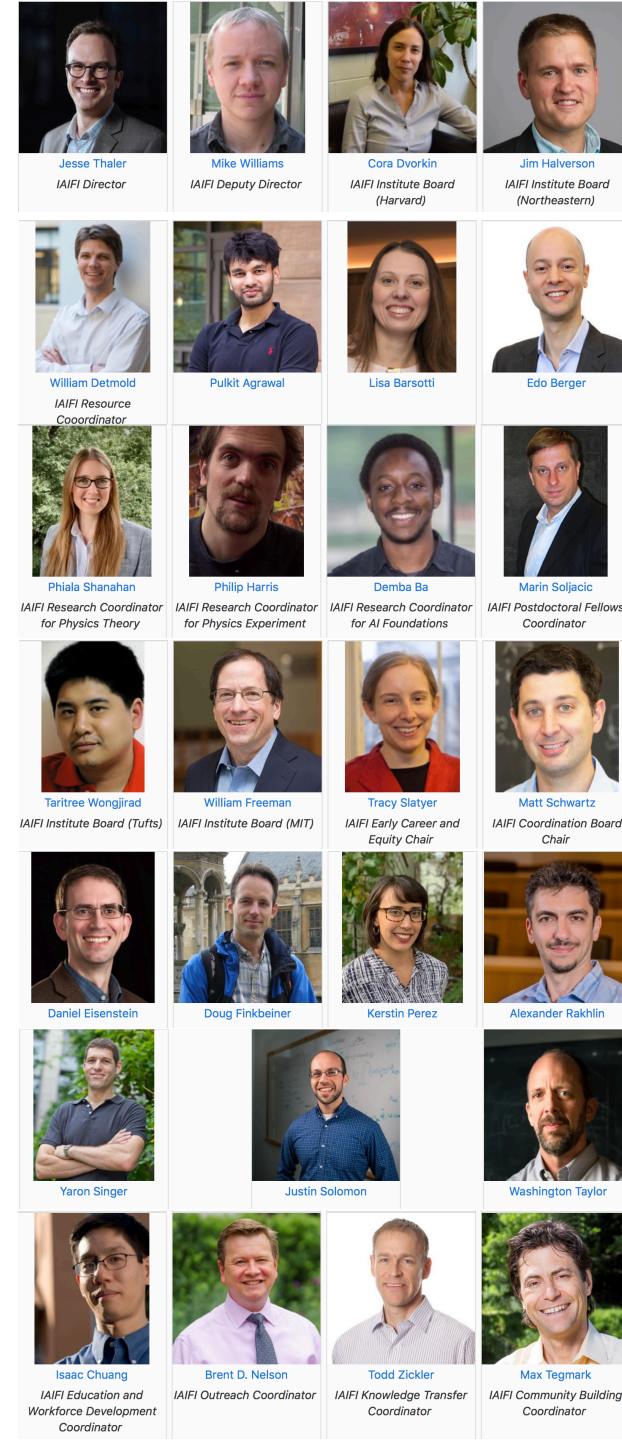
AI + physics = IAIFI

IAIFI.org



The NSF AI Institute for Artificial Intelligence and Fundamental Interactions (IAIFI)

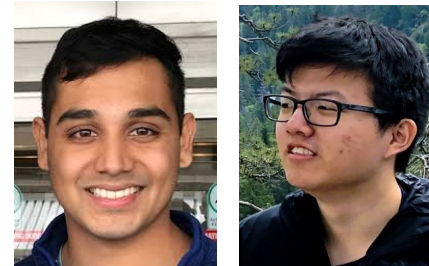
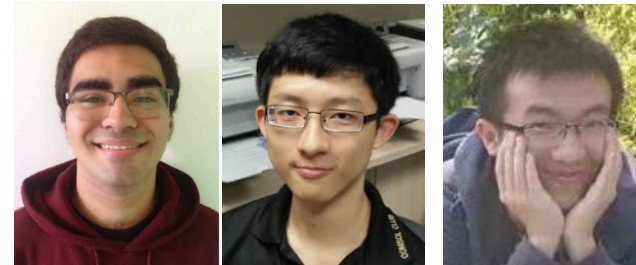
The NSF AI Institute for Artificial Intelligence and Fundamental Interactions (IAIFI) is one of the inaugural NSF AI research institutes. The IAIFI will advance physics knowledge – from the smallest building blocks of nature to the largest structures in the Universe – and galvanize AI research innovation. The IAIFI is comprised of both physics and AI researchers at MIT, Harvard, Northeastern, and Tufts. Learn more about our proposed research at the Physics/AI intersection and about our IAIFI Fellows program.



Please apply to MIT!



AI for physics & physics for AI



Please apply
to MIT!



AI OPPORTUNITIES

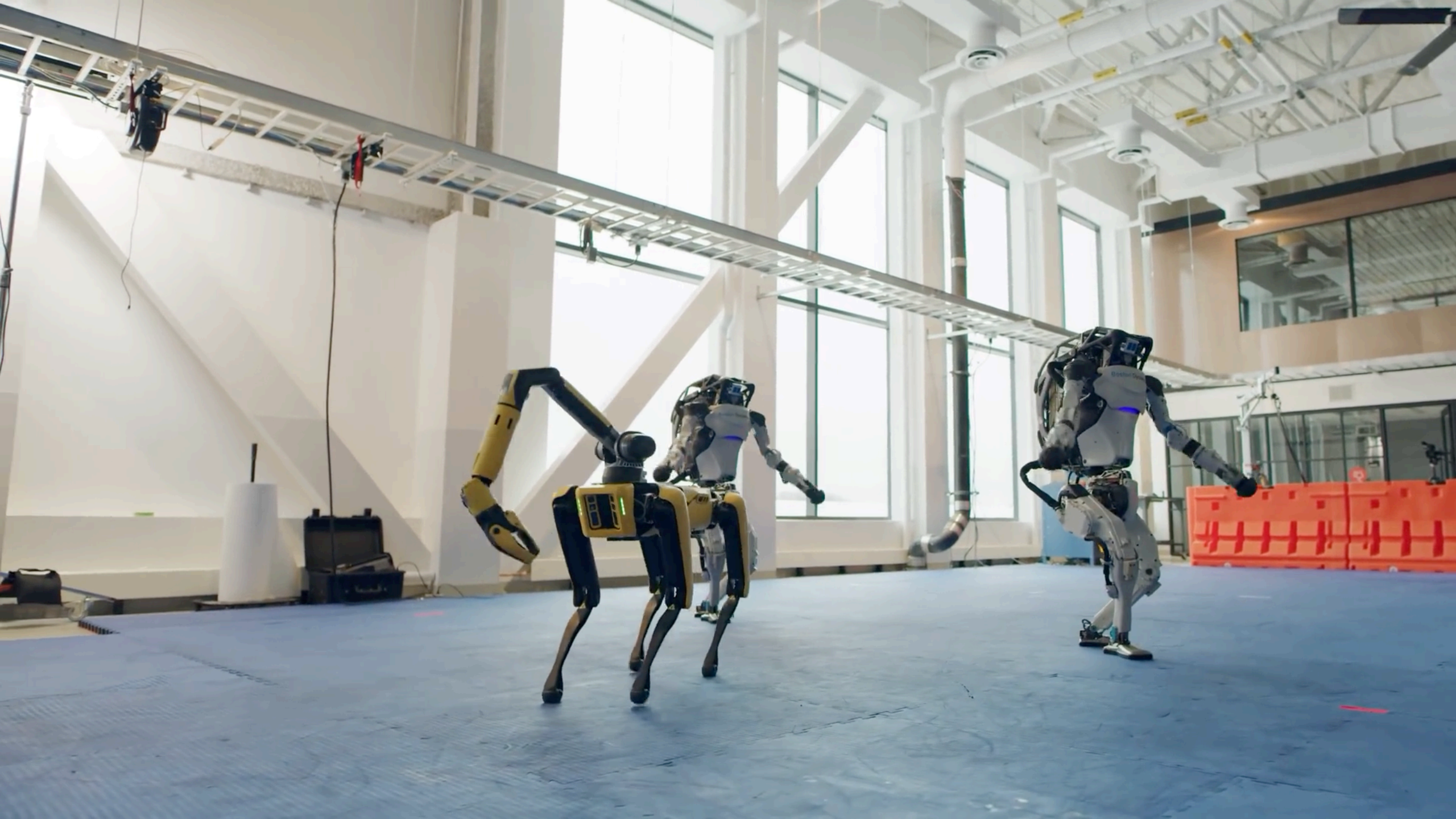
IRPLEX

FAIRPLEX

FAIRPLEX



6:16:34 05/06/2015

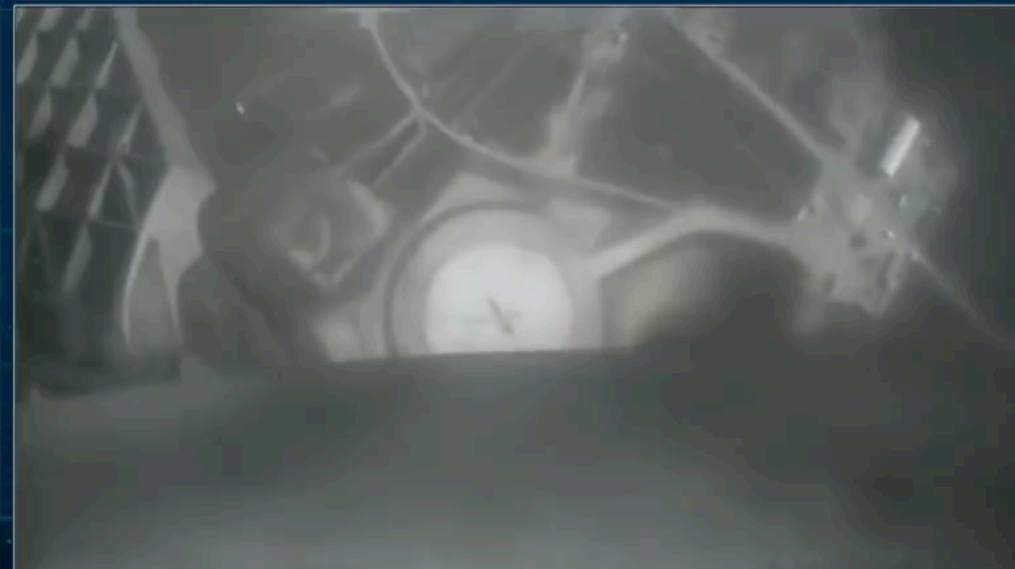


T+ 00:07:58

STAGE 2 TELEMETRY

SPEED

ALTITUDE

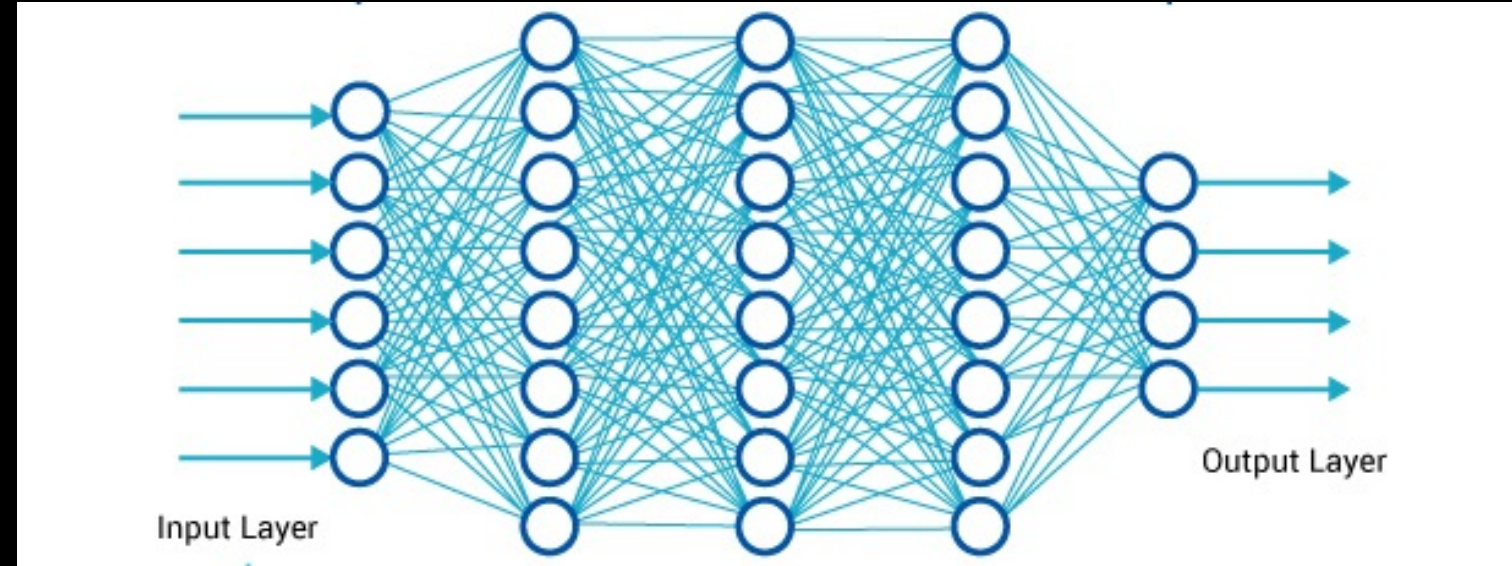


FALCON HEAVY TEST FLIGHT

SpaceX

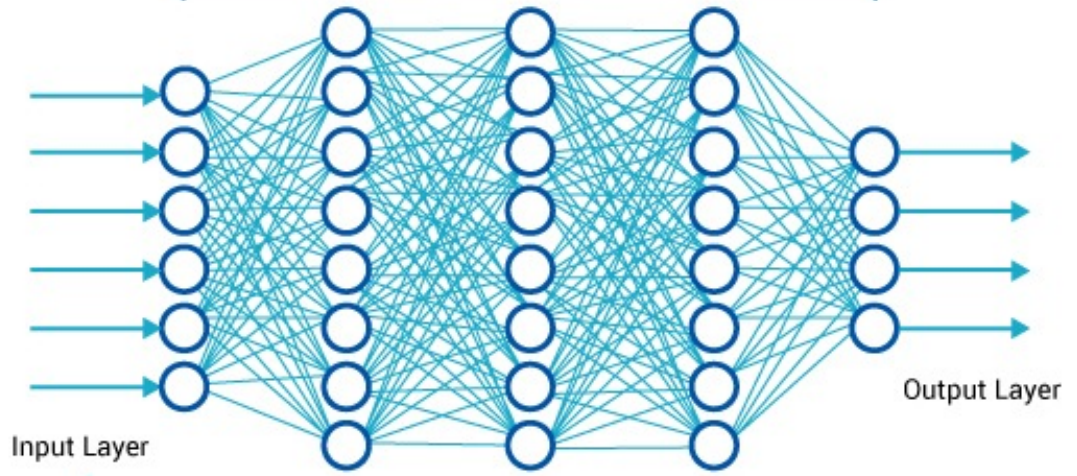
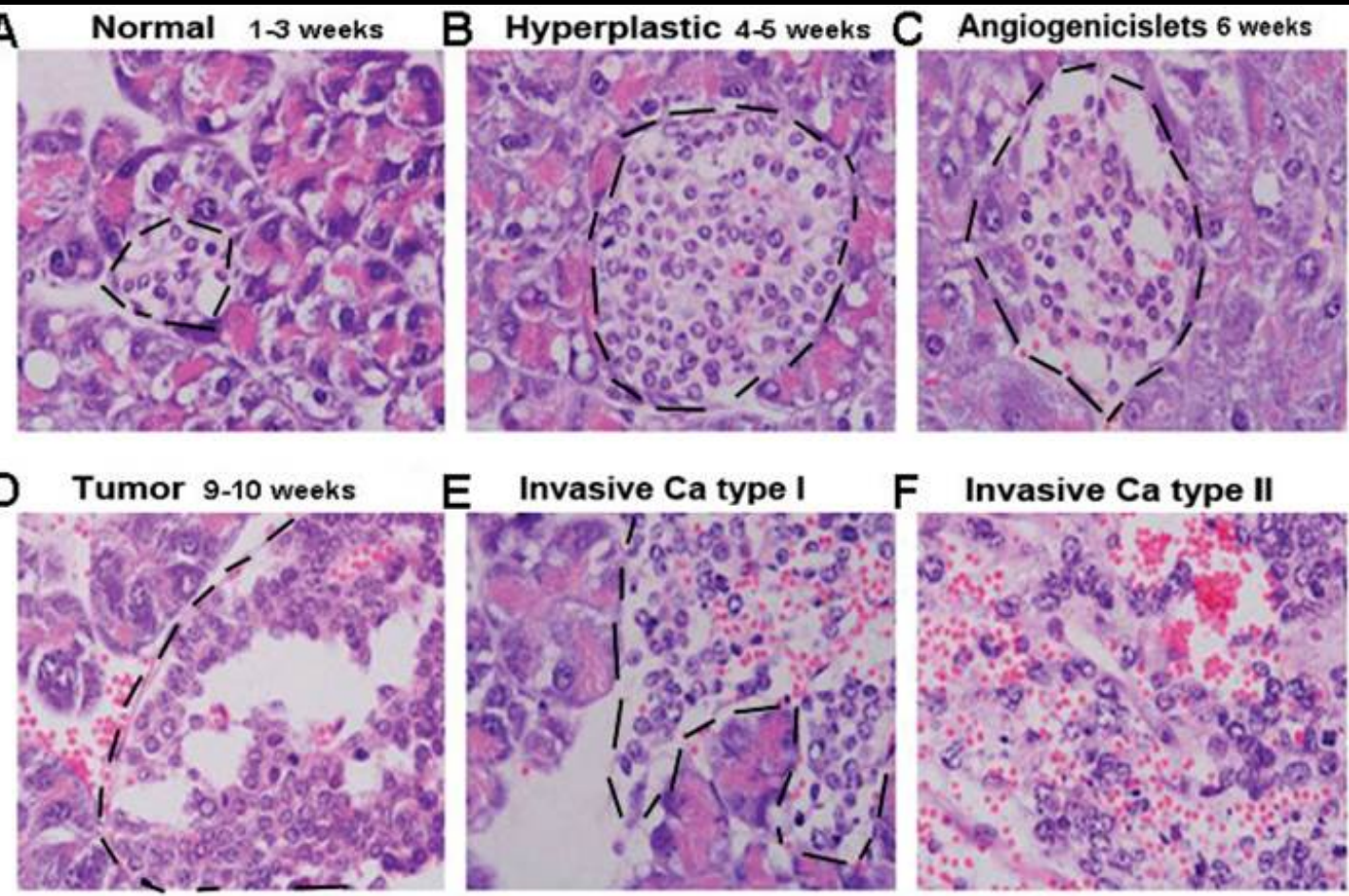


AI diagnosis - prostate cancer



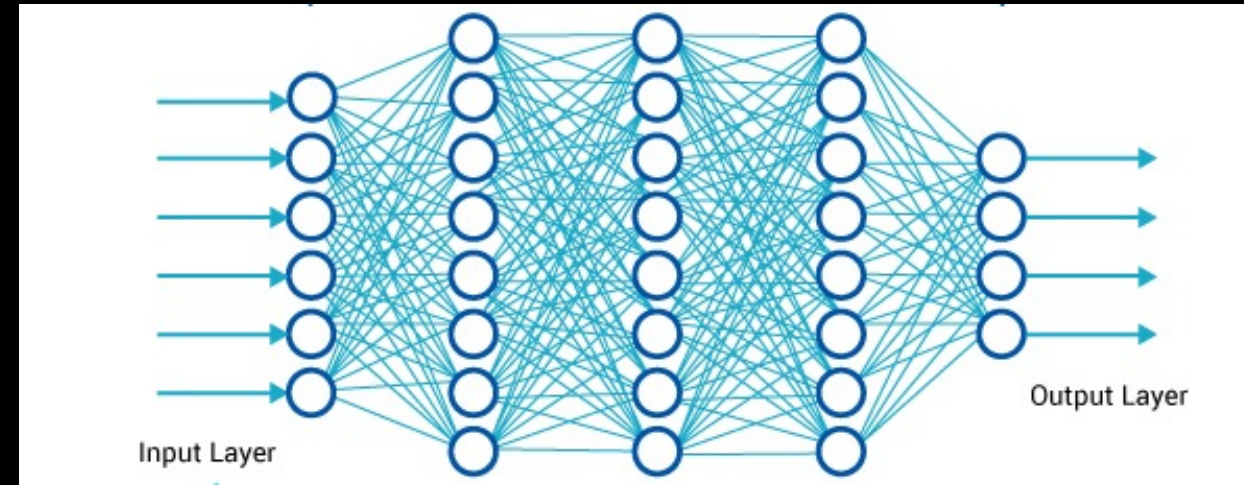
A recent Dutch study showed that AI diagnosis of prostate cancer using MRI was as good as that of human radiologists

AI diagnosis - lung cancer



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

AI diagnosis - blindness

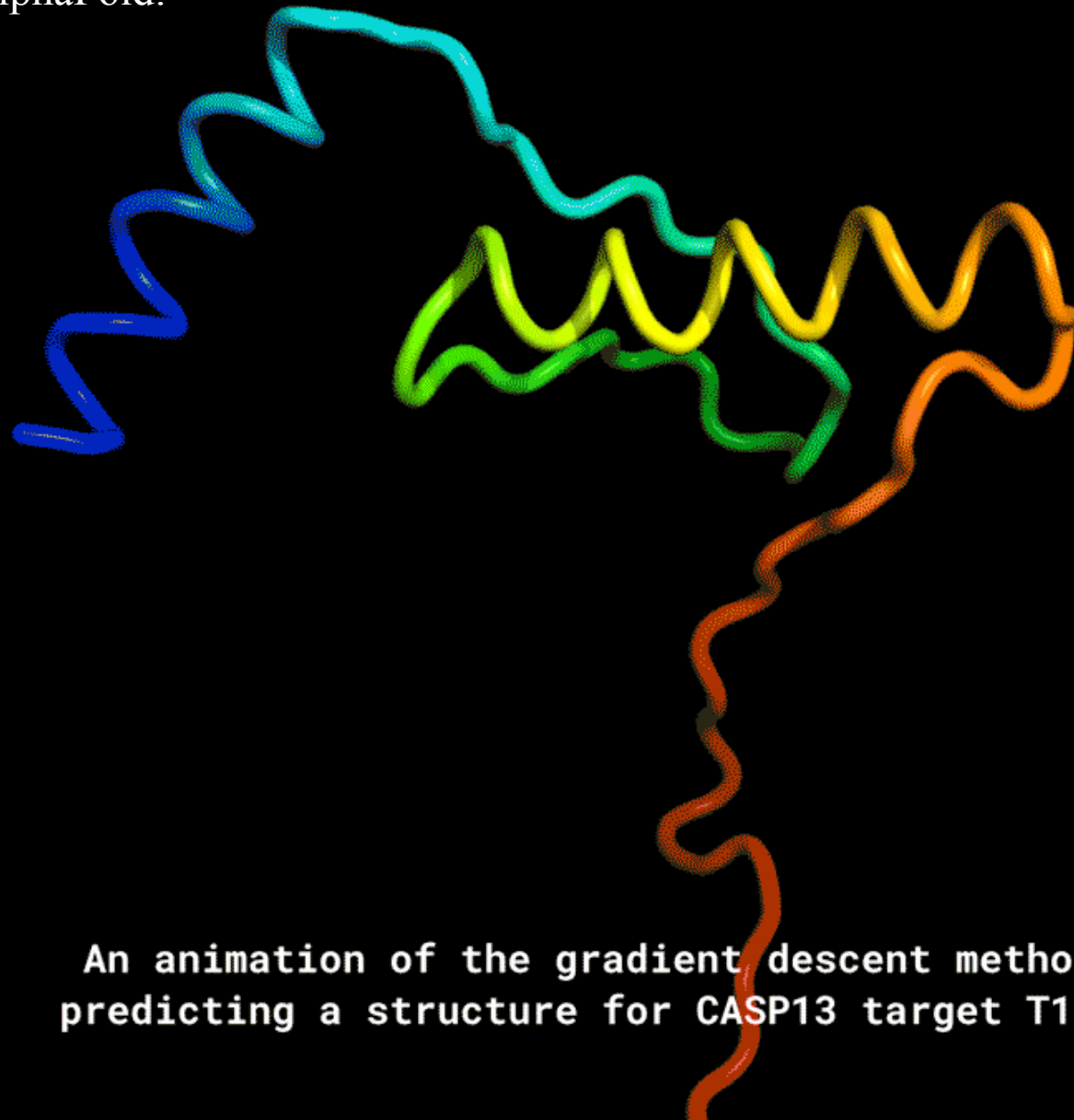


A recent Google DeepMind study showed that AI could diagnose 50 retinal diseases from optical coherence tomography as human optalmologists

Input amino acid sequence:

TDELLERLRQLFEELHERGTEIVVEVHINGERDEIRVRNISKEELKKLLERIREKIEREGSSEVEVNVHSGGQTWTFNEK

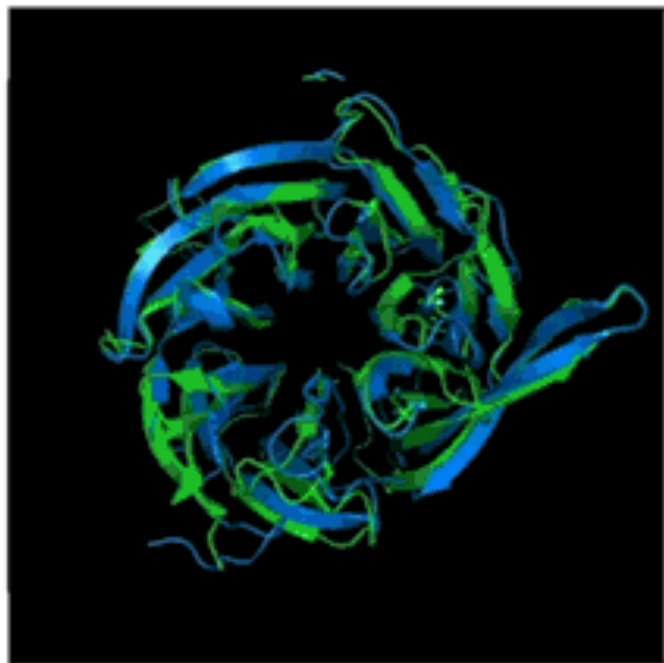
Protein shape predicted by AlphaFold:



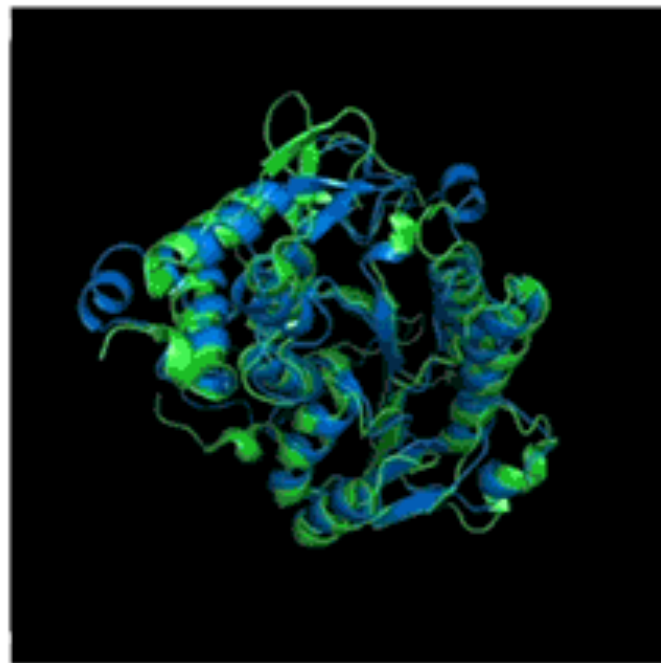
An animation of the gradient descent method
predicting a structure for CASP13 target T1008

Structures:
Ground truth (green)
Predicted (blue)

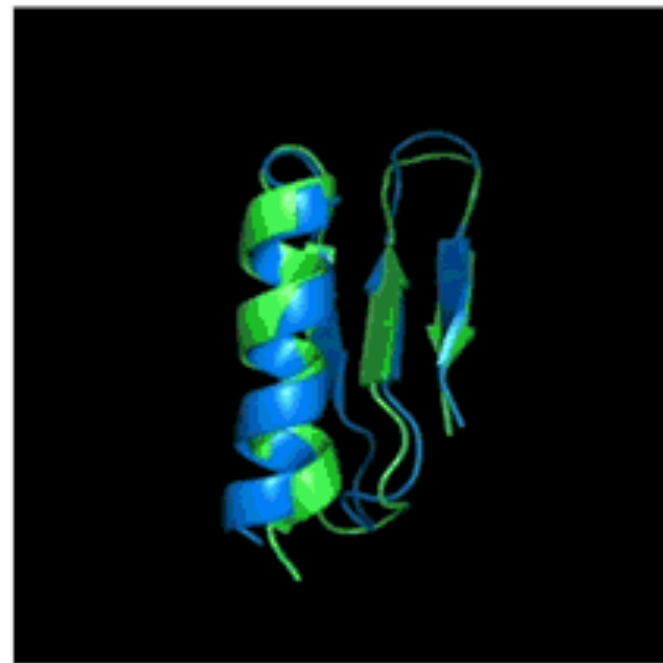
T0954 / 6CVZ



T0965 / 6D2V

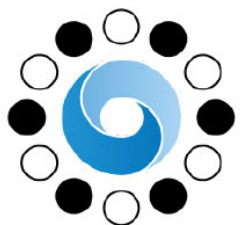


T0955 / 5W9F

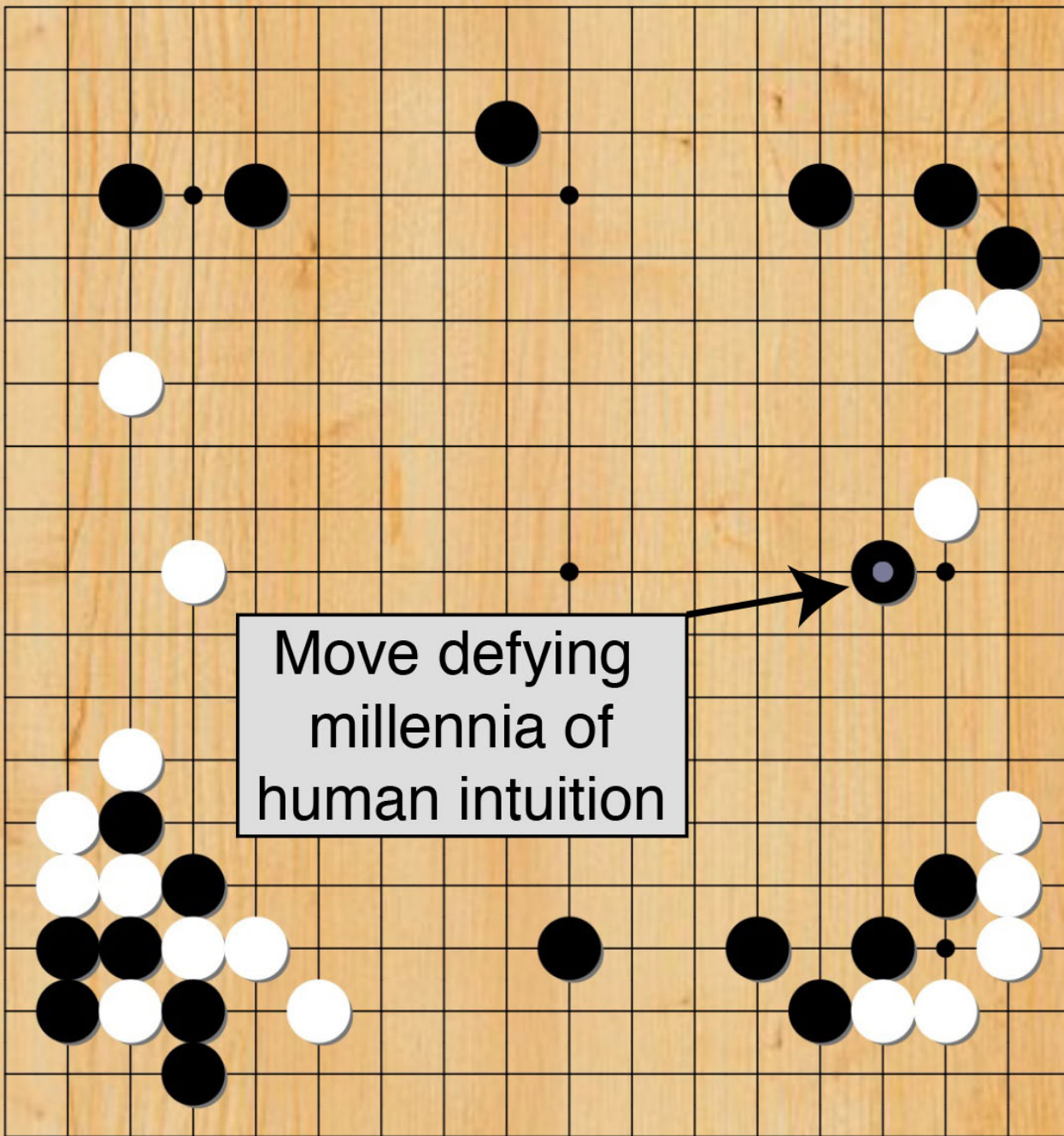


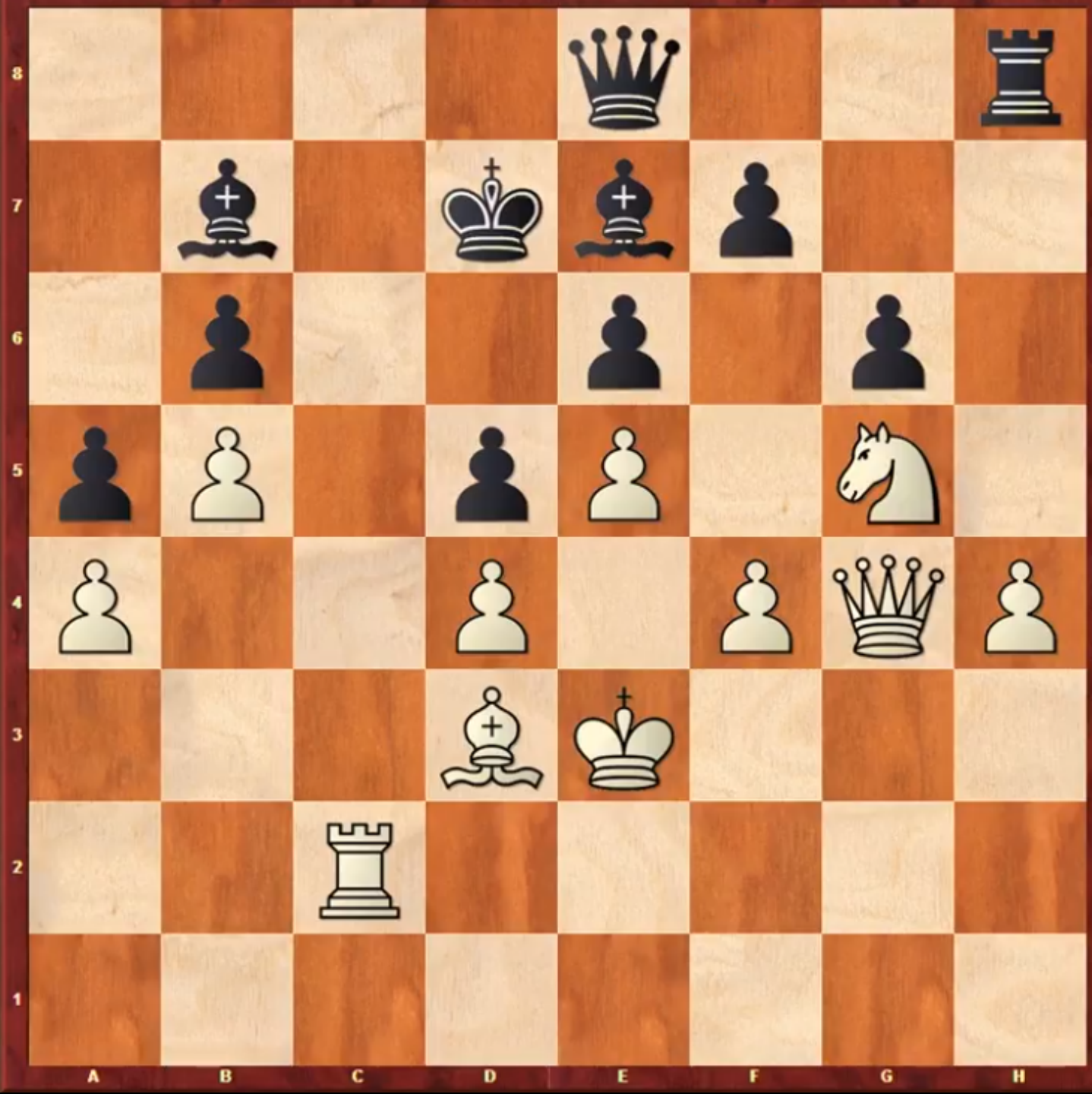


vs.



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 FOR PHYSICS

AI for physics examples @ MIT physics department

Aram Harrow

Using quantum computers to improve machine learning and optimization algorithms

Christoph Paus

With my students and postdocs, we use various machine learning techniques to push our CMS data analysis and squeeze the most out of the data. Specifically, we work on lepton identification, boosted particle reconstruction including subjet structure information, improved mass and energy reconstruction.

Eddie Farhi

Quantum computing and machine learning

Ike Chuang

Understanding the physics of machine learning, including natural realizations in quantum systems and noisy classical systems

Jackie Hewitt

Analysis and interpretation of 21 cm cosmology data from HERA

Jeremy England

The non-equilibrium statistical mechanics of neural networks

Jesse Thaler

Studying LHC events using techniques from weak supervision & topic modeling. In the process, discovered linear basis for collider observables, allowing linear regression to match the classification performance of deep neural networks.

Liang Fu

Modeling many-body systems near phase transitions with machine learning-accelerated Monte Carlo simulation

Marin Soljačić

Novel AI techniques to help with physics research. Physics-inspired algorithms for better machine learning. Novel hardware for AI.

Markus Klute

Machine learning tools for reconstructing, monitoring, and analyzing experimental particle physics data CMS

Matt Evans

LIGO event detection and analysis with machine learning

Max Tegmark

Modeling phase transitions with deep learning. Using physics-inspired techniques to make deep learning algorithms more efficient, transparent and trustworthy.

Mike Williams

Machine learning tools for analyzing experimental particle physics data from LHCb & CMS

Philip Harris

Real-time LHC data reconstruction using FPGA-based DNN inference; Higgs coupling extraction via adversarial NNs for automated tuning of MC QCD simulations

Sarah Seager

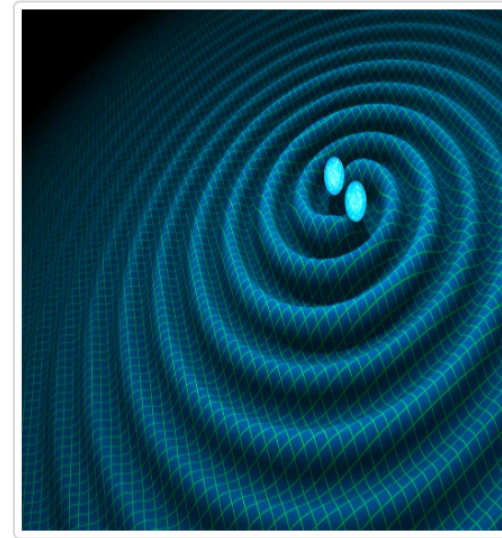
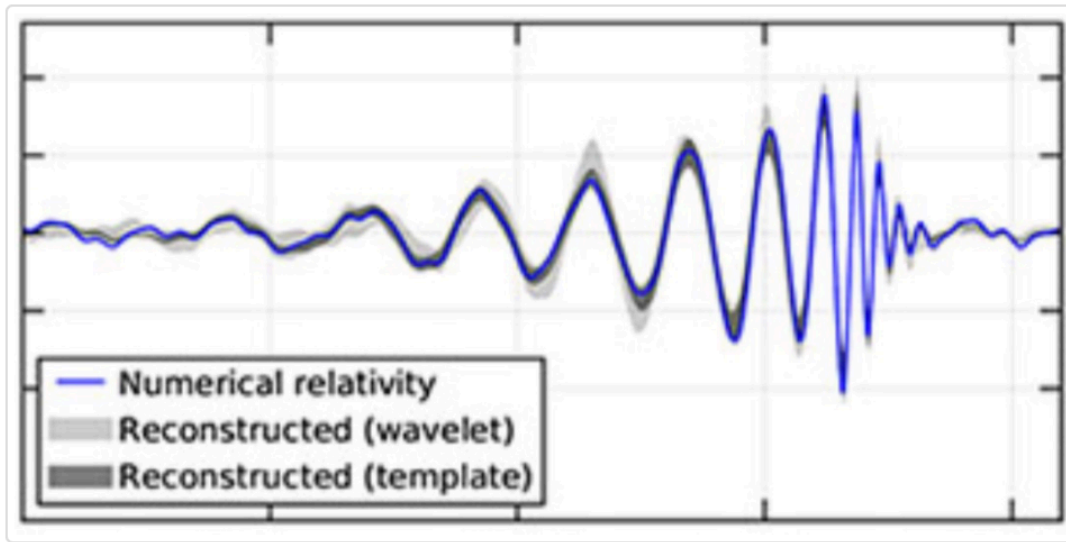
Machine learning for finding exoplanets in large data sets

Seth Lloyd

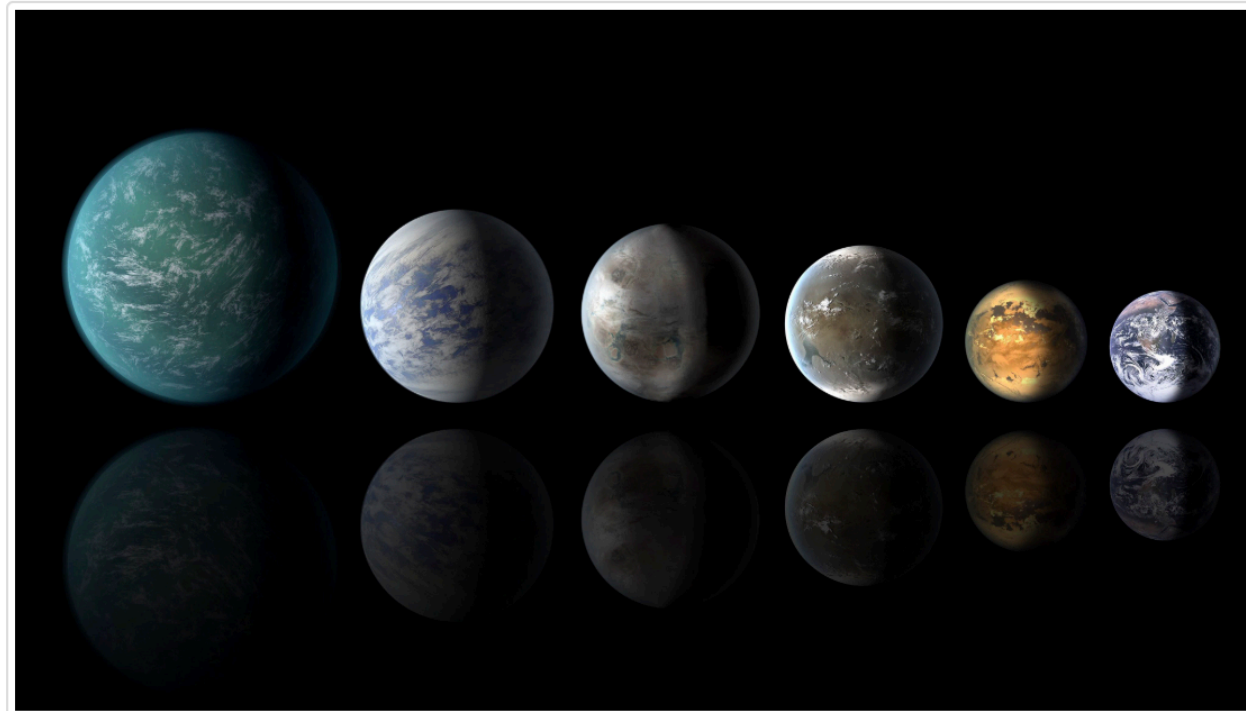
Quantum machine learning

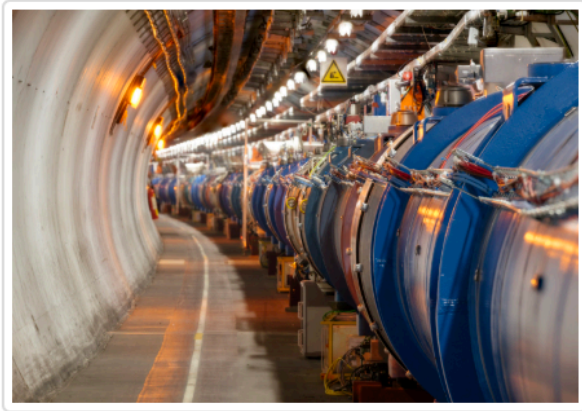
Will Detmold

Use machine learning to speed up lattice QCD calculations for particle and nuclear physics (e.g. arXiv:1801.05784)

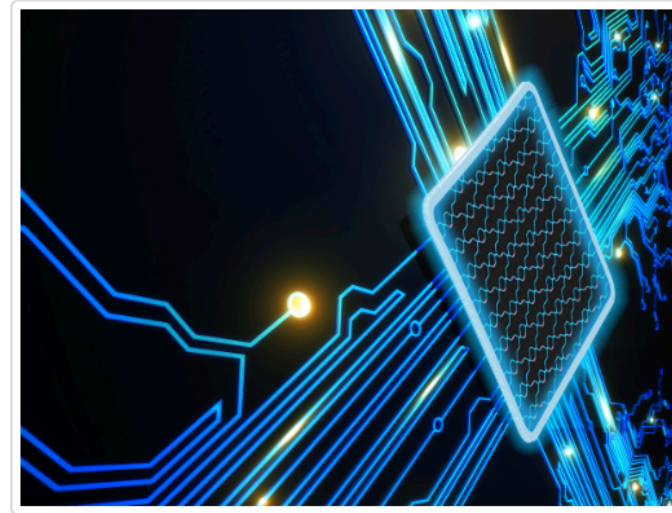
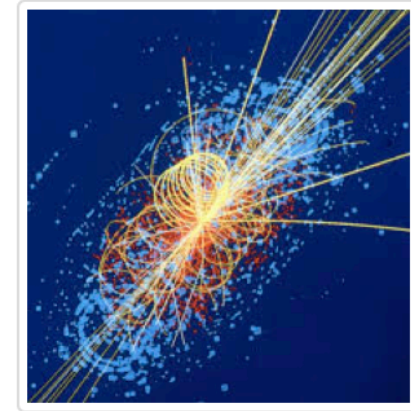


We use machine-learning techniques to detect extrasolar planets and gravitational waves from colliding black holes.

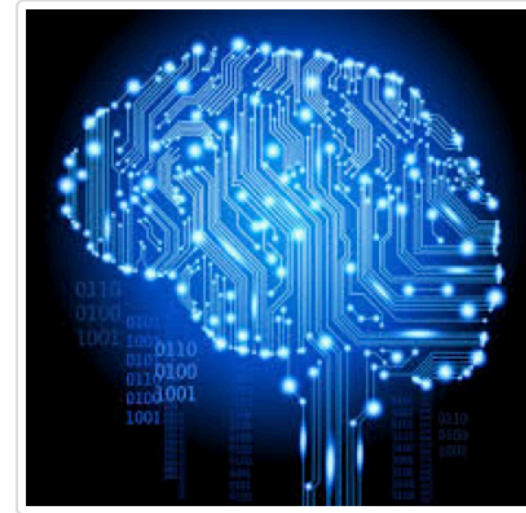




We're using machinelearning tools to analyze particle physics data from the Large Hadron Collider.



We're developing technology for faster and more energy-efficient deep learning using optical chips, that compute using photons instead of electrons.



We're using techniques from condensed matter physics to help understand how our brains process information,

**PHYSICS
FOR
AI**

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





What caused the Ethiopian Airlines plane crash?

0:00 / 1:24 

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.

Every single Yahoo account was hacked - 3 billion in all

by Selena Larson @selenalarson

🕒 October 4, 2017: 6:36 AM ET



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.

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 Golovnirov/SOPA Images/LightRocket via Getty Images) GETTY

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

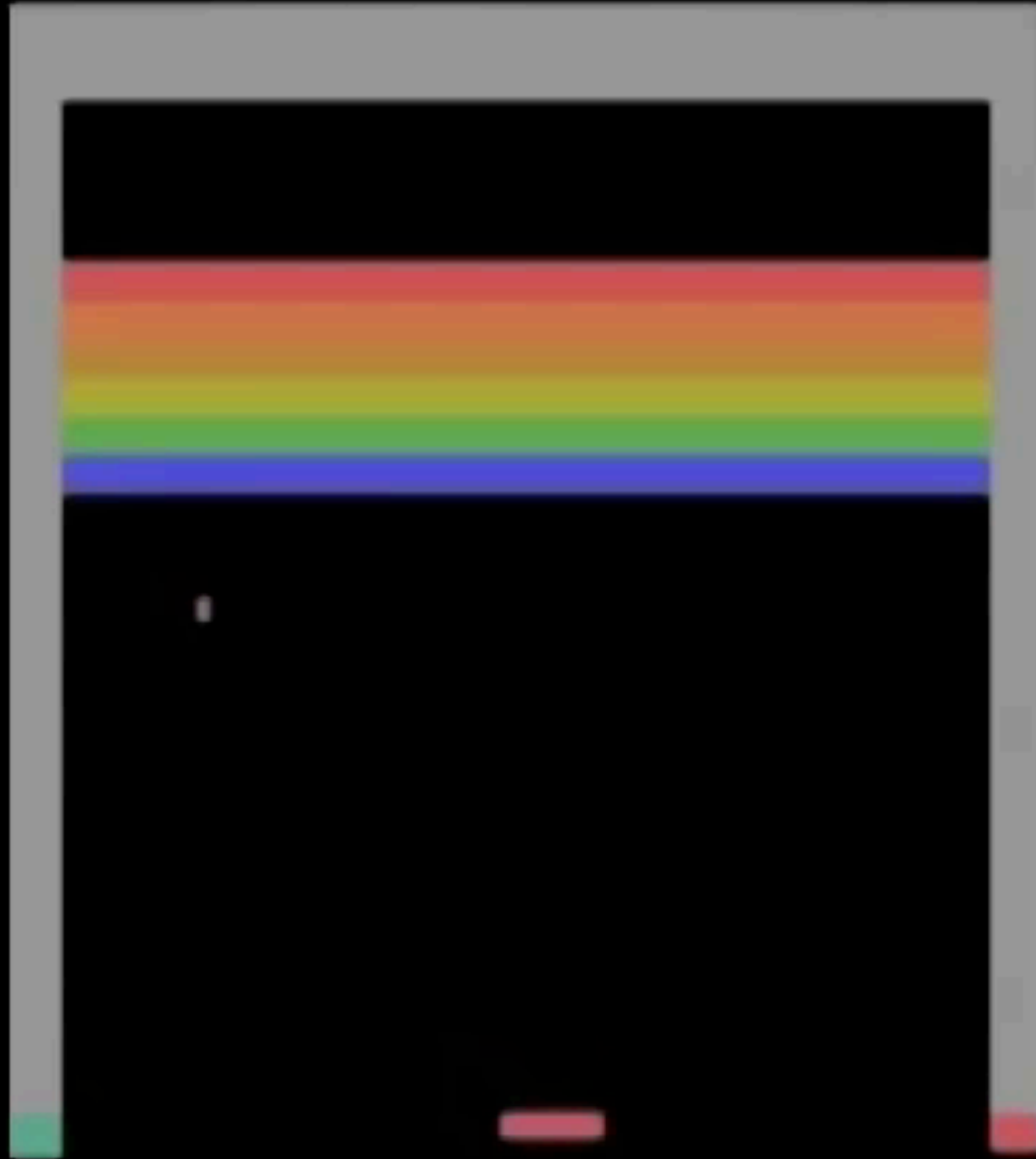
2



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.

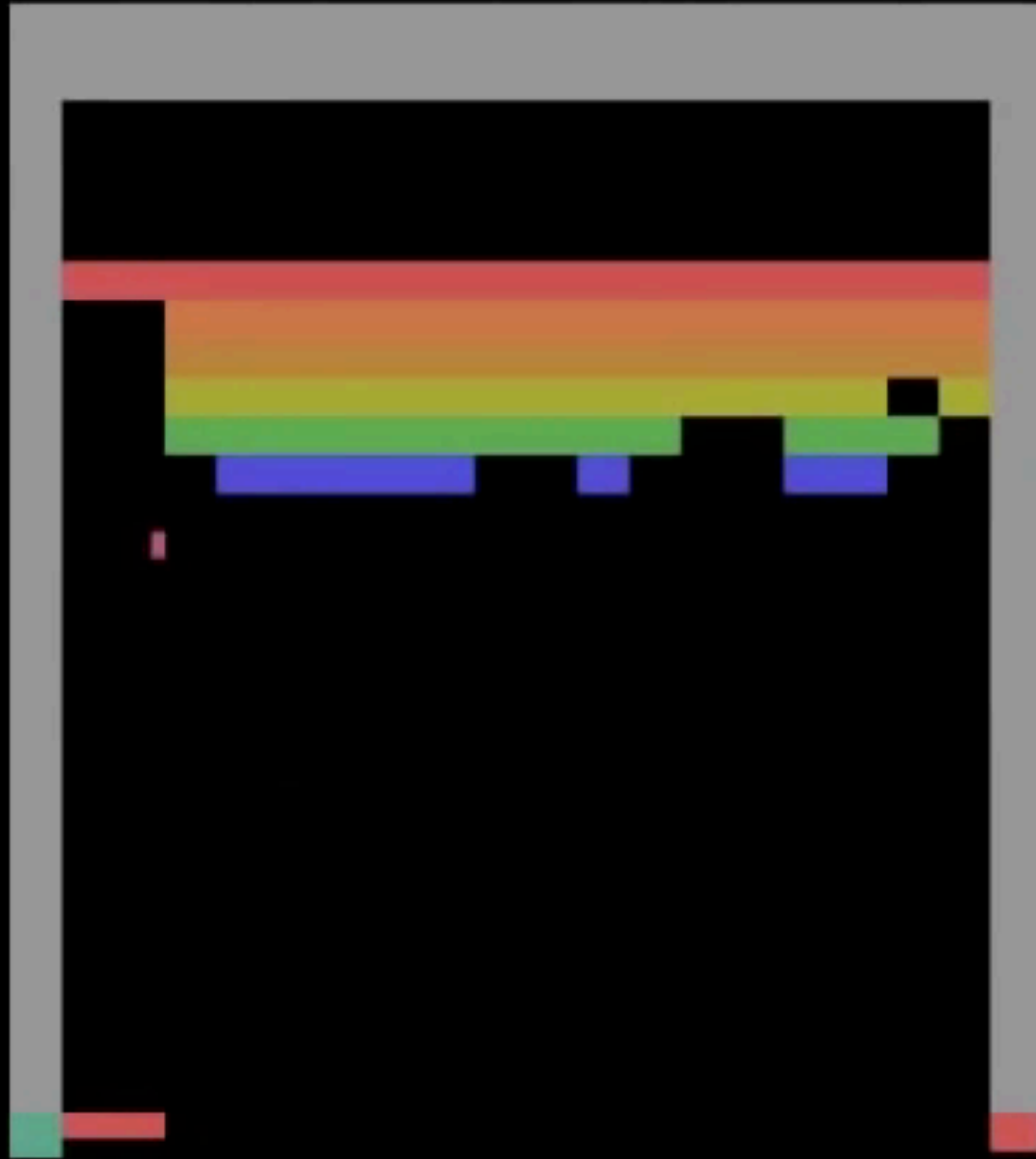
Photo: Mark Wilson (Getty)

000 5 1

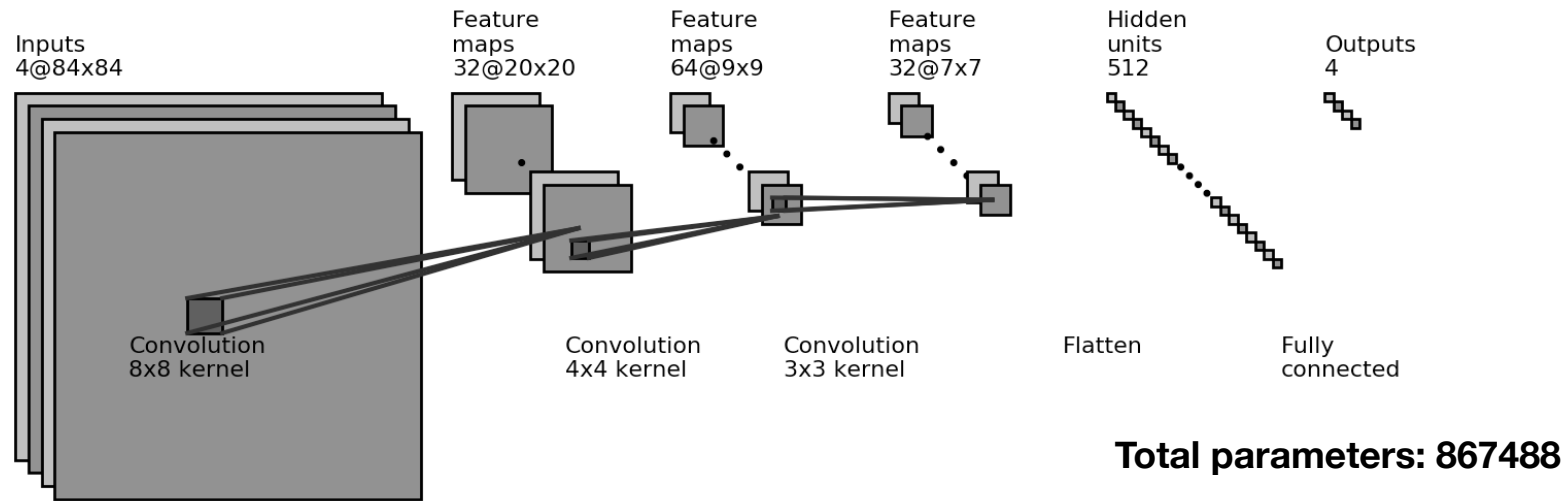


DeepMind

049 2 1



DeepMind



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

```

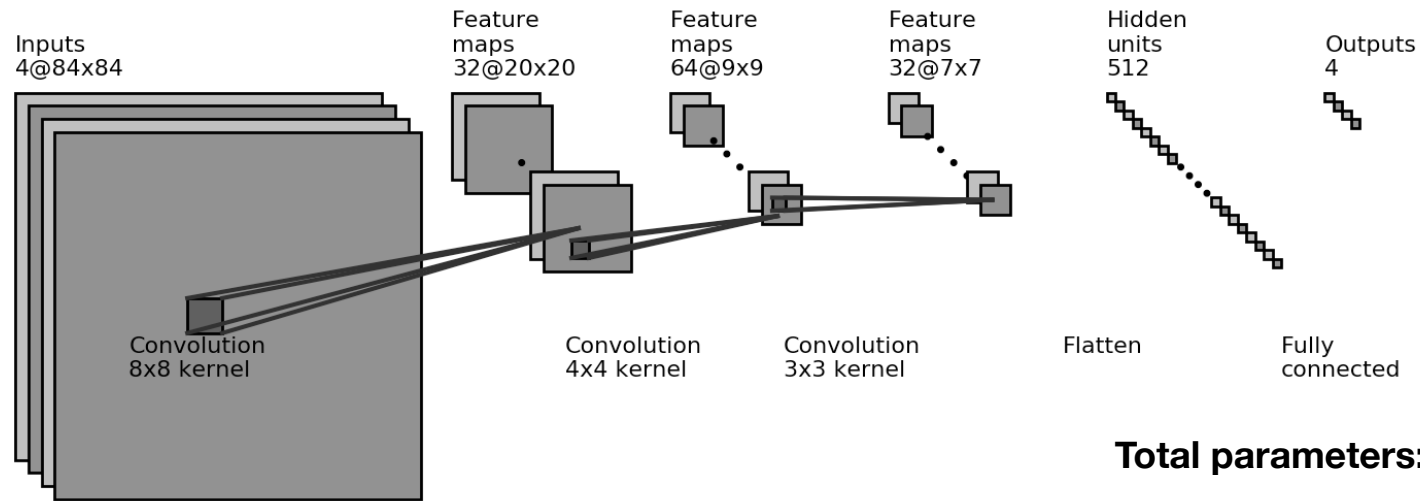
[[[-0.094, -0.008, -0.037, 0.141, 0.088, -0.095, 0.022, -0.101],
  [-0.085, -0.121, -0.552, -0.538, -0.435, 0.103, 0.109, 0.129],
  [ 0.208, -0.103, -0.6, -0.743, -0.409, 0.089, 0.025, 0.044],
  [ 0.048, 0.019, 0.074, -0.066, -0.044, 0.023, 0.006, -0.002],
  [ 0.132, 0.089, -0.008, 0.063, 0.019, -0.023, 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]],

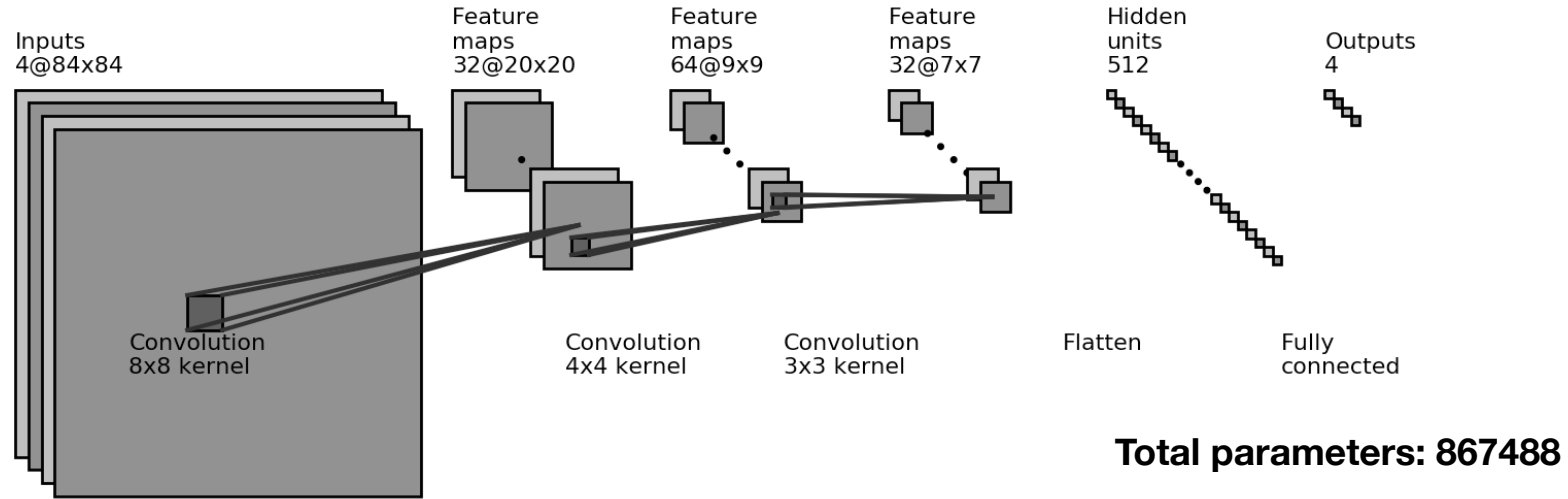
[[ 0.265, 0.268, 0.451, 0.291, 0.352, 0.279, 0.012, 0.053],
 [ 0.172, 0.268, 0.296, 0.223, 0.399, 0.2, 0.022, -0.342],
 [ 0.164, 0.345, 0.273, 0.29, 0.2, -0.094, -0.146, -0.087],
 [ 0.064, 0.094, 0.101, 0.108, -0.004, -0.151, -0.135, -0.169],
 [-0.076, 0.081, -0.087, -0.107, -0.584, -0.284, -0.036, -0.169],
 [ 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**



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]],

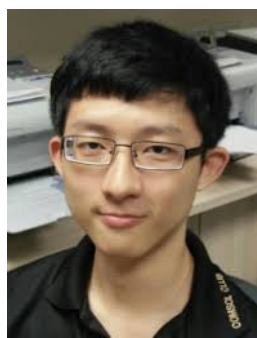
[[ 0.265, 0.268, 0.451, 0.291, 0.352, 0.279, 0.012, 0.053],
 [ 0.172, 0.268, 0.296, 0.223, 0.399, 0.2, 0.022, -0.342],
 [ 0.164, 0.345, 0.273, 0.29, 0.2, -0.094, -0.146, -0.087],
 [ 0.064, 0.094, 0.101, 0.108, -0.004, -0.151, -0.135, -0.169],
 [-0.076, 0.081, -0.087, -0.107, -0.584, -0.284, -0.036, -0.169],
 [ 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

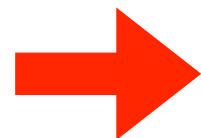
Our focus: Intelligible Intelligence



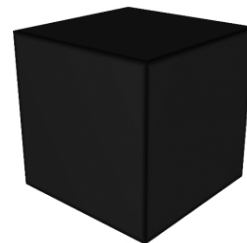
Data

```
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
```

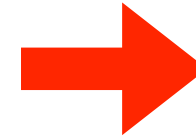
*Neural
network*



Inscrutable
black box model



*Our
focus*



Model we can
understand

$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

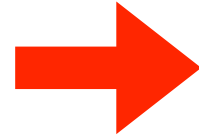
Our focus: Intelligible Intelligence



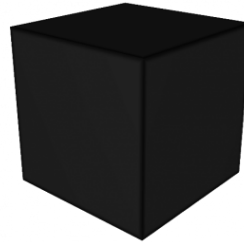
Data

```
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
```

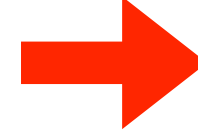
*Neural
network*



Inscrutable
black box model



*Our
focus*



Model we can
understand

$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

- Today I'll discuss auto-discovery of
- ◆ equations (symbolic regression)
 - ◆ conserved quantities
 - ◆ useful degrees of freedom ("pregression")

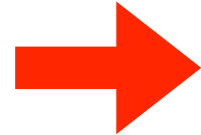
Our focus: Intelligible Intelligence



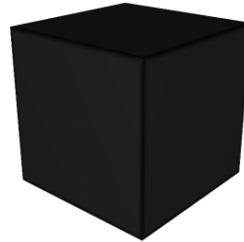
Data

```
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
```

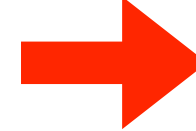
*Neural
network*



Inscrutable
black box model



*Our
focus*



Model we can
understand

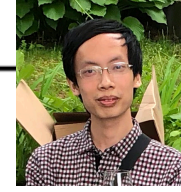
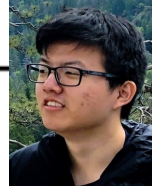
$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

Today I'll discuss auto-discovery of

- ◆ equations (symbolic regression)
- ◆ useful degrees of freedom ("pregression")
- ◆ conserved quantities

Regression

AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity



Silviu-Marian Udrescu, Andrew Tan, Jiahai Feng, Orisvaldo Neto, Tailin Wu & Max Tegmark

MIT Dept. of Physics & Center for Brains, Minds & Machines

Cambridge, MA 02139

sudrescu@mit.edu

(Accepted to NeurIPS 2020)



**Massachusetts
Institute of
Technology**



**Institute for Artificial Intelligence
and Fundamental Interactions**



**CENTER FOR
Brains
Minds+
Machines**

AI Feynman 2.0: Pareto-optimal **symbolic regression** exploiting graph modularity

What's this?

data table

symbolic formula

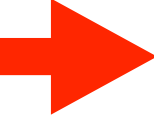
ω_0

ω

θ

$f(\omega_0, \omega, \theta)$

1.1431209959193709	2.7700644791483753	1.7508575540193472	0.23452895063856676
2.1.1431209959193709	2.7700644791483753	1.7508575540193472	0.23452895063856676
2.2.4680655653881054	2.2073166947348444	1.7761705838854234	0.15919403345867914
1.2.7621479700455853	1.4168131204210188	1.5378176974809339	0.14429337334417677
2.1.9536888384746354	2.7336267945043491	1.2592849110534683	0.15360014539410058
1.2.1532278876457527	1.5016008010765851	1.4218686278023172	0.18514940761978987
1.1.9899434091665062	1.4250958039594244	2.5409132056932424	0.17131474581788358
2.1.2841783534277345	2.5038413591290976	1.2255232096430668	0.18928439532548705
2.2.3550853261290494	2.2555822345853405	1.4525468706453792	0.15982929556091857
1.2.7529820467784543	2.6405850369222492	1.6148891450043024	0.13519598787103054
1.1.2043936184306594	1.4441117081403013	1.4546229392136278	0.33122650381723288
1.3423962980280737	2.0552387587225684	2.8071816262301414	0.25403615776576924


$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

AI Feynman 2.0: Pareto-optimal **symbolic regression** exploiting graph modularity

What's this?

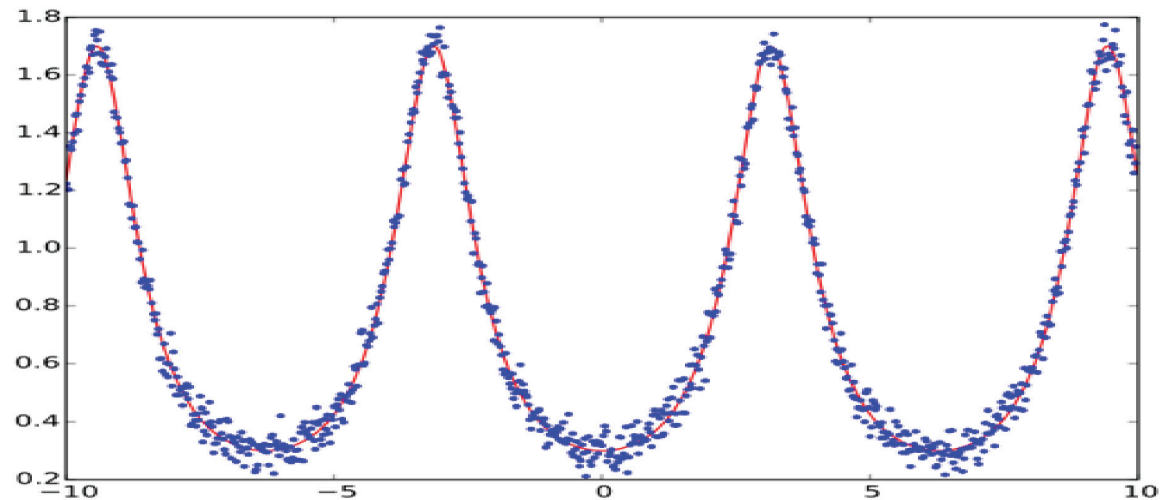
data table

symbolic formula

ω_0	ω	θ	$f(\omega_0, \omega, \theta)$
------------	----------	----------	-------------------------------

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

$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

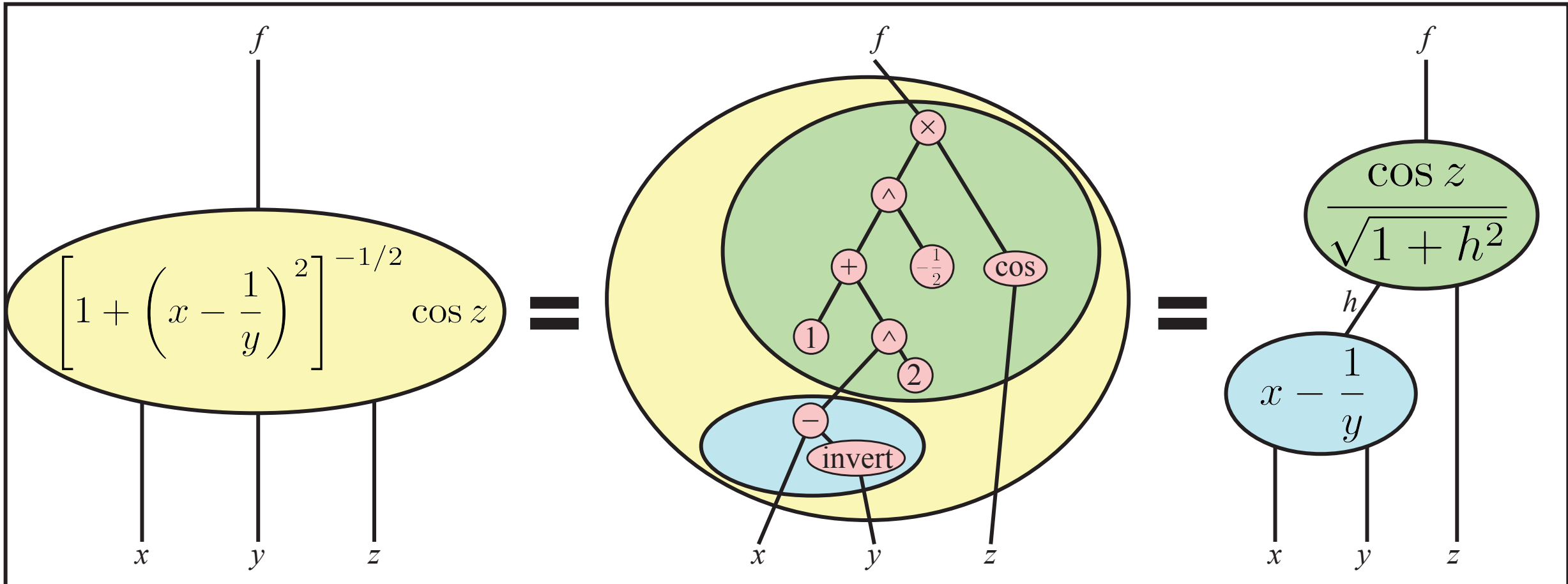


$$r = \frac{a(1 - e^2)}{1 + e \cos(\theta_1 - \theta_2)}$$

AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

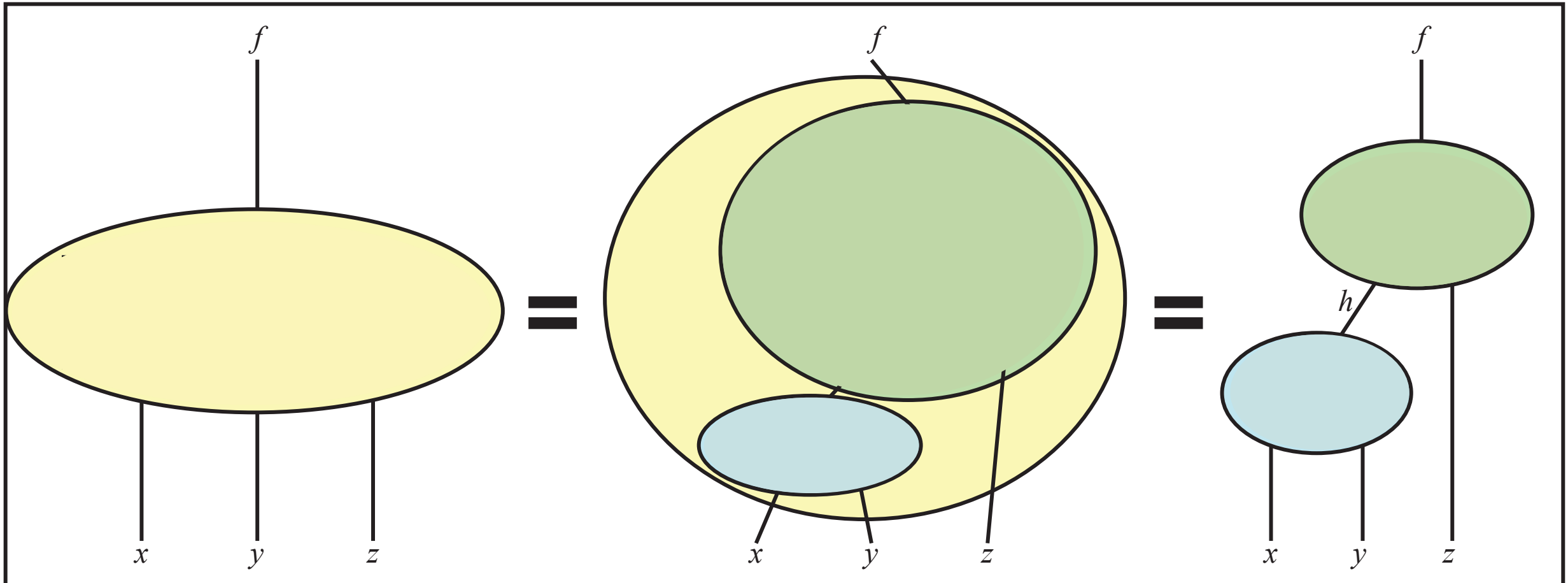
What's this?

Use neural network fit to recursively discover simplifying properties & break problem into simpler ones:



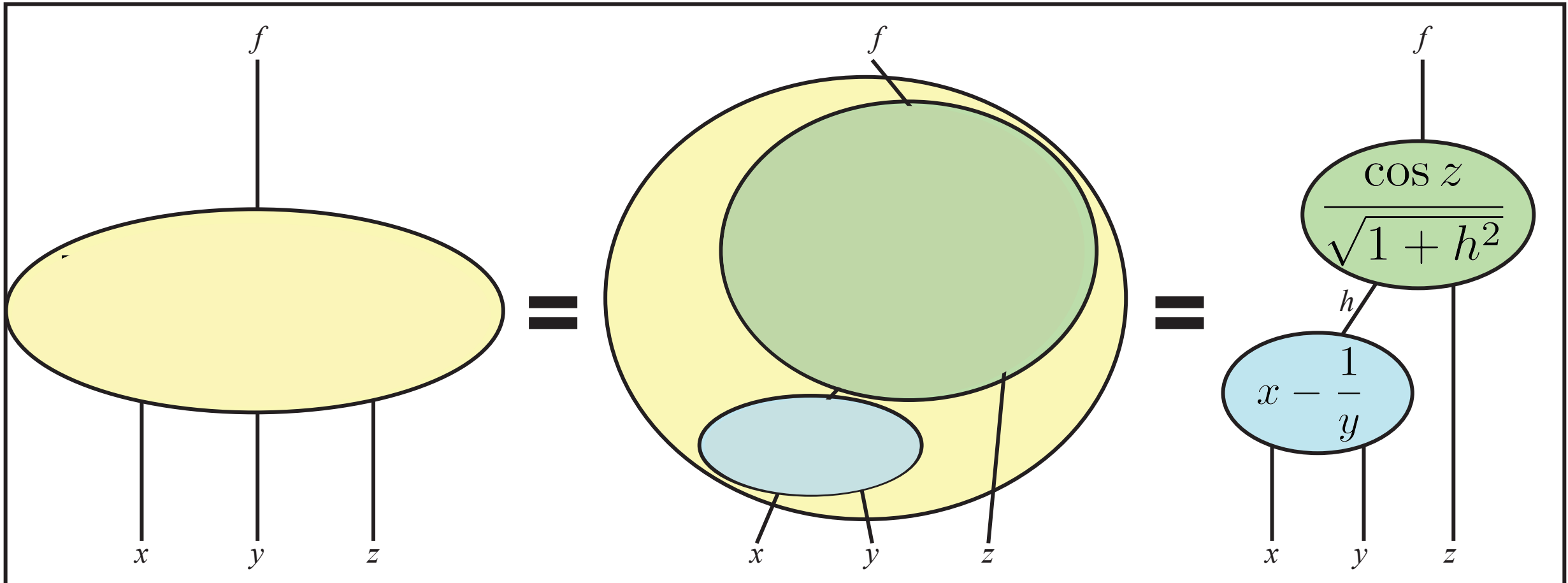
AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

Use neural network fit to recursively discover simplifying properties & break problem into simpler ones:



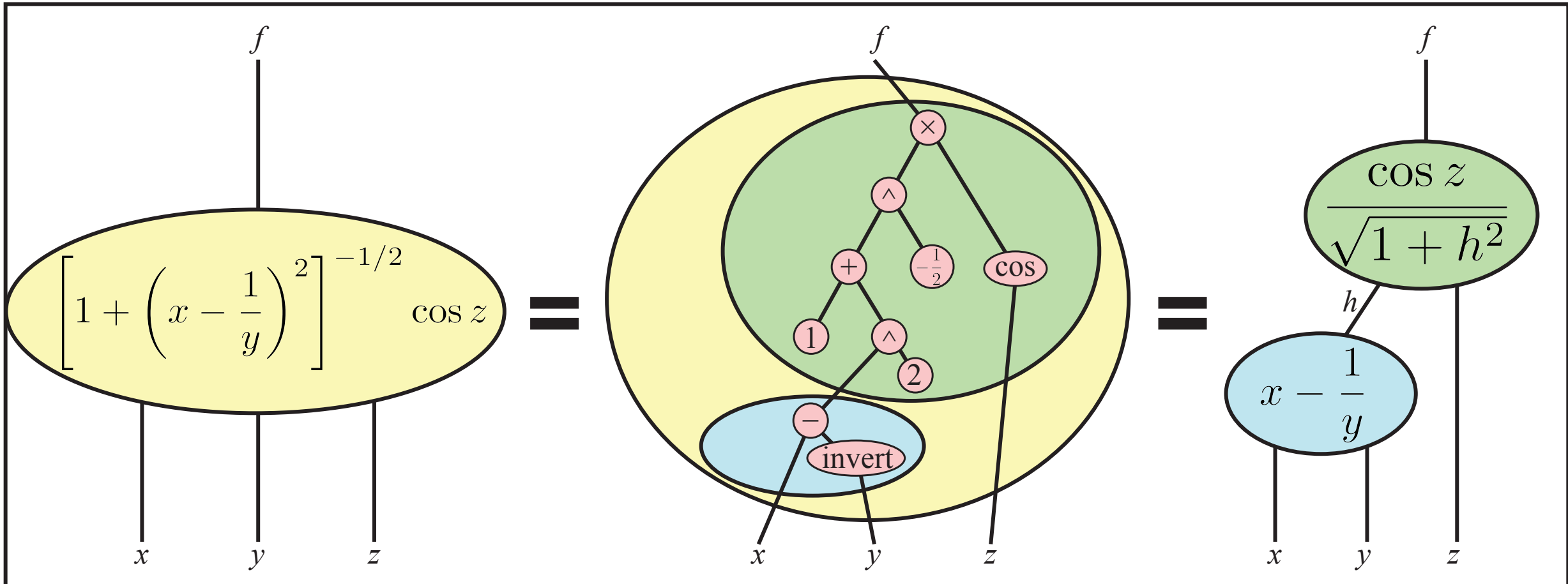
AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

Use neural network fit to recursively discover simplifying properties & break problem into simpler ones:



AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

Use neural network fit to recursively discover simplifying properties & break problem into simpler ones:



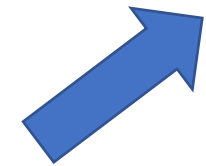
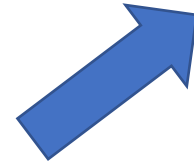
AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

Divide & conquer!



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 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 0.179767914498666503
1.7000728811732311 1.5155671571998763 1.5295642030291683 0.23465391617605863
2.3516545980830331 1.0329627134609363 2.4473327905843174 0.14157574334122566
2.6157973581192184 2.0432144044920815 1.4576627239471807 0.14873897153317067

1.1431209959193709 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

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



Dimensional Analysis

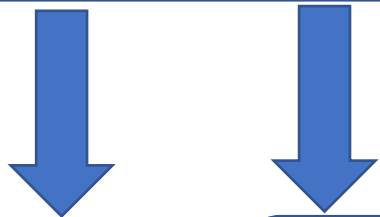
$$\frac{Gm_1^2}{x_1^2}$$

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



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

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



Dimensional Analysis

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

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



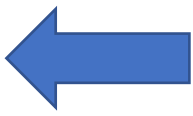
$$\frac{a}{(b - 1)^2 + (c - d)^2 + (e - f)^2}$$

Translational Symmetry
(e-f → h)

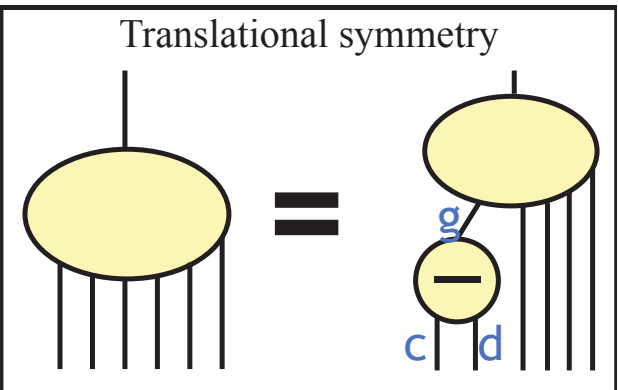
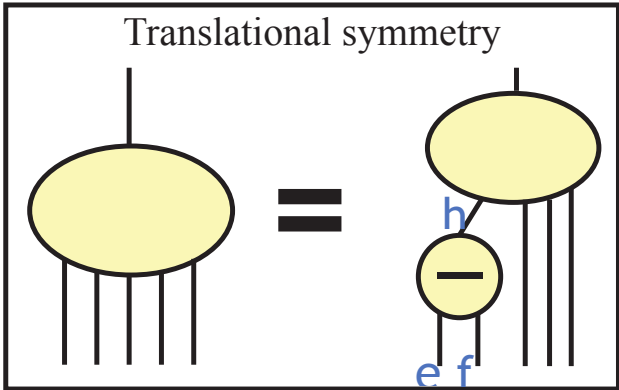
Translational Symmetry
(c-d → g)



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



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



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

Dimensional Analysis

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

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

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

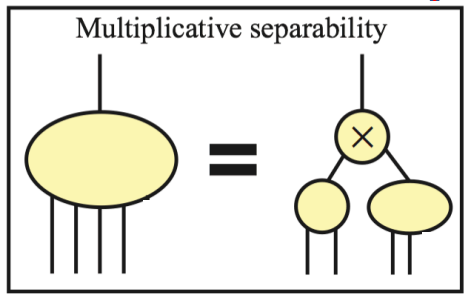
Translational Symmetry
(c-d → g)

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

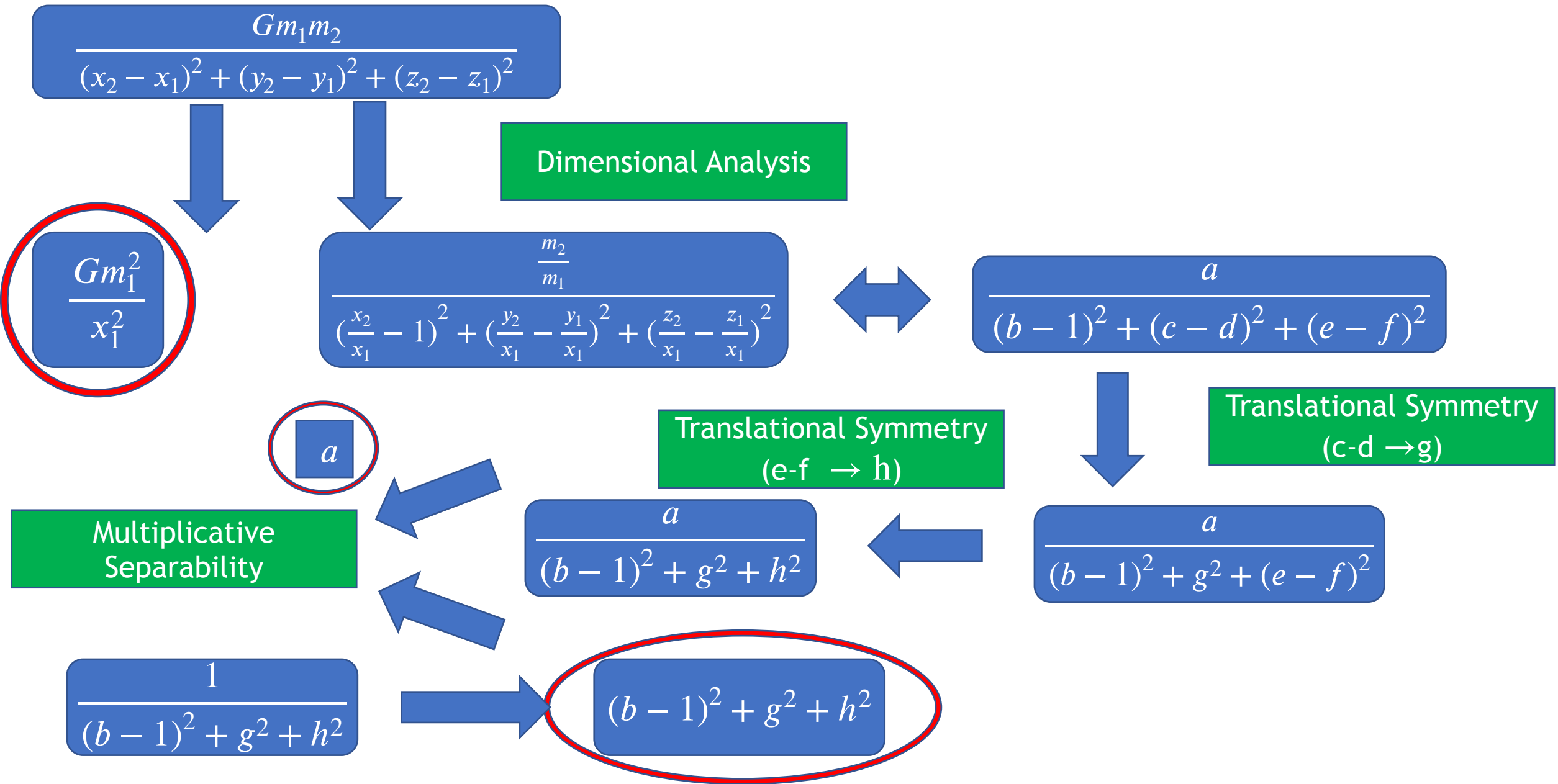
Translational Symmetry
(e-f → h)

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

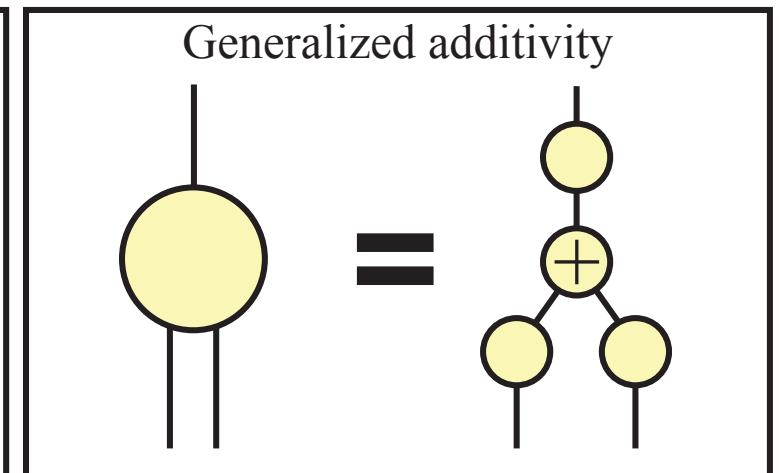
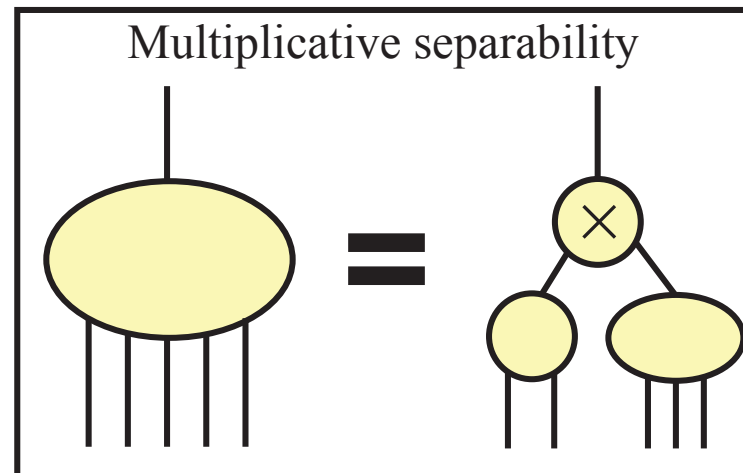
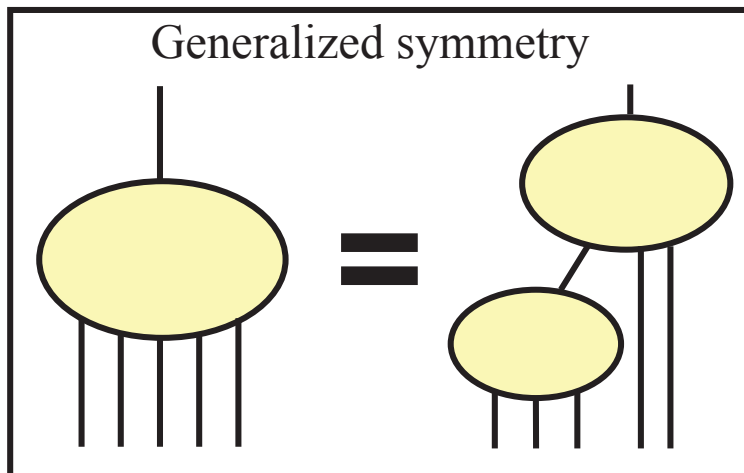
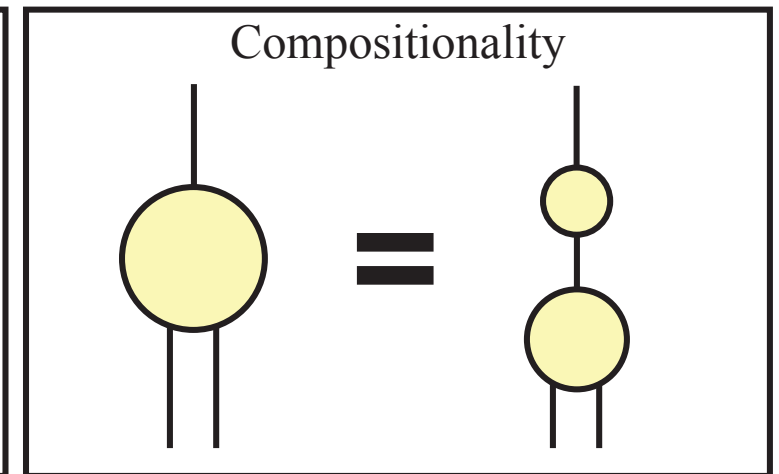
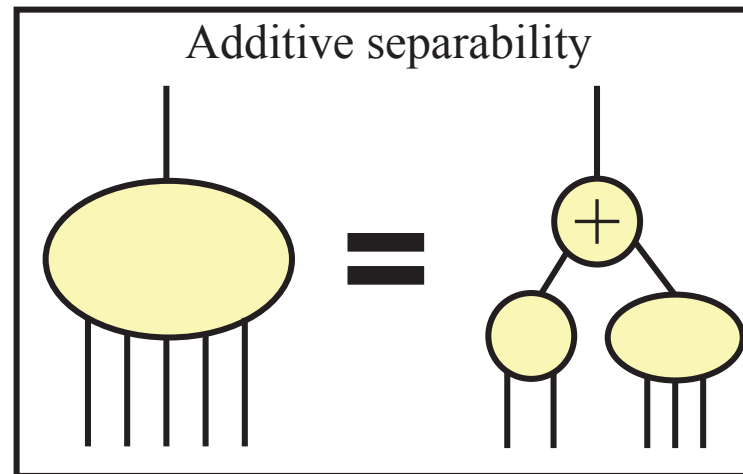
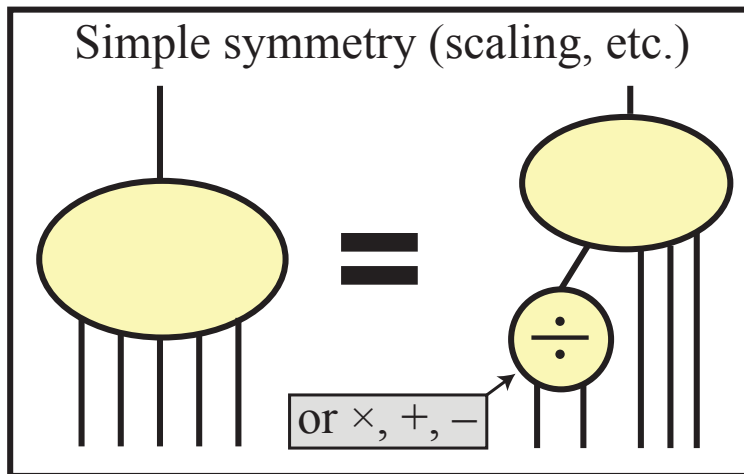
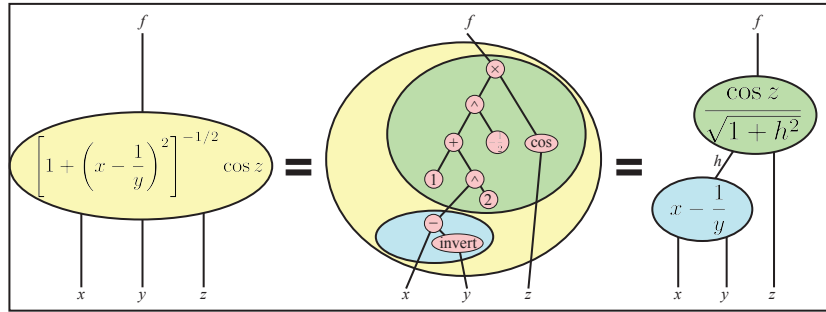
$$a$$



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

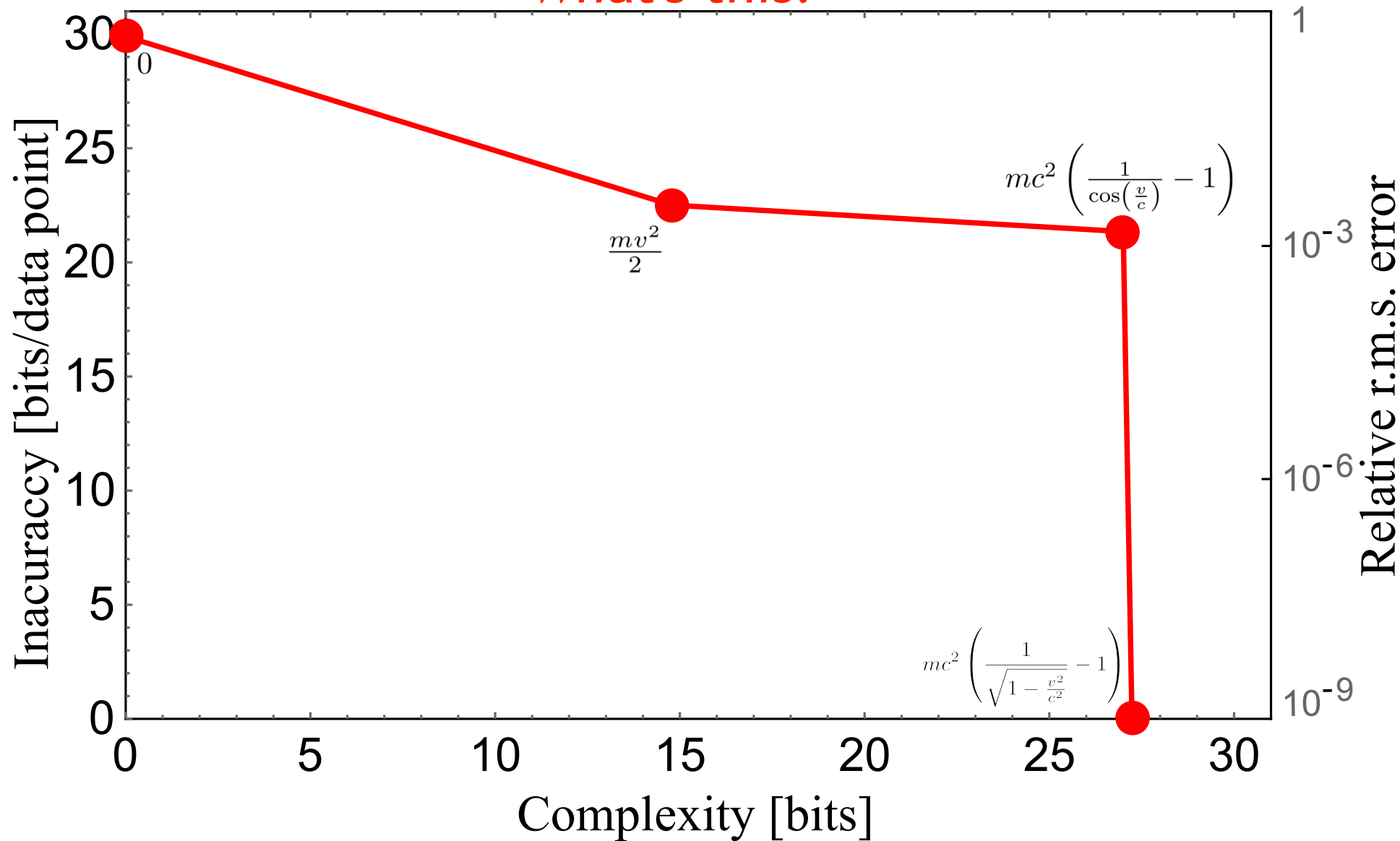


AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

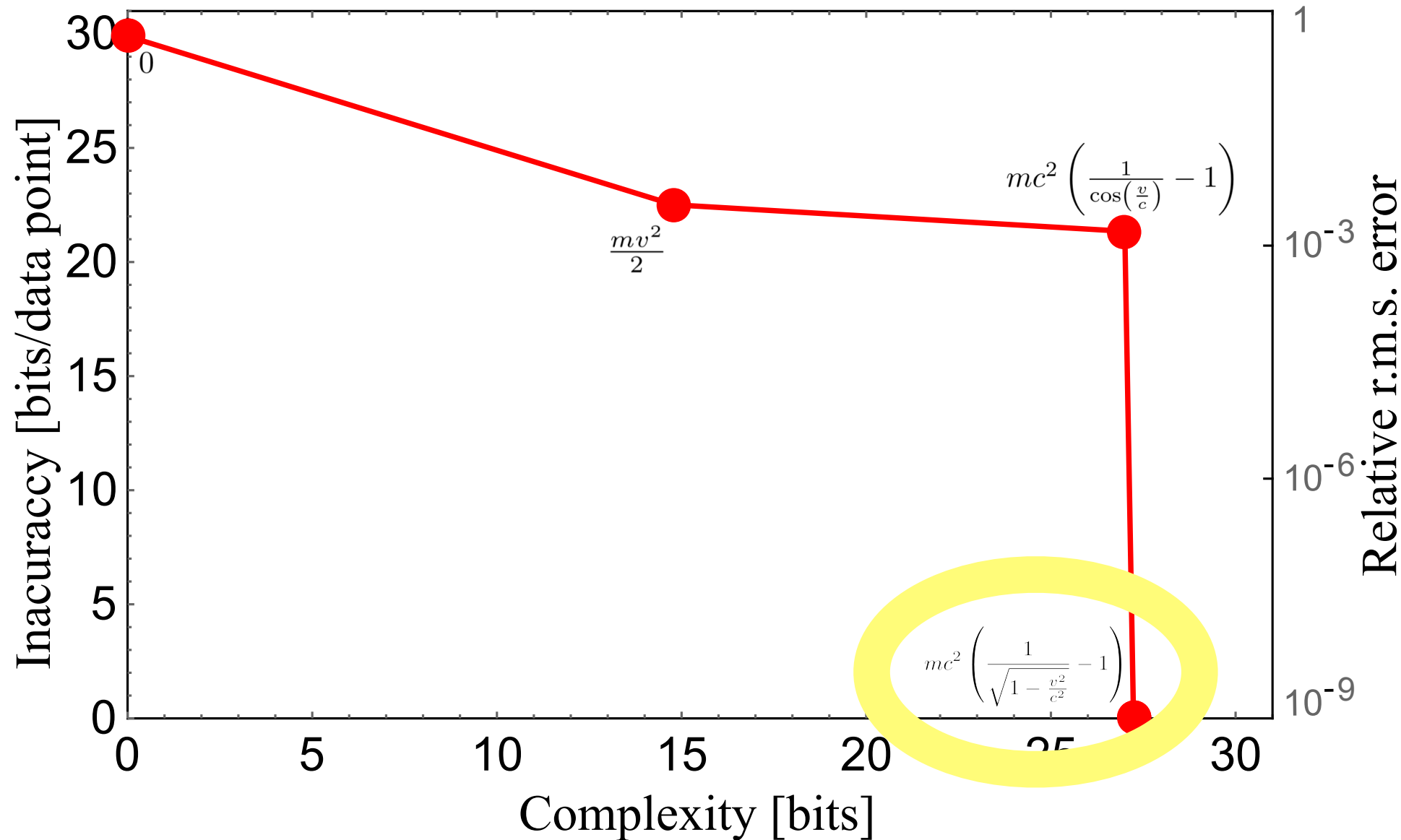


AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity

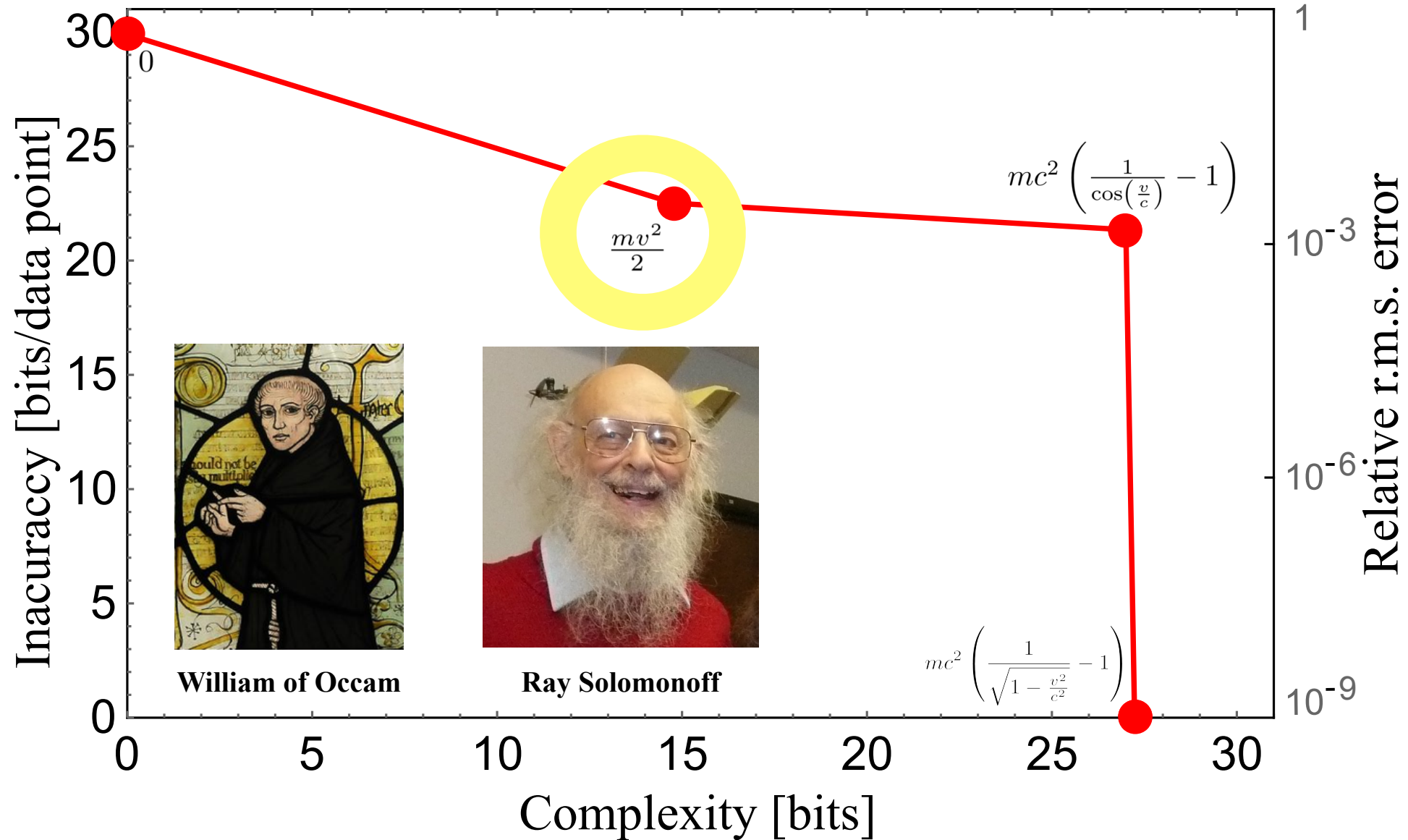
What's this?



AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity



AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity



Complexity definitions:



William of Occam:
too vague to code up

Our compromise:
quantitative & fast to compute

Ray Solomonoff:
rigorous but
too hard to compute



Object	Symbol	Description length L_d
Natural number	n	$\log_2 n$
Integer	m	$\log_2(1 + m)$
Natural number	m/n	$L_d(m) + L_d(n) = \log_2[(1 + m)n]$
Real number	r	$\log_+\left(\frac{r}{\epsilon}\right), \quad \log_+(x) \equiv \frac{1}{2} \log_2(1 + x^2)$
Parameter vector	\mathbf{p}	$\sum_i L_d(p_i)$
Parametrized function	$f(\mathbf{x}; \mathbf{p})$	$L_d(\mathbf{p}) + k \log_2 n; n$ basis functions appear k times

Benchmarks for numerical experiments:

◆ 100 most famous/complicated equations from Feynman Lectures on Physics

Examples:

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

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

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

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

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

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

$$L = mrv\sin\theta$$

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

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

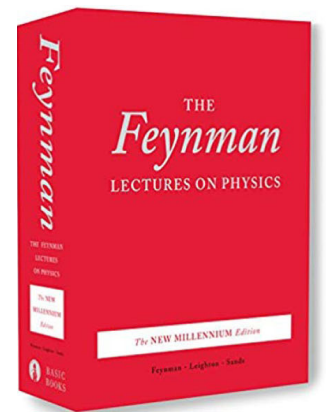
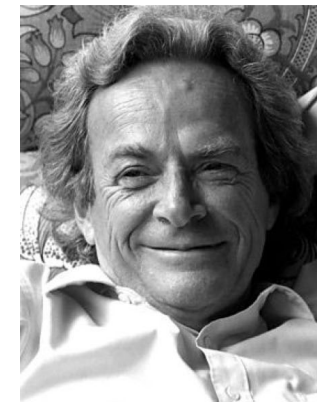
$$E = 2U(1 - \cos kd)$$

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

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

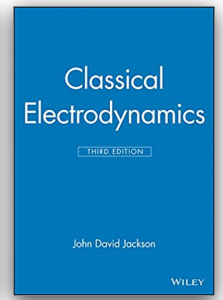
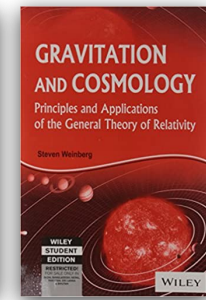
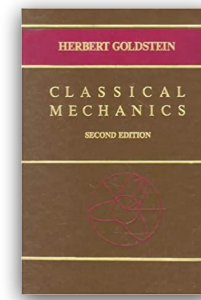
$$L = \frac{\hbar\omega^3}{\pi^2c^2(e^{\hbar\omega/k_bT} - 1)}$$

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



Benchmarks for numerical experiments:

- ◆ 100 most famous/complicated equations from Feynman Lectures on Physics
- ◆ 20 harder equations from graduate physics books



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

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

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

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

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

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

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

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

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

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

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

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

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

Benchmarks for numerical experiments:

- ◆ 100 most famous/complicated equations from Feynman Lectures on Physics
- ◆ 20 harder equations from graduate physics books
- ◆ 12 harder equations with generalized modularity

Examples:

Parallel resistors

$$V = [R_1^{-1} + R_2^{-1} + R_3^{-1} + R_4^{-1}]^{-1} I_0 \cos \omega t$$

Generalized symmetry, separability

RLC circuit

$$I_0 = \frac{V_0}{\sqrt{R^2 + (\omega L - \frac{1}{\omega C})^2}}$$

Generalized symmetry, separability

RLC circuit

$$I = \frac{V_0 \cos \omega t}{\sqrt{R^2 + (\omega L - \frac{1}{\omega C})^2}}$$

Generalized symmetry, separability

Wheatstone bridge

$$V_2 = \left(\frac{R_2}{R_1 + R_2} - \frac{R_x}{R_x + R_3} \right) V_1$$

Generalized symmetry, scaling symmetry, separability

Velocity addition

$$v = c \frac{(v_1 + v_2 + v_3)/c + v_1 v_2 v_3 / c^3}{1 + (v_1 v_2 + v_1 v_3 + v_2 v_3) / c^2}$$

Generalized symmetry

Velocity addition

$$v = c \frac{(v_1 + v_2 + v_3 + v_4)/c + (v_2 v_3 v_4 + v_1 v_3 v_4 + v_1 v_2 v_4 + v_1 v_2 v_3) / c^3}{1 + (v_1 v_2 + v_1 v_3 + v_1 v_4 + v_2 v_3 + v_2 v_4 + v_3 v_4) / c^2 + v_1 v_2 v_3 v_4 / c^4}$$

L_4 norm

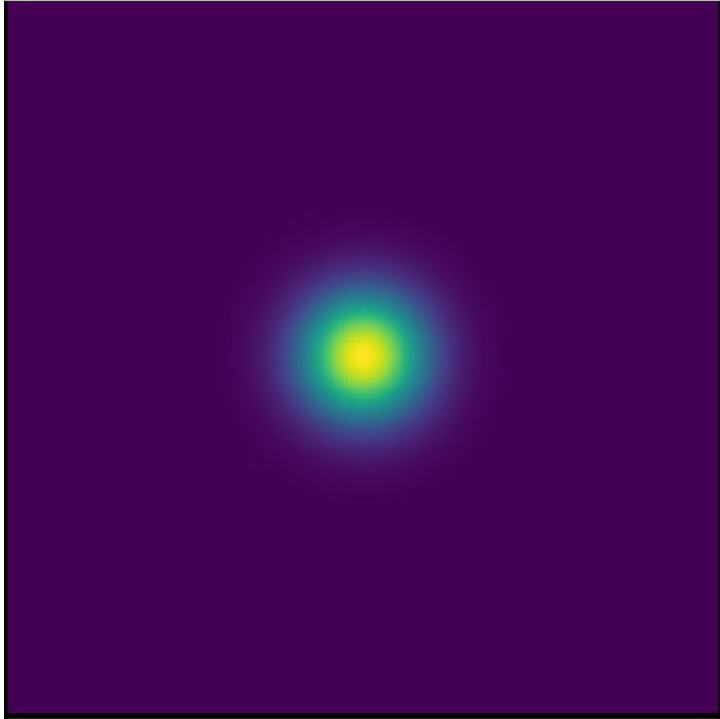
$$z = (x^4 + y^4)^{1/4}$$

Compositionality, Generalized symmetry, generalized additivity

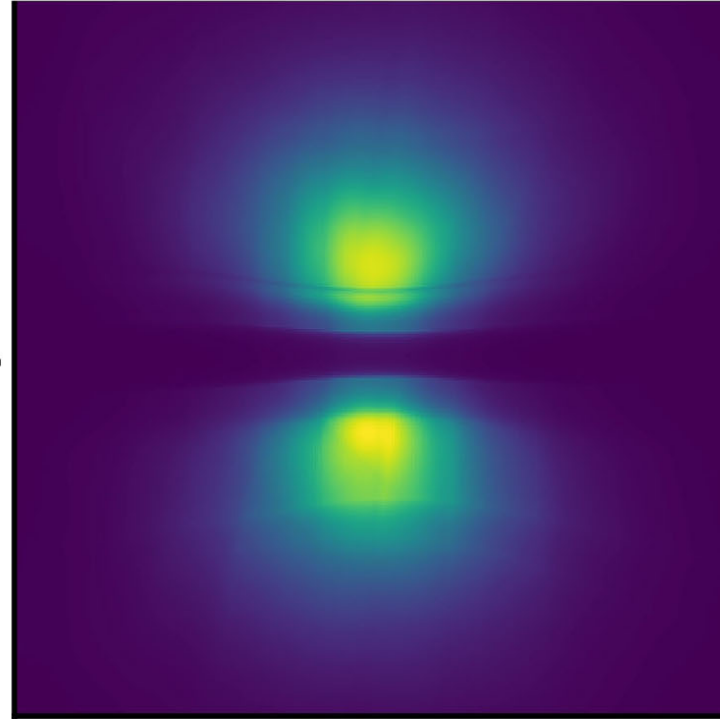
Benchmarks for numerical experiments:

- ◆ 100 most famous/complicated equations from Feynman Lectures on Physics
- ◆ 20 harder equations from graduate physics books
- ◆ 12 harder equations with generalized modularity
- ◆ 10 probability distributions whose samples were converted to density estimates using normalizing flows

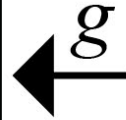
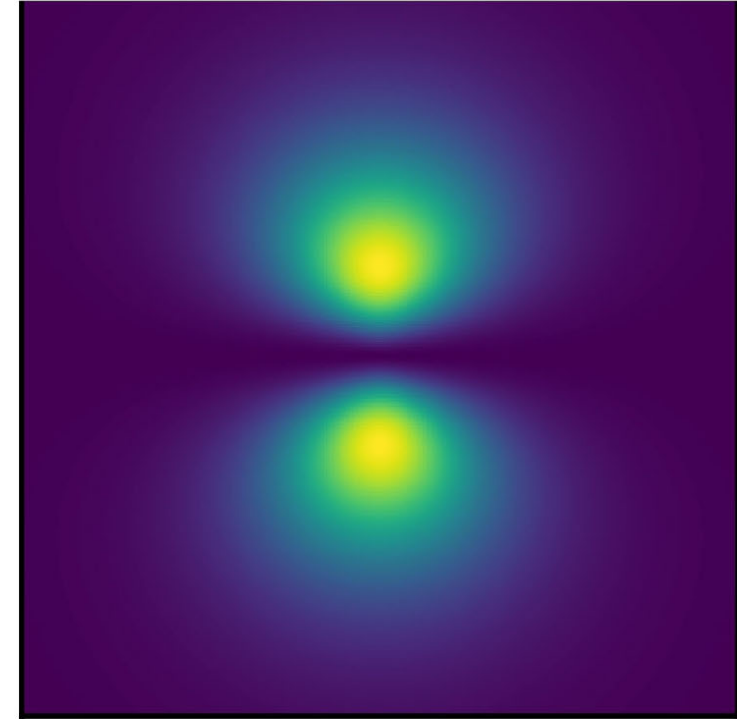
Normal distribution

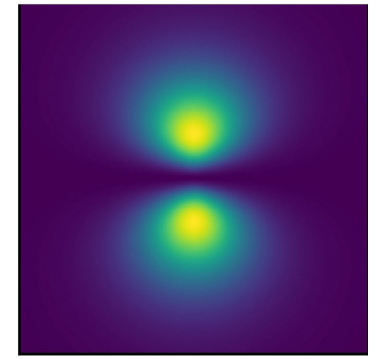
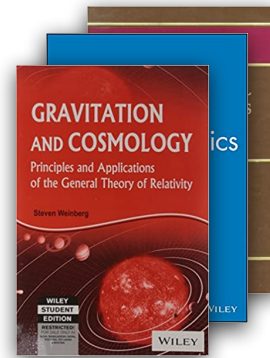
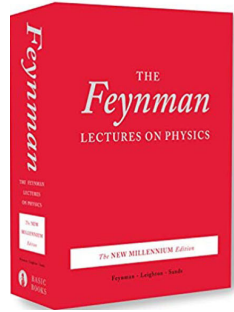


Fit distribution

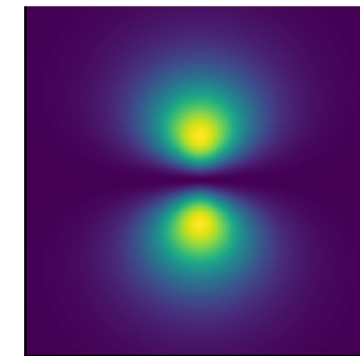
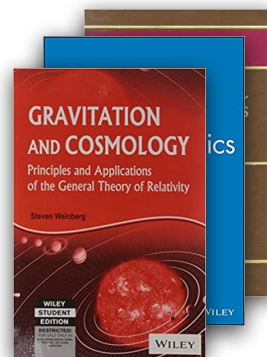
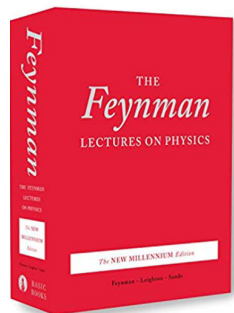


True distribution





	Feynman Database for Symbolic Regression, basic 100	Feynman Database for Symbolic Regression, harder 20	12 modular equations	10 probability distributions
Schmidt & Lipson 2009	71%	15%	42%	60%
Udrescu & Tegmark 2020	100%	85%	42%	70%
This paper	100%	90%	100%	80%



	Feynman Database for Symbolic Regression, basic 100	Feynman Database for Symbolic Regression, harder 20	12 modular equations	10 probability distributions
Schmidt & Lipson 2009	71%	15%	42%	60%
Udrescu & Tegmark 2020	100%	85%	42%	70%
This paper	100%	90%	100%	80%

New robustness		10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	Total
Old robustness	10^{-5}	0	1	2	2	0	5
	10^{-4}	0	1	3	5	12	21
	10^{-3}	0	0	5	6	24	35
	10^{-2}	0	0	0	2	37	39
Total		0	2	10	15	73	100

How *you* can use this:

```
pip install aifeynman
```

Please email us about collaborating on analyzing your data or generalizing this to other domains!

tegmark@mit.edu

<https://ai-feynman.readthedocs.io/en/latest/>

AI Feynman

[Navigation](#)

[Installation](#)

[Usage](#)

[Input format](#)

[Output format](#)

[FAQ](#)

Quick search

AI Feynman

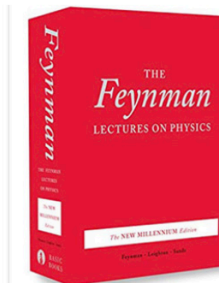


$$L = \frac{\hbar\omega^3}{\pi^2c^2(e^{\hbar\omega/k_bT} - 1)}$$

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

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

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



Quickstart

```
$ pip install aifeynman

$ python
>>> import aifeynman
>>> aifeynman.getdemos('example_data') # download examples from server
>>> aifeynman.launch("../example_data/", "example1.txt", 30,
                    "14ops.txt", polyfit_deg=3, NN_epochs=500)
```

This example will get solved in about 10-30 minutes depending on what computer you have and whether you have a GPU.

Here 'example.txt' contains the data table to perform symbolic regression on, with columns separated by spaces, commas or tabs. The other parameters control the search: here the brute-force modules tries combinations of the 14 basic operations in '14ops.txt' for up to 30 seconds, polynomial fits are tried up to degree 3, and the interpolating neural network is trained for up to 500 epochs.

Note that for now, AI Feynman is supported only for Linux and Mac environments.

More examples

```
python examples/example.py
```

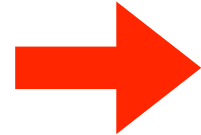
Our focus: Intelligible Intelligence



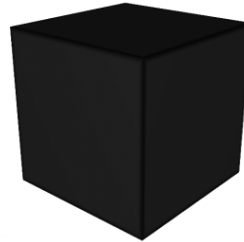
Data

```
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
```

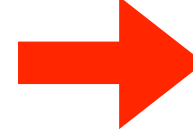
*Neural
network*



Inscrutable
black box model



*Our
focus*



Model we can
understand

$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

- Today I'll discuss auto-discovery of
- ◆ equations (symbolic regression)
 - ◆ **useful degrees of freedom ("pregression")**
 - ◆ conserved quantities

Pregression

arXiv:2005.11212



Silviu Marian-Udrescu

arXiv:2005.11212v2 [cs.CV] 11 Sep 2020

Symbolic Progression: Discovering Physical Laws from Distorted Video

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

(Dated: September 14, 2020)

We present a method for unsupervised learning of equations of motion for objects in raw and optionally distorted unlabeled synthetic video. We first train an autoencoder that maps each video frame into a low-dimensional latent space where the laws of motion are as simple as possible, by minimizing a combination of non-linearity, acceleration and prediction error. Differential equations describing the motion are then discovered using Pareto-optimal symbolic regression. We find that our pre-regression (“pregression”) step is able to rediscover Cartesian coordinates of unlabeled moving objects even when the video is distorted by a generalized lens. Using intuition from multi-dimensional knot-theory, we find that the pregression step is facilitated by first adding extra latent space dimensions to avoid topological problems during training and then removing these extra dimensions via principal component analysis. An inertial frame is auto-discovered by minimizing the combined equation complexity for multiple experiments.

I. INTRODUCTION

A central goal of physics and science more broadly is to discover mathematical patterns in data. For example, after four years of analyzing data tables on planetary orbits, Johannes Kepler started a scientific revolution in 1605 by discovering that Mars’ orbit was an ellipse [1]. There has been great recent progress in automating such tasks with *symbolic regression*: discovery of a symbolic expression that accurately matches a given data set [2–23]. Open-source software now exists that can discover quite complex physics equations by combining neural networks with techniques inspired by physics and information theory [22, 23].

However, there is an important underlying problem that symbolic regression does not solve: how to decide which parameters of the observed data we should try to describe with equations. Eugene Wigner famously stated that “the world is very complicated and ... the complications are called initial conditions, the domains of regularity, laws of nature” [24], so how can the discovery of these regularities be automated? In Figure 1, how can

an unsupervised algorithm learn that to predict the next video frame, it should focus on the x - and y -coordinates of the rocket, not on its color or on the objects in the background? More generally, given an evolving data vector with N degrees of freedom, how can we auto-discover which $n < N$ degrees of freedom are most useful for prediction? Renormalization addresses this question in a particular context, but we are interested in generalizing this. In Kepler’s case, for example, the raw data corresponds to two-dimensional sky images: how can a computer algorithm presented with a series of images with say $N = 10^6$ pixels automatically learn that the most useful degrees of freedom are the $n = 2$ position coordinates of the small reddish dot corresponding to Mars?

The goal of our paper is to tackle this pre-regression problem, which we will refer to this as “pregression” for brevity. Automated pregression enables laws of motion to be discovered starting with raw observational data such as videos. This can be viewed as a step toward unsupervised learning of physics, whereby an algorithm learns from raw observational data without any human supervision or prior knowledge [25–27].

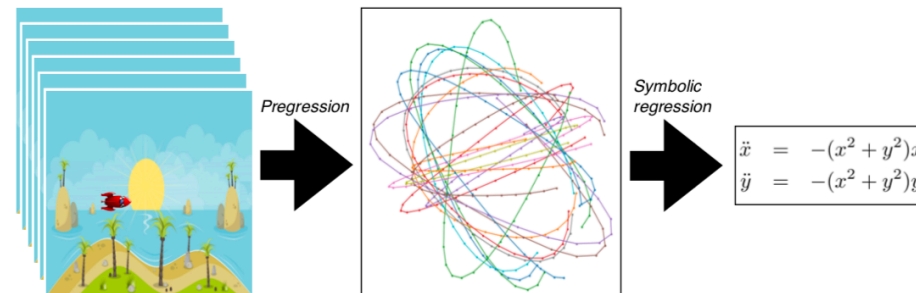
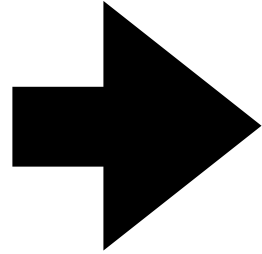


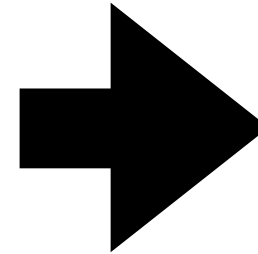
FIG. 1: Our pregression algorithm seeks to autoencode a sequence of video frames (left) corresponding to a specific type of motion into a low-dimensional latent space (middle) where the laws of motion (right) are as simple as possible, in this example those of a quartic oscillator. In the middle figure each point corresponds to the x and y of the rocket in a given frame, while points having the same color and connected by a line belong to the same trajectory.



Pregression



Symbolic regression



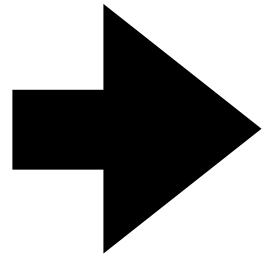
$$\begin{aligned}\ddot{x} &= -(x^2 + y^2)x \\ \ddot{y} &= -(x^2 + y^2)y\end{aligned}$$

Cheating!

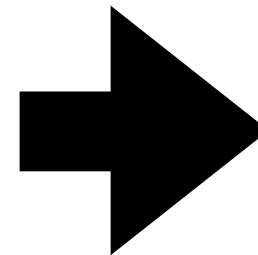
(Also Schmidt & Lipson 2009, *Science*, 324, 81)



Pregression



Symbolic regression



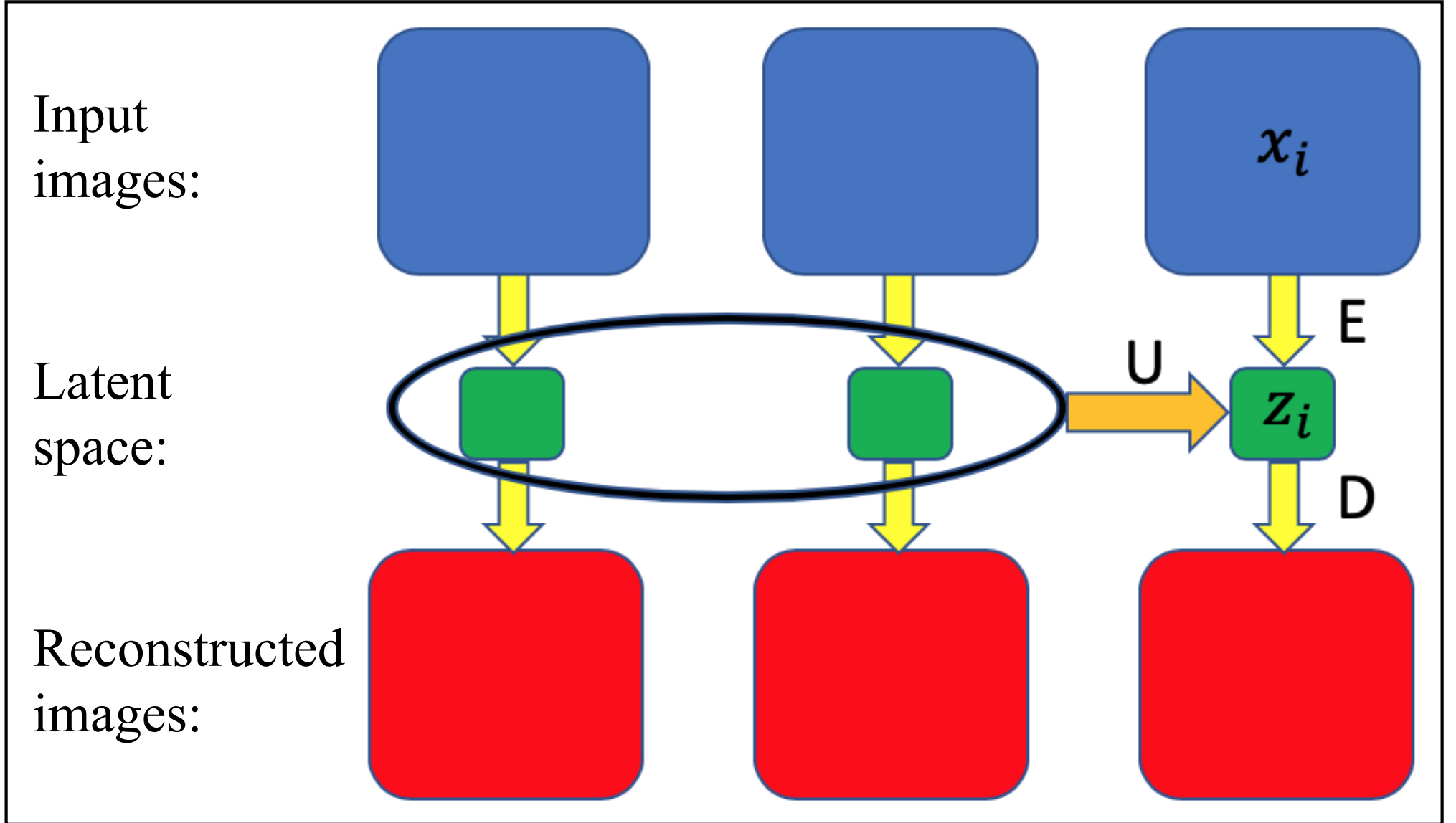
$$\begin{aligned}\ddot{x} &= -(x^2 + y^2)x \\ \ddot{y} &= -(x^2 + y^2)y\end{aligned}$$

Symbolic "pregression":
auto-discovering the relevant variables too



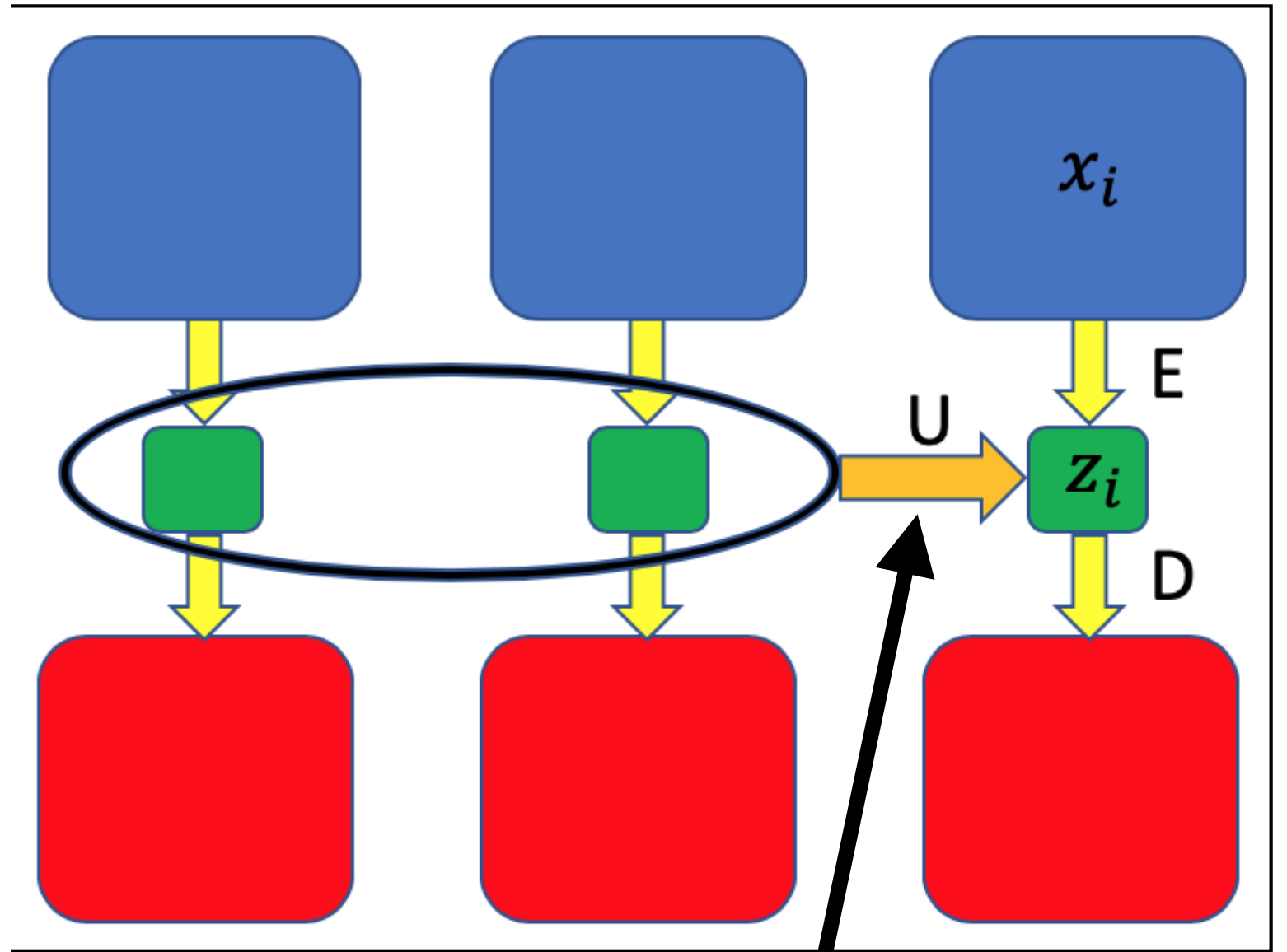


***Harder
than you
might think!***

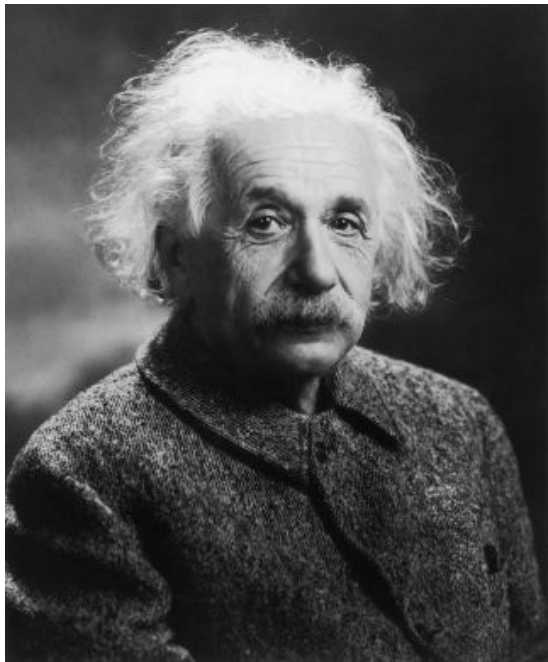




William of Occam



How define simple?

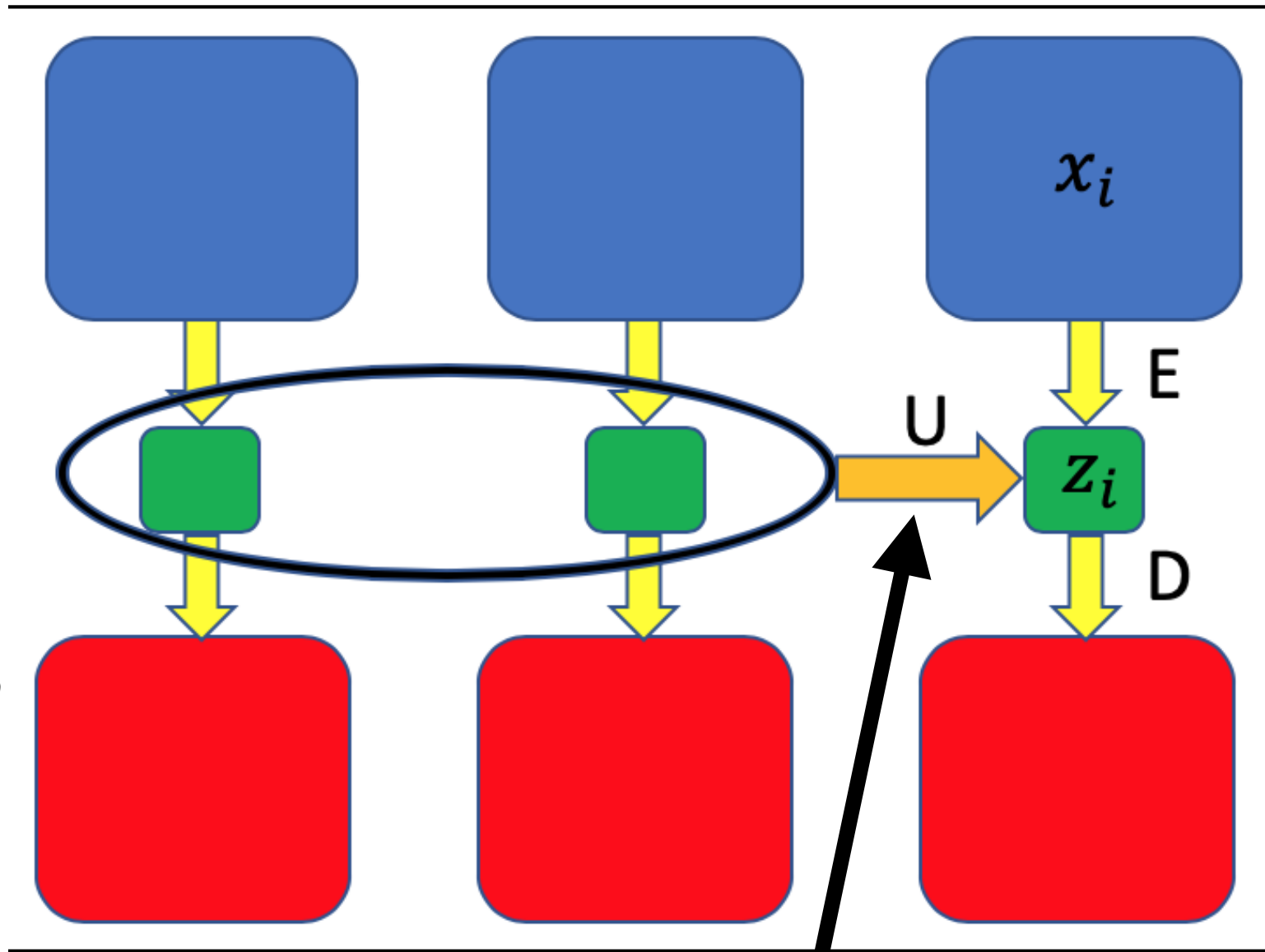


$$\mathcal{L}_{\text{curv}} \equiv R_{\mu\nu\beta}^{\alpha} R_{\alpha}^{\mu\nu\beta}$$

$$R_{\mu\nu\beta}^{\alpha} \equiv \Gamma_{\nu\beta,\mu}^{\alpha} - \Gamma_{\mu\beta,\nu}^{\alpha} + \Gamma_{\mu\beta}^{\gamma} \Gamma_{\nu\gamma}^{\alpha} - \Gamma_{\nu\beta}^{\gamma} \Gamma_{\mu\gamma}^{\alpha},$$

$$\Gamma_{\mu\nu}^{\alpha} \equiv \frac{1}{2} g^{\alpha\sigma} (g_{\sigma\mu,\nu} + g_{\sigma\nu,\mu} - g_{\mu\nu,\sigma}),$$

$$\mathbf{g} \equiv \mathbf{J}\mathbf{J}^t,$$



How define simple?

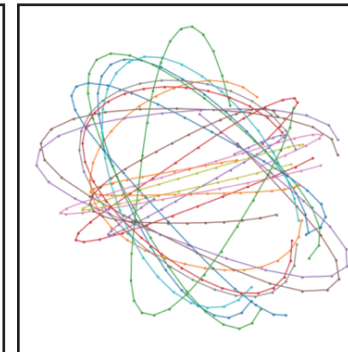
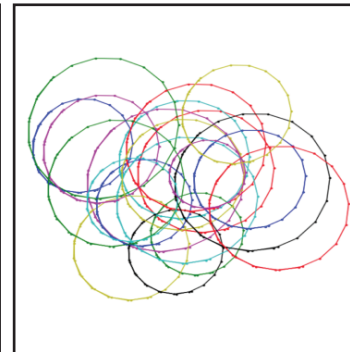
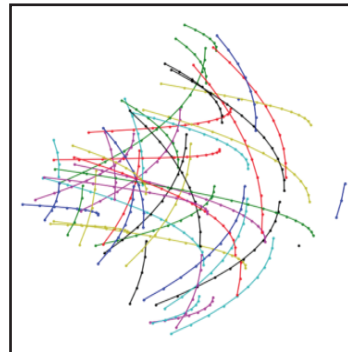
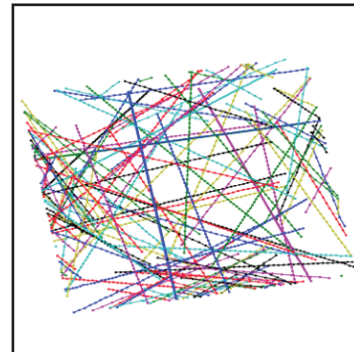
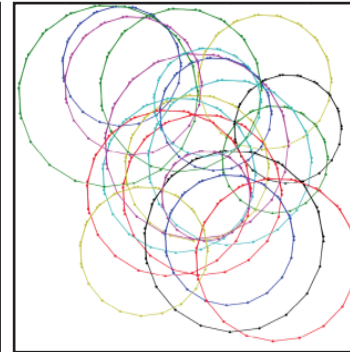
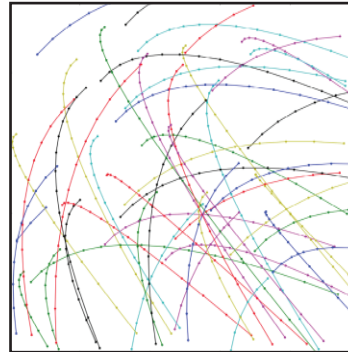
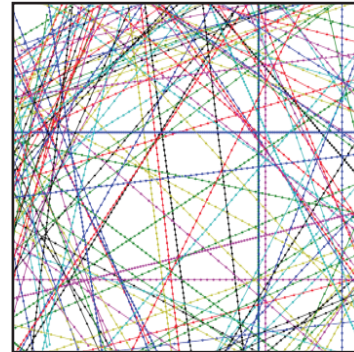
No force

Gravity

Magnetic field

2D harmonic oscillator

Quartic oscillator

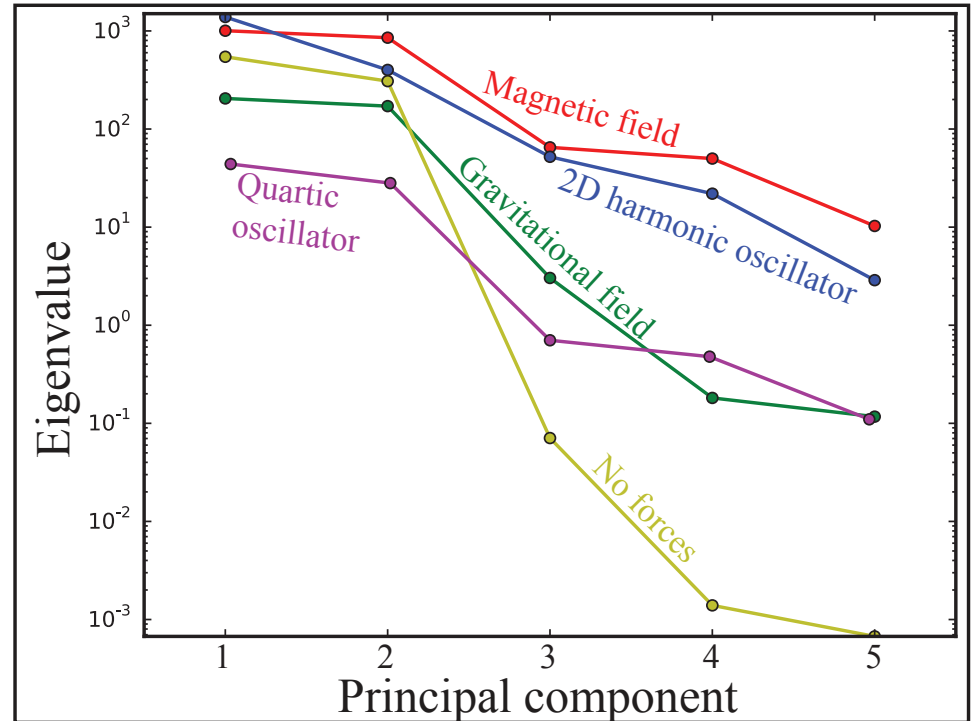
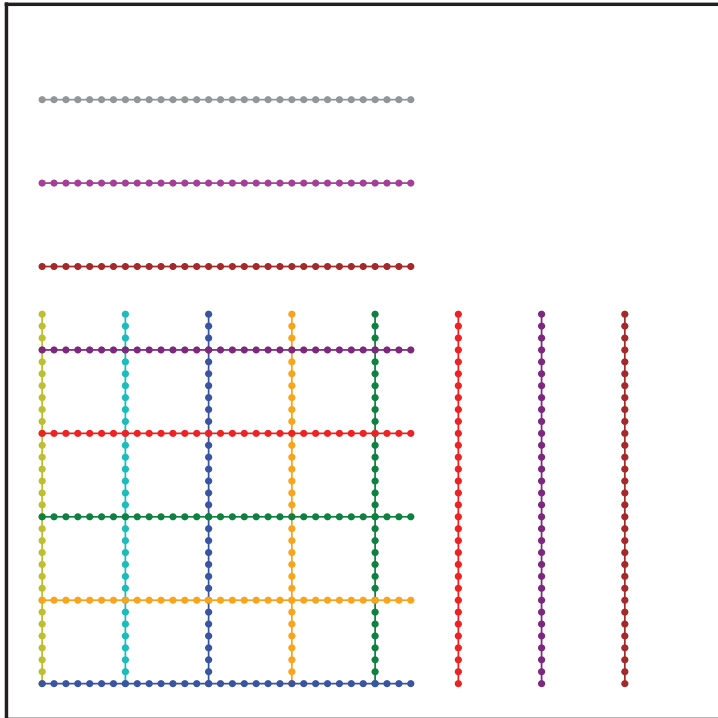


Equation Name	Correct Equation
Uniform	$\ddot{x} = 0$ $\ddot{y} = 0$
Acceleration	$\ddot{x} = 5$ $\ddot{y} = -5$
Magnetic	$\ddot{x} = -\frac{1}{3}\dot{y}$ $\ddot{y} = \frac{1}{3}\dot{x}$
Harmonic oscillator	$\ddot{x} = -\frac{4}{9}x$ $\ddot{y} = -\frac{4}{9}y$
Quartic oscillator	$\ddot{x} = -\frac{4 \times 10^{-6}}{9}x(x^2 + y^2)$ $\ddot{y} = -\frac{4 \times 10^{-6}}{9}y(x^2 + y^2)$



No acceleration in physical space

Latent space – oops!

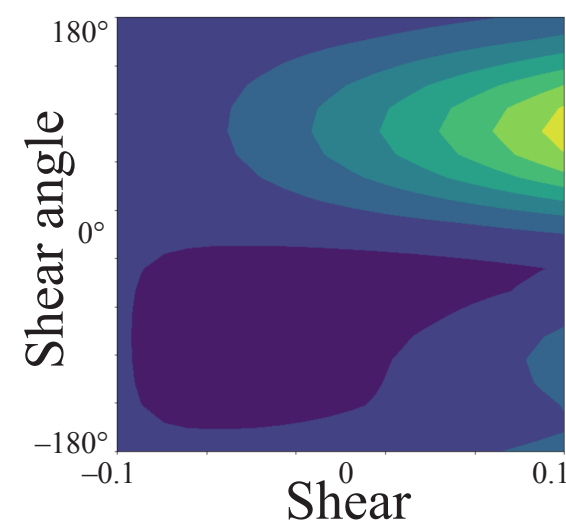
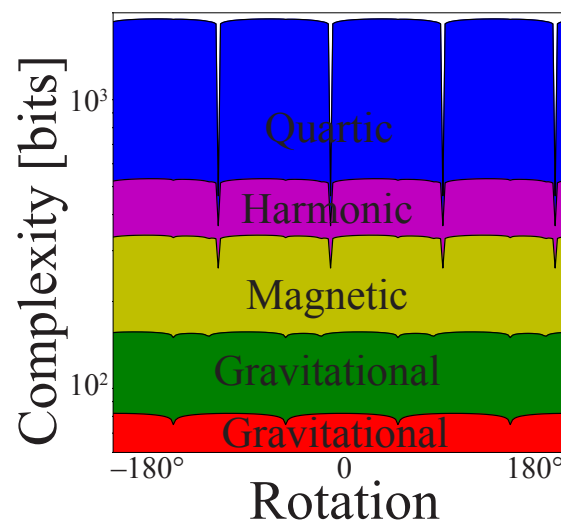
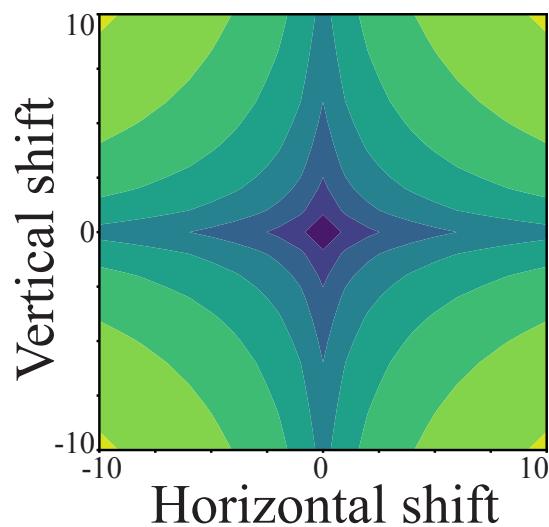
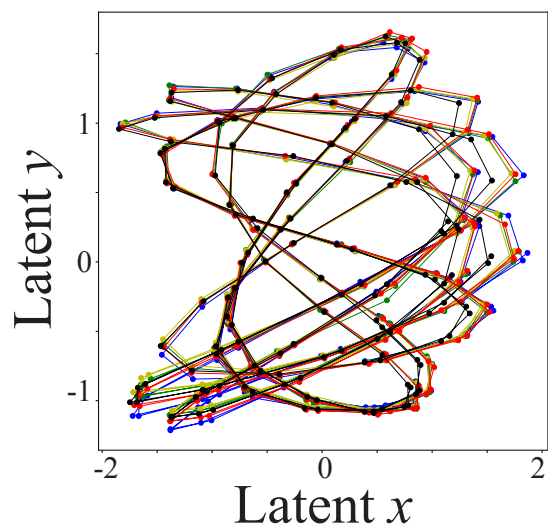


Which affine transformation

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$

of this space makes the equations of motion maximally simple?

Equation Name	Correct Equation
Uniform	$\ddot{x} = 0$ $\ddot{y} = 0$
Acceleration	$\ddot{x} = 5$ $\ddot{y} = -5$
Magnetic	$\ddot{x} = -\frac{1}{3}\dot{y}$ $\ddot{y} = \frac{1}{3}\dot{x}$
Harmonic oscillator	$\ddot{x} = -\frac{4}{9}x$ $\ddot{y} = -\frac{1}{9}y$
Quartic oscillator	$\ddot{x} = -\frac{4 \times 10^{-6}}{9}x(x^2 + y^2)$ $\ddot{y} = -\frac{4 \times 10^{-6}}{9}y(x^2 + y^2)$



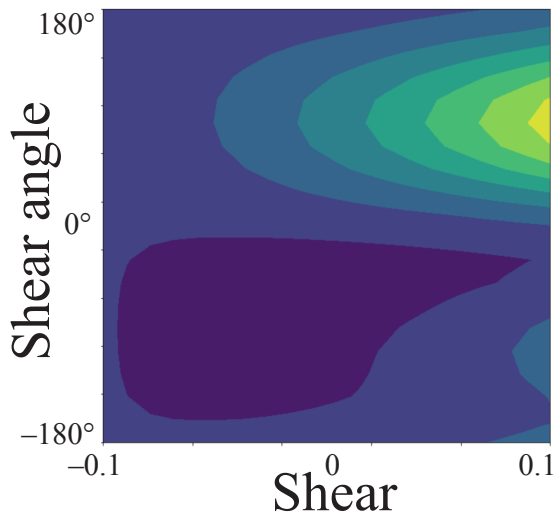
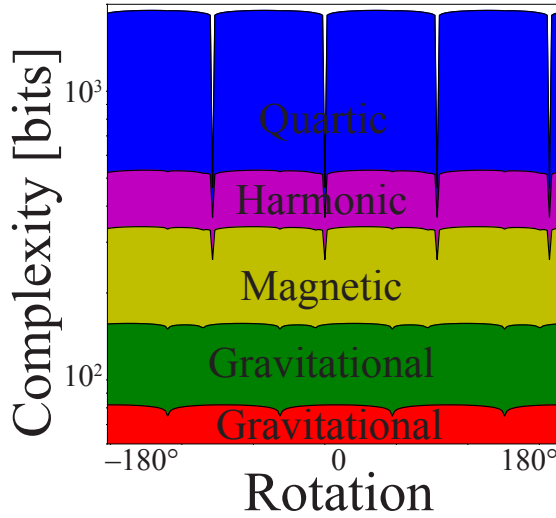
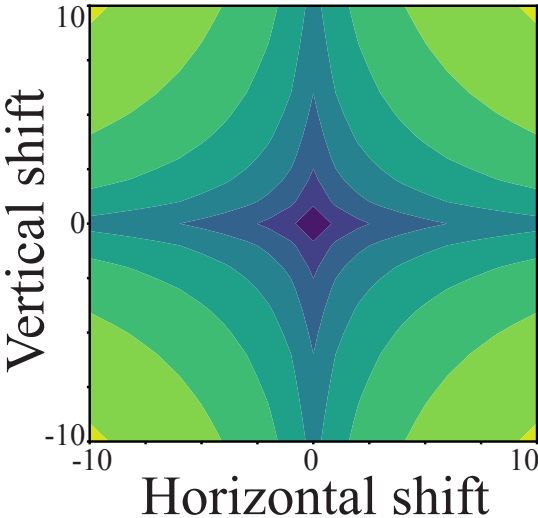
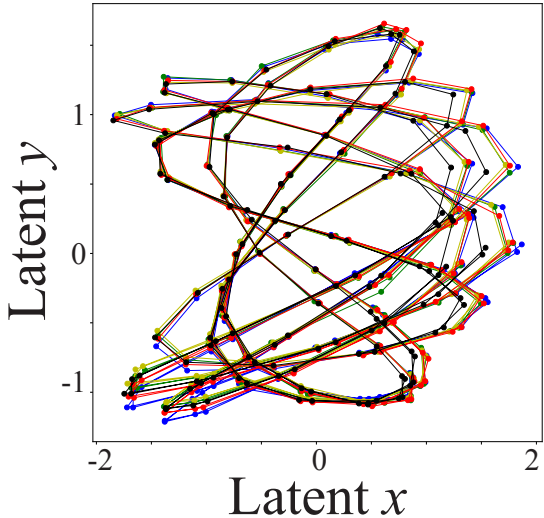
Can ML discover new physics not put in by hand?

- **Can ML discover new physics equations not put in by hand?**

Yes!

- **Can ML discover inertial frames not put in by hand?**

Yes!



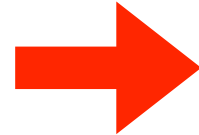
Our focus: Intelligible Intelligence



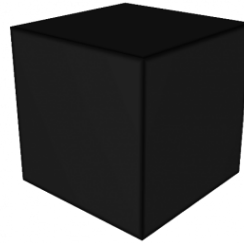
Data

```
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
```

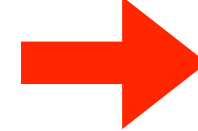
*Neural
network*



Inscrutable
black box model



*Our
focus*



Model we can
understand

$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

- Today I'll discuss auto-discovery of
- ◆ equations (symbolic regression)
 - ◆ useful degrees of freedom ("pregression")
 - ◆ conserved quantities

AI Poincare

We present *AI Poincaré*, a machine learning algorithm for auto-discovering conserved quantities using trajectory data from unknown dynamical systems. We test it on five Hamiltonian systems, including the gravitational 3-body problem, and find that it discovers not only all conserved quantities, but also periodic orbits, regular/chaotic phase transitions and conservation breakdown timescales.



Ziming Liu



INTRODUCTION

While machine learning has contributed to many physics advances, such as improving the speed or quality of numerical simulations, laboratory experiments and astronomical observations [1–7], a more ambitious goal is to design intelligent machines to make new scientific discoveries such as physical symmetries [8–11] and formulas via symbolic regression [12–16]. In this spirit, the goal the present paper is to auto-discover conservation laws from trajectories of dynamical systems.

Physicists have traditionally derived conservation laws in a *model-driven* way, such as when Poincaré proved [17] that the 3D gravitational 3-body problem has only 8 conserved quantities. In contrast, this paper aims to discover conservation laws in a *data-driven* way, using only observed trajectory data as input while treating the underlying dynamical equations as unknown.

To the best of our knowledge, [11] has pursued the goal closest to ours, but with an orthogonal approach detecting symmetry with an auto-encoder, requiring hand-crafted features precluding full automation, and testing on relatively simple examples. Other work linking conservation law and machine learning [9, 18–20] focus on embedding physical inductive biases into machine learning, but not the other way around to automate physical discoveries with machine learning.

Our ambitious goal of automating conservation law discovery is enabled by recent machine-learning techniques for analyzing on *manifolds*, which are intimately related to dynamical systems as summarized in Table I: viewing each state as a point in a phase space \mathbb{R}^n , the topological closure of the set of all states on a trajectory form a manifold $\mathcal{M} \subset \mathbb{R}^n$. Each conservation law removes one degree of freedom from the dynamical system and one dimension from \mathcal{M} , so the number of conserved quantities is simply n minus the dimensionality of \mathcal{M} . The local tangent space of \mathcal{M} represents all local displacements allowed by conservation laws, while the space perpendicular to the tangent space is spanned by gradients of conserved quantities.

We introduce our notations and AI Poincaré algorithm in the Methods section. In the Results section, we test AI Poincaré on five Hamiltonian systems and its ability to discover conserved quantities (numerically and symboli-

TABLE I: Manifold/dynamical system correspondence

Manifold	Dynamical System
Dimension Reduction	Conservation Law
Tangent Space	(Local) Conserved Quantity Isosurface
Orthogonal Space	Gradients of Conserved Quantities

cally), periodic orbits, regular/chaotic phase transitions, and conservation breakdown timescales.

METHOD

Manifold learning is an active field in machine learning and has developed powerful tools [21–26] to explore and visualize the latent structure of a low-dimensional manifold embedded in high-dimensional space. However, none of these methods is immediately applicable for our use because they do not aim to exactly determine the manifold dimensionality, instead focusing on performance on downstream tasks (*e.g.*, image/video generation) or on visualization where that the dimensionality is a user-specified input parameter. Moreover, the manifold dimensionality is poorly defined due to discreteness and noise; we circumvent this by treating dimensionality as a renormalized quantity that is a function of length scale σ_L .

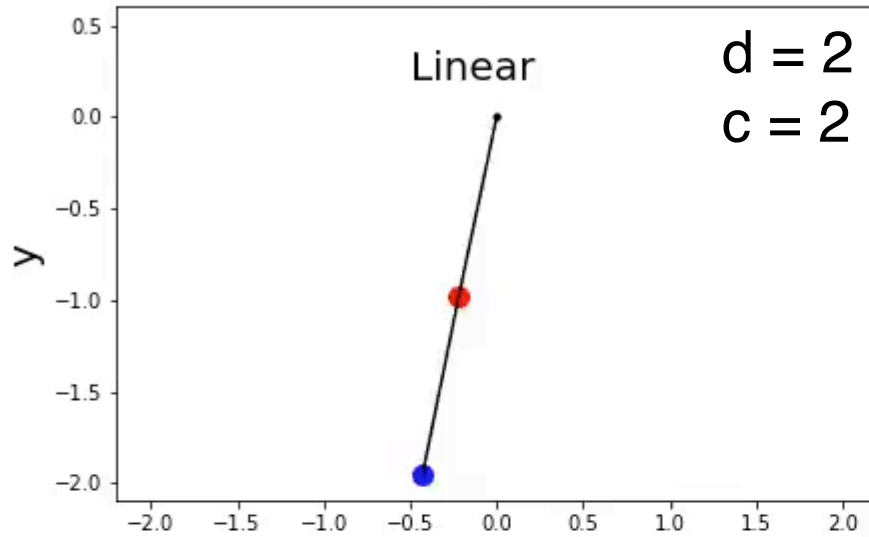
Problem and notation: Hamiltonian systems play an important role in classical physics [28]. A k -body Hamiltonian system in d -dimensional space can be described by a position vector $\mathbf{q} = (\mathbf{q}^{(1)}, \dots, \mathbf{q}^{(k)}) \in \mathbb{R}^{kd}$ and a momentum vector $\mathbf{p} = (\mathbf{p}^{(1)}, \dots, \mathbf{p}^{(k)}) \in \mathbb{R}^{kd}$ which is conjugate to \mathbf{q} . We define the state of a Hamiltonian system as $\mathbf{x} = (\mathbf{q}, \mathbf{p}) \in \mathbb{R}^{2kd}$. The system has a Hamiltonian function $H_0(\mathbf{q}, \mathbf{p})$ and the canonical equations describing the dynamics are

$$\frac{d\mathbf{p}^{(i)}}{dt} = -\frac{\partial H_0}{\partial \mathbf{q}^{(i)}}, \quad \frac{d\mathbf{q}^{(i)}}{dt} = \frac{\partial H_0}{\partial \mathbf{p}^{(i)}} \quad (i = 1, \dots, k). \quad (1)$$

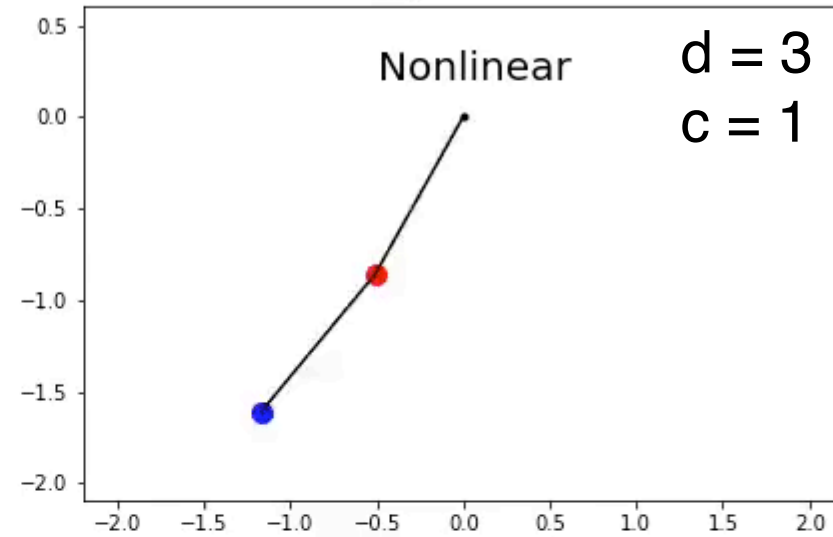
Conservation laws are important characteristics of Hamiltonian systems, common examples including conservation of energy, momentum, angular momentum, Runge-Lenz vector, magnetic moment, *etc.*. We express a set of independent conservation laws as $H_j(\mathbf{q}, \mathbf{p}) = h_j$ ($j = 0, \dots, m - 1$), valid exactly or approximately for all

Double pendulum ($n = 4$)

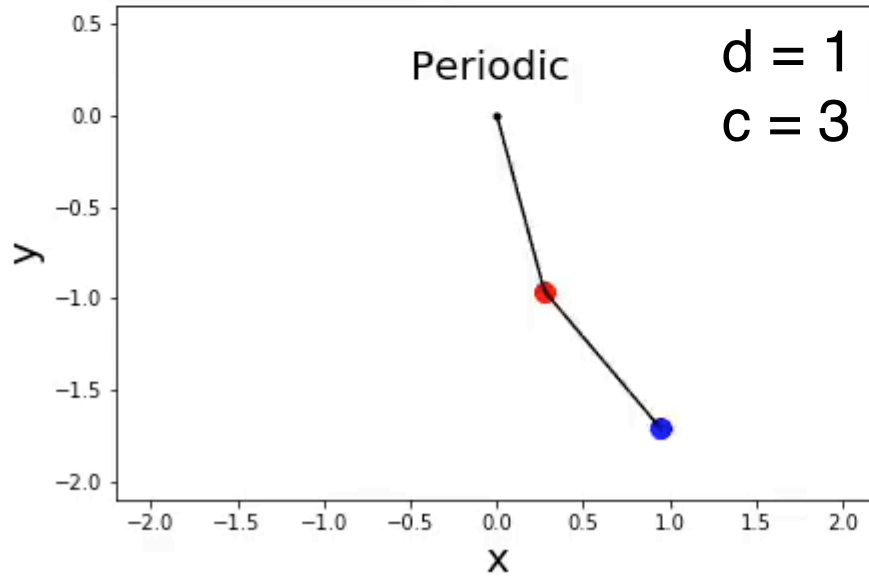
$\theta_0 = 20^\circ$



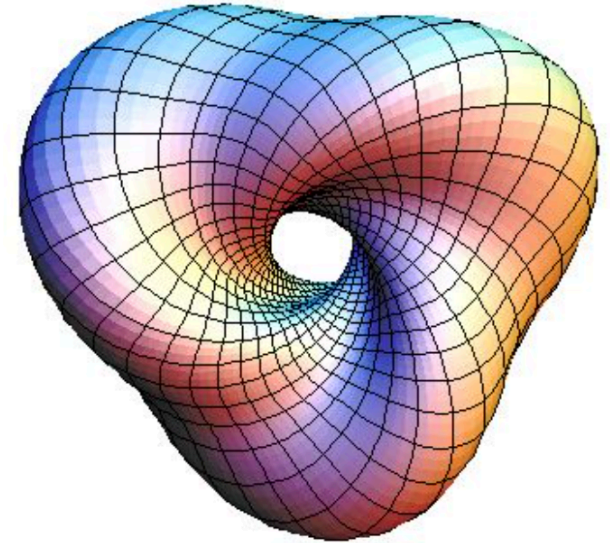
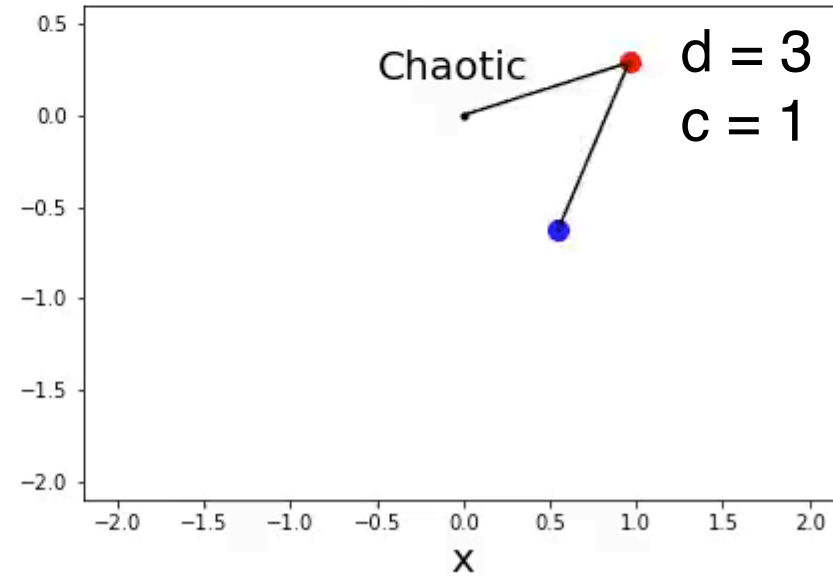
$\theta_0 = 50^\circ$



$\theta_0 = 65^\circ$



$\theta_0 = 90^\circ$

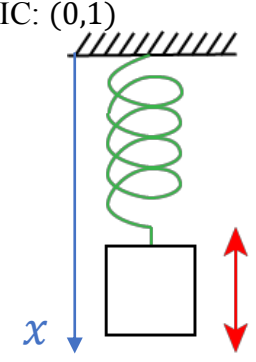
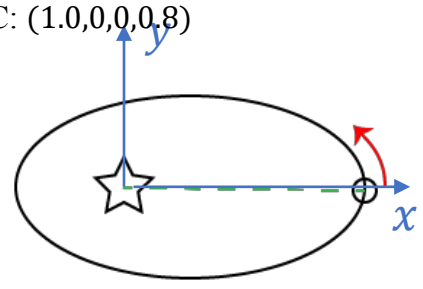
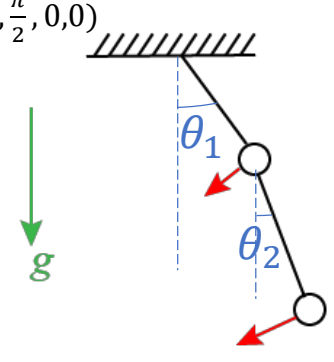
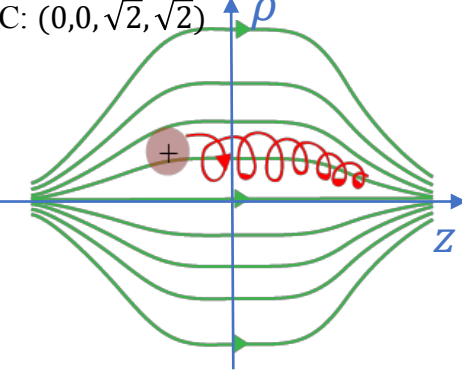
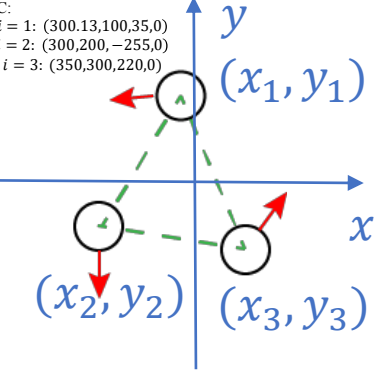


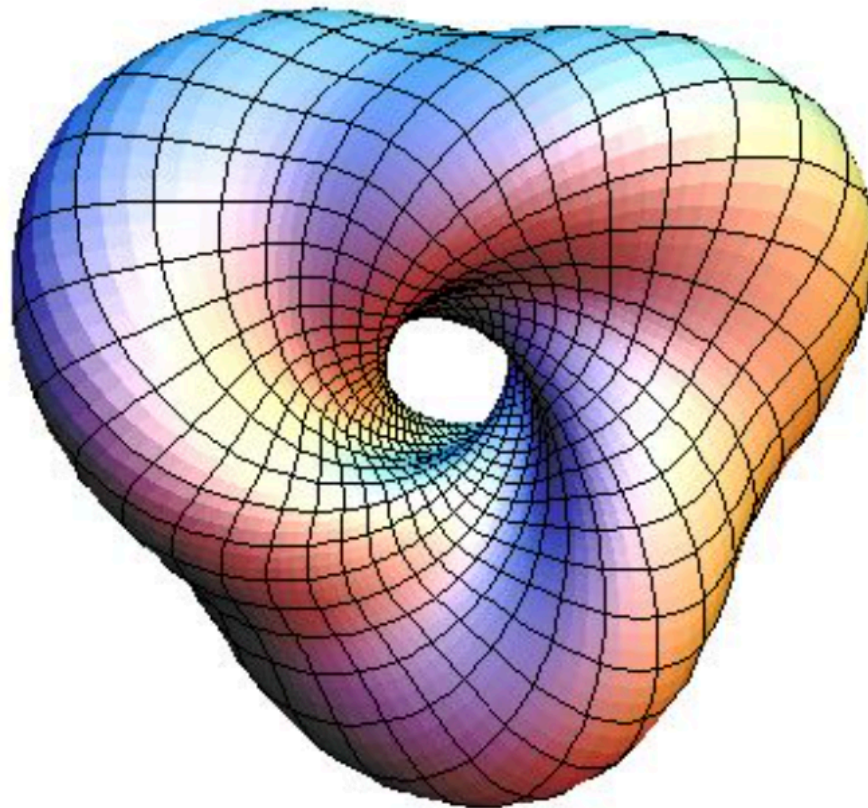
$$c = n - d$$

c = # of conserved quantities

n = phase space dimensionality

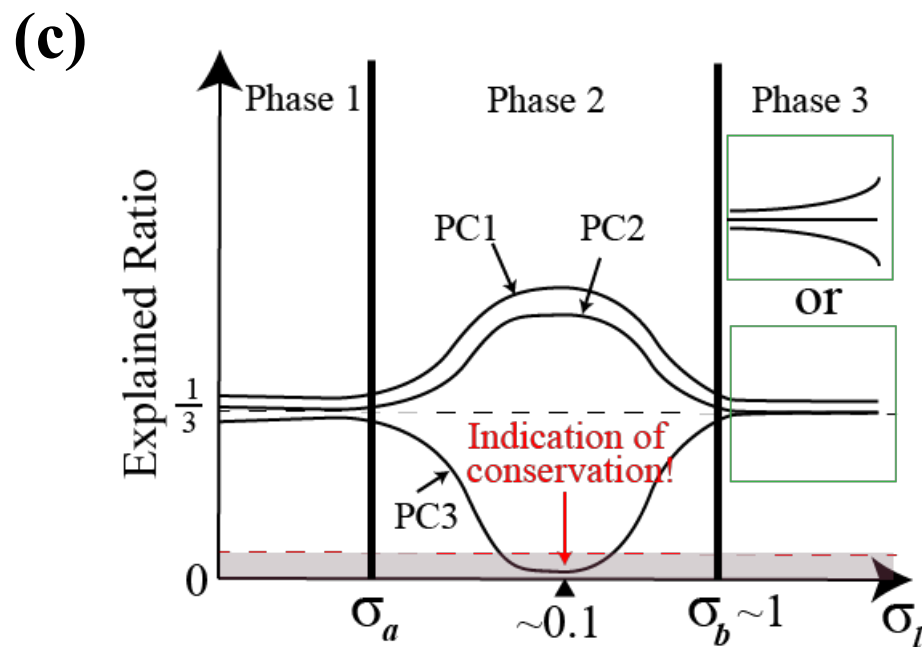
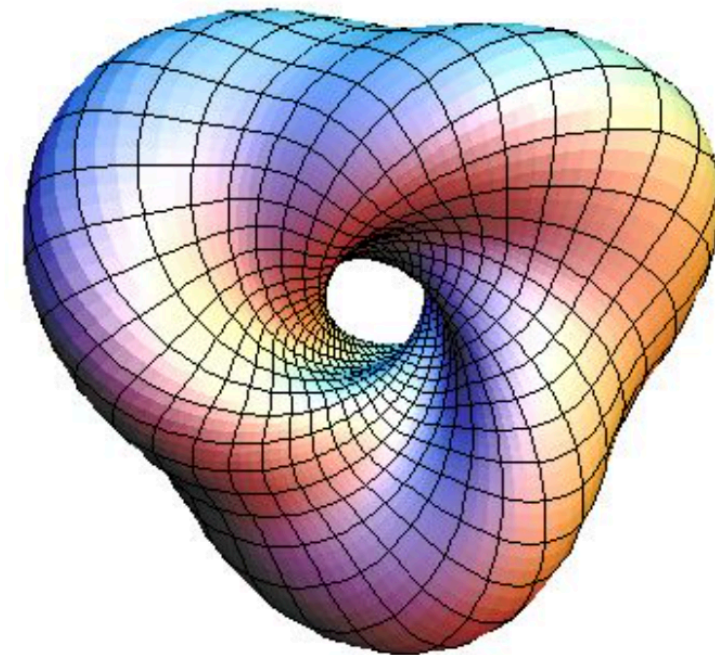
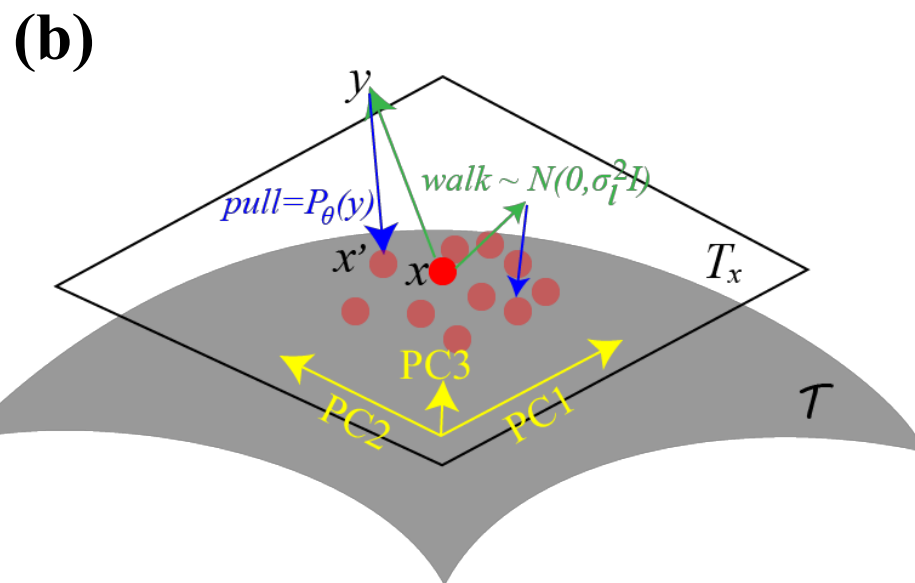
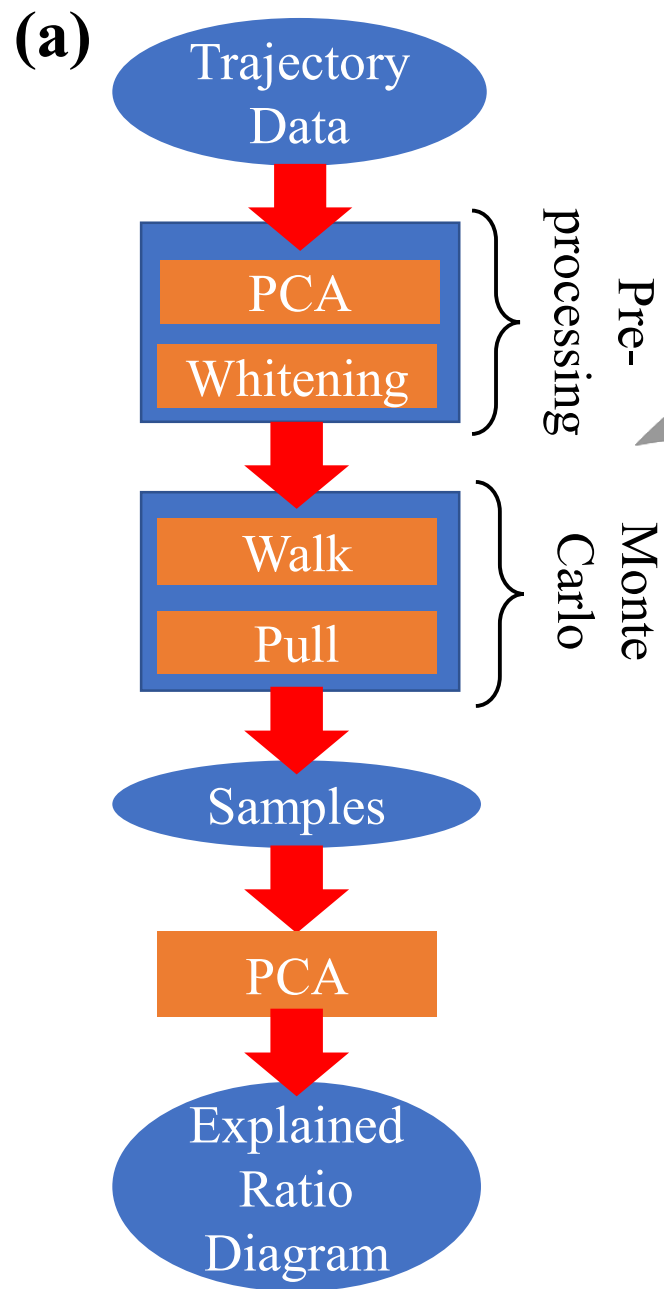
d = state manifold dimensionality

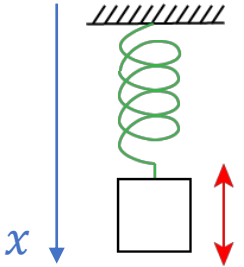
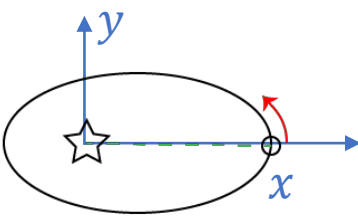
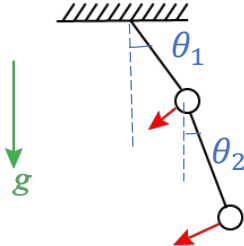
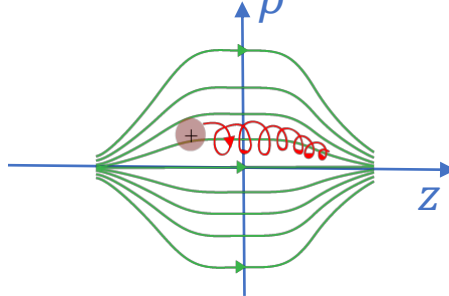
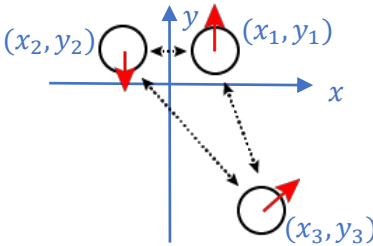
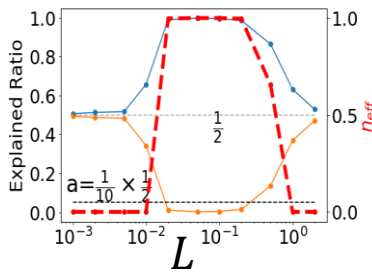
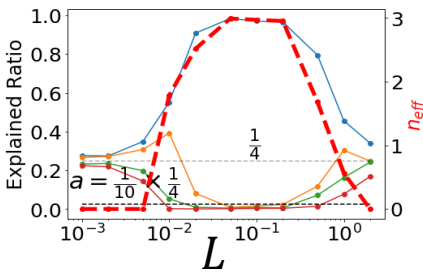
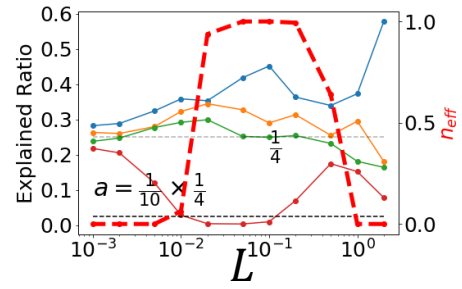
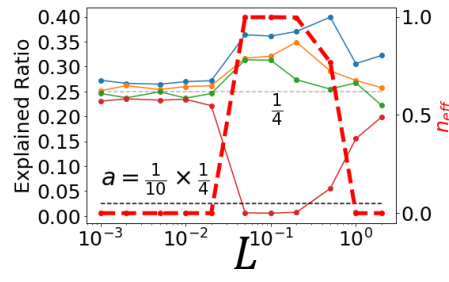
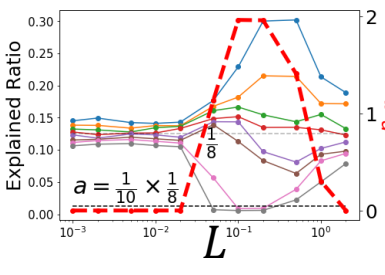
Model	Harmonic Oscillator	Kepler Problem	Double Pendulum	Magnetic Mirror	Three-Body Problem
Illustration	IC: (0,1) 	IC: (1.0,0,0,0,8) 	IC: $(\frac{\pi}{2}, \frac{\pi}{2}, 0, 0)$ 	IC: $(0, 0, \sqrt{2}, \sqrt{2})$ 	IC: $i = 1: (300, 13, 100, 35, 0)$ $i = 2: (300, 200, -255, 0)$ $i = 3: (350, 300, 220, 0)$ 

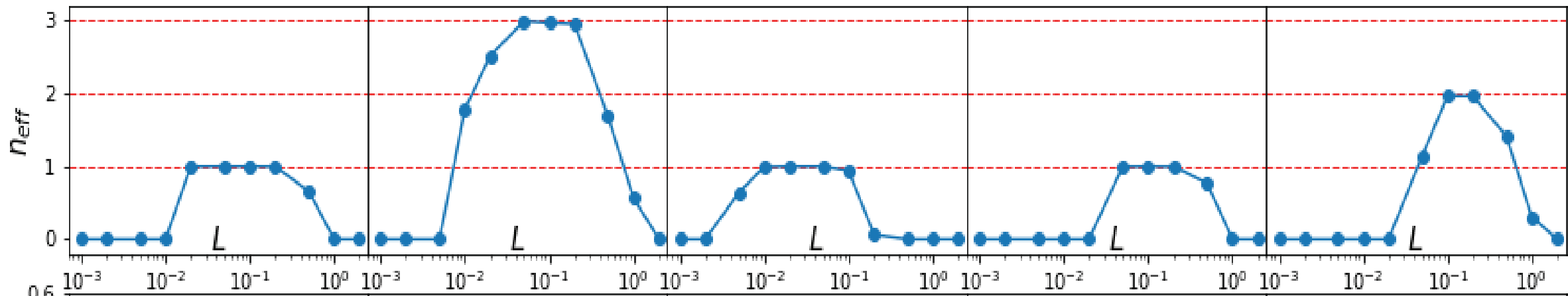


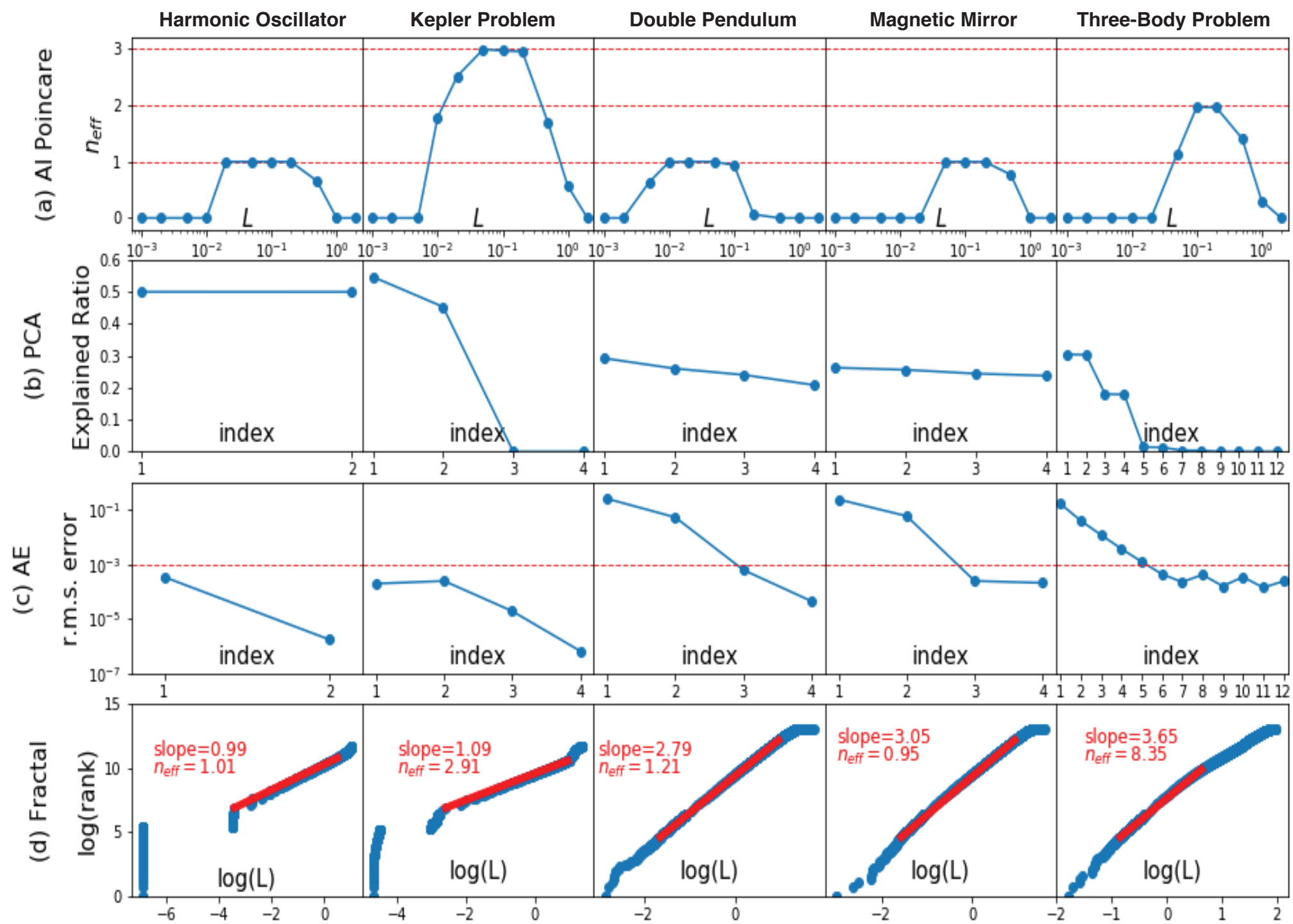
$$c = n - d$$

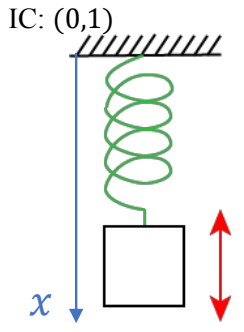
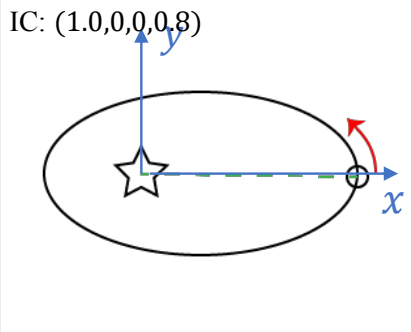
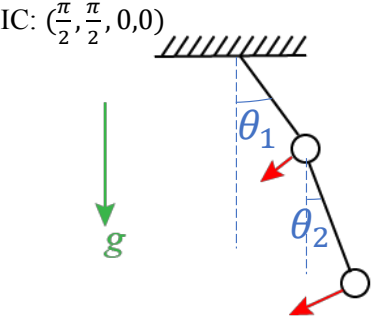
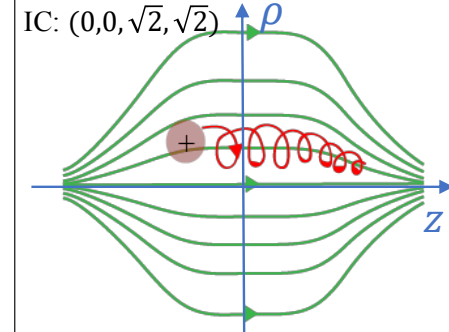
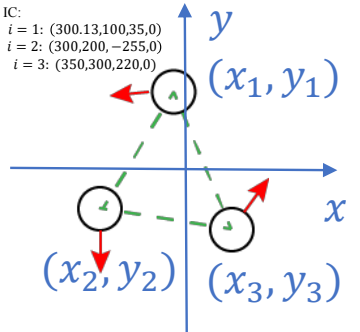
c = # of conserved quantities
 n = of phase space dimensionality
 d = state manifold dimensionality



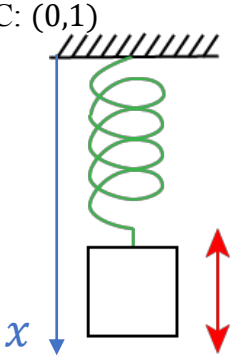
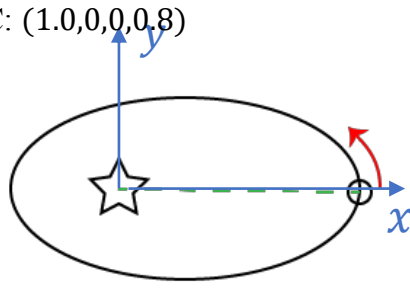
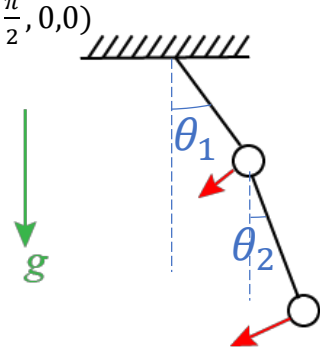
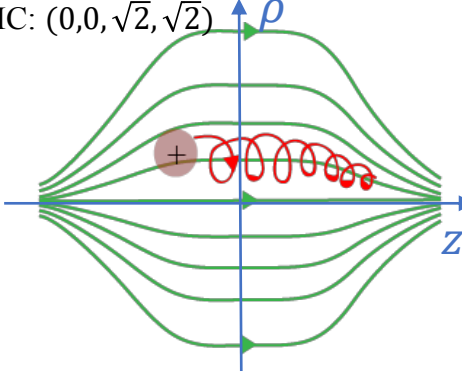
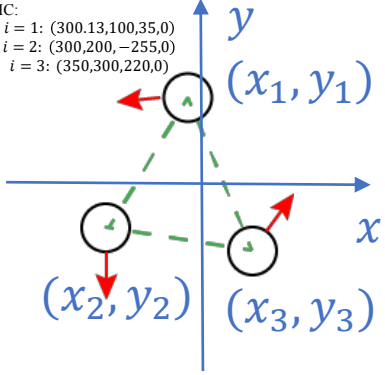
Model	Harmonic Oscillator	Kepler Problem	Double Pendulum	Magnetic Mirror	Three-Body Problem
Illustration	<p>State: (x, \dot{x}) IC: $(0, 1)$</p> 	<p>State: (x, y, \dot{x}, \dot{y}) IC: $(1.0, 0, 0, 0.8)$</p> 	<p>State: $(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$ IC: $(\frac{\pi}{2}, \frac{\pi}{2}, 0, 0)$</p> 	<p>State: $(\rho, z, \dot{\rho}, \dot{z})$ IC: $(0, 0, \sqrt{2}, \sqrt{2})$</p> 	<p>State: $(x_1, y_1, \dot{x}_1, \dot{y}_1, x_2, y_2, \dot{x}_2, \dot{y}_2, x_3, y_3, \dot{x}_3, \dot{y}_3)$ IC: $(300.13, 100.35, 0, 300, 200, -255, 0, 350, 300, 220, 0)$</p> 
$(d, k, N, n, \epsilon, steps)$	$(1, 1, 2, 1, 10^{-2}, 10^3)$	$(2, 1, 4, 3, 10^{-2}, 10^5)$	$(2, 2, 8, 1, 10^{-3}, 10^6)$	$(2, 1, 4, 1, 10^{-3}, 10^5)$	$(2, 3, 12, 6, 10^{-3}, 2 \times 10^5)$
Conservation Law	Energy (1)	Energy (1), Runge-Lenz vector (1), Angular momentum (1)	Energy (1)	Energy (1)	Center of Mass (2), Momentum (2), Energy (1), Angular Momentum (1)
Explained Ratio Diagram					

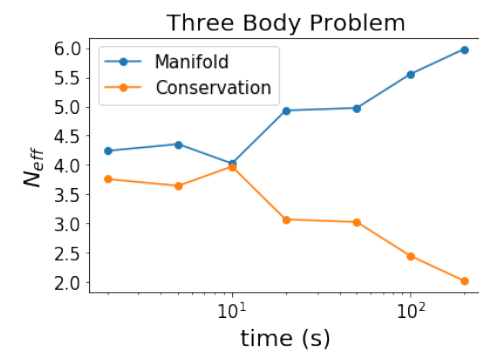
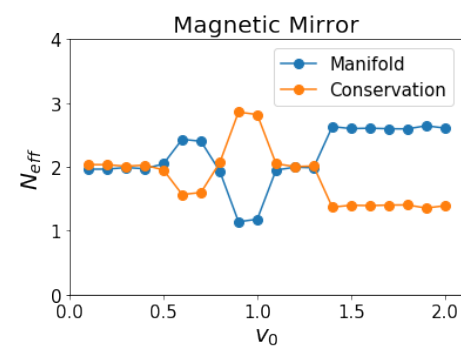
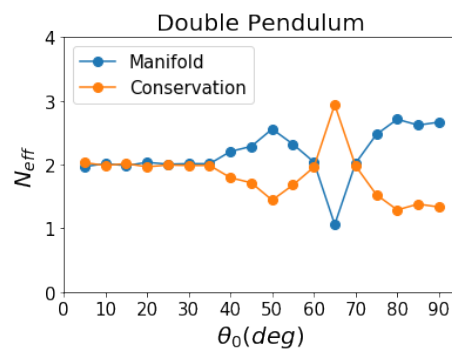
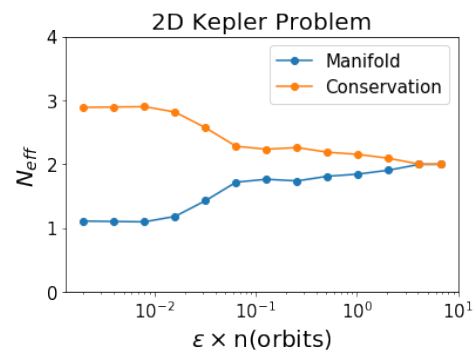




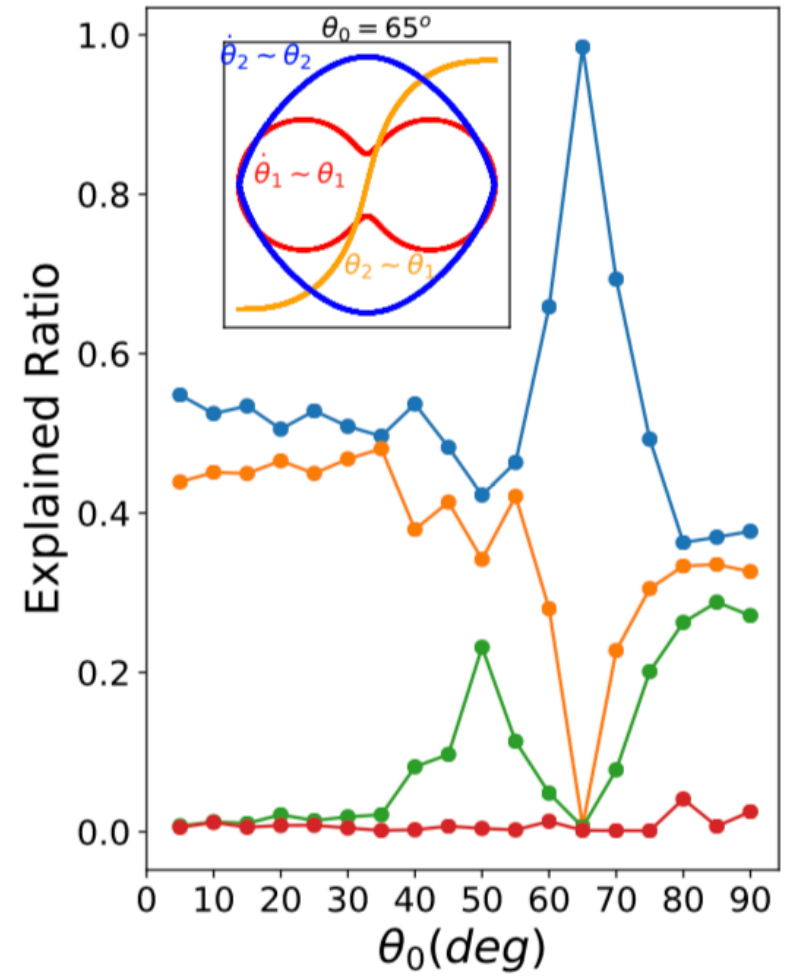
Model	Harmonic Oscillator	Kepler Problem	Double Pendulum	Magnetic Mirror	Three-Body Problem
Illustration	IC: (0,1) 	IC: (1.0,0,0,0,8) 	IC: $(\frac{\pi}{2}, \frac{\pi}{2}, 0, 0)$ 	IC: $(0, 0, \sqrt{2}, \sqrt{2})$ 	IC: $i = 1: (300, 13, 100, 35, 0)$ $i = 2: (300, 200, -255, 0)$ $i = 3: (350, 300, 220, 0)$ 

Model	Conservation	Success
Harmonic Oscillator	$H = \frac{1}{2}(x^2 + \dot{x}^2)$	Yes
Kepler Problem	$H = \frac{1}{2}(\dot{x}^2 + \dot{y}^2) - \frac{1}{\sqrt{x^2 + y^2}}$	Yes
	$L = x\dot{y} - y\dot{x}$	Yes
	$A = \arg(L(-\dot{y}, \dot{x}) - \hat{r})$	No
Double Pendulum	(Small angle) $H_s = 10\theta_1^2 + 5\theta_2^2 + \dot{\theta}_1^2 + \frac{1}{2}\dot{\theta}_2^2 + \dot{\theta}_1\dot{\theta}_2$	Yes
	(Large angle) $H_l = -20\cos\theta_1 - 10\cos\theta_2 + \dot{\theta}_1^2 + \frac{1}{2}\dot{\theta}_2^2 + \dot{\theta}_1\dot{\theta}_2\cos(\theta_1 - \theta_2)$	No
Magnetic Mirror	$H = \frac{1}{2}(\dot{\rho}^2 + \dot{z}^2) + \frac{1}{2}(\rho^2 + \frac{1}{5}z^2 + \rho^2 z^2)$	Yes
Three Body Problem	$x_c = \frac{1}{3}(x_1 + x_2 + x_3) = C_1$	Yes
	$y_c = \frac{1}{3}(y_1 + y_2 + y_3) = C_2$	Yes
	$\dot{x}_c = \frac{1}{3}(\dot{x}_1 + \dot{x}_2 + \dot{x}_3) = C_3$	Yes
	$\dot{y}_c = \frac{1}{3}(\dot{y}_1 + \dot{y}_2 + \dot{y}_3) = C_4$	Yes
	$L = \sum_{i=1}^3 x_i \dot{y}_i - y_i \dot{x}_i$	Yes
	$H = \sum_{i=1}^3 \frac{m}{2}(\dot{x}_i^2 + \dot{y}_i^2) - m^2(\frac{1}{r_{12}} + \frac{1}{r_{13}} + \frac{1}{r_{23}}) \quad (m = 5e6)$	No

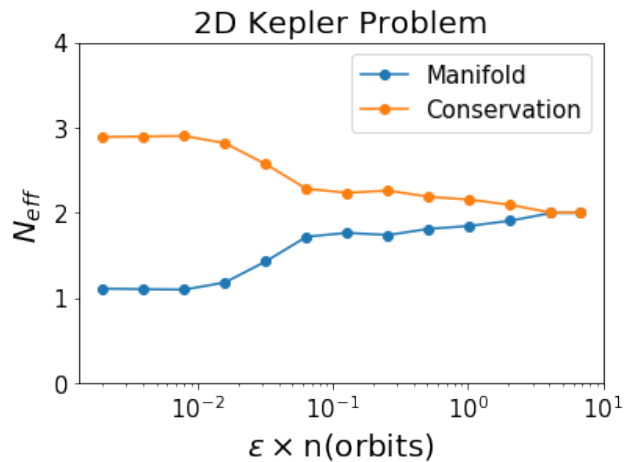
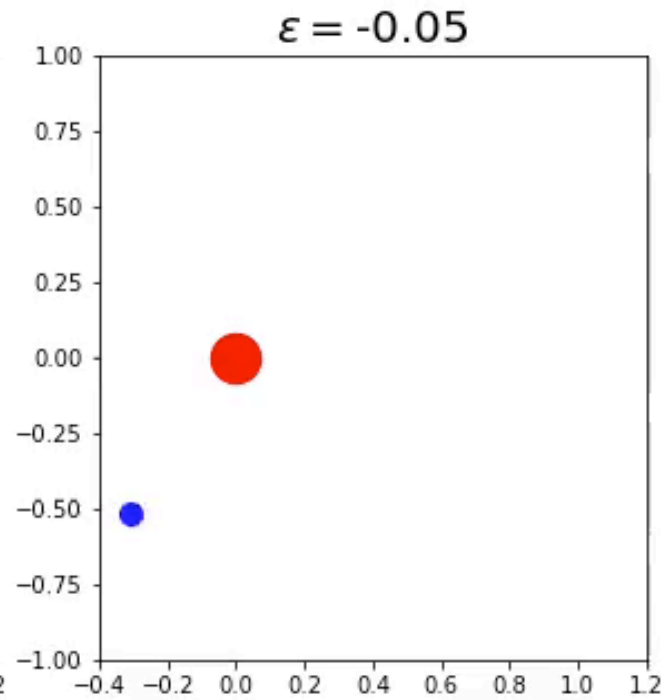
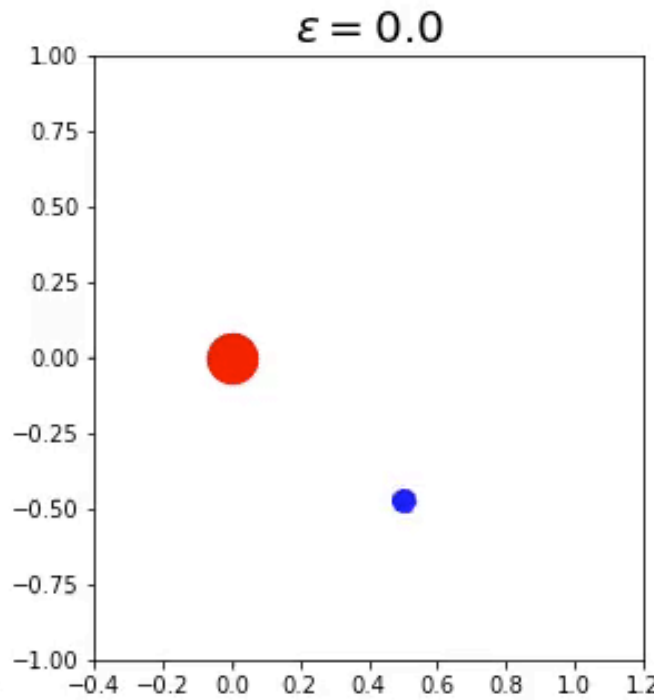
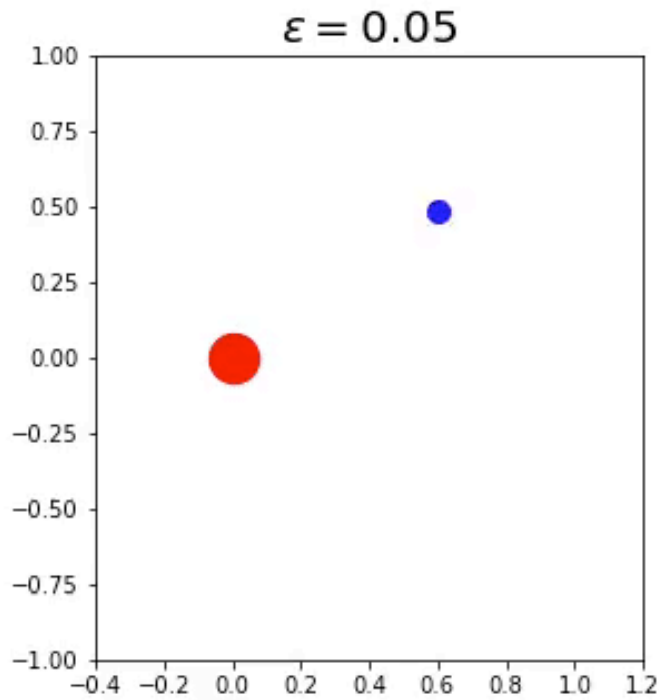
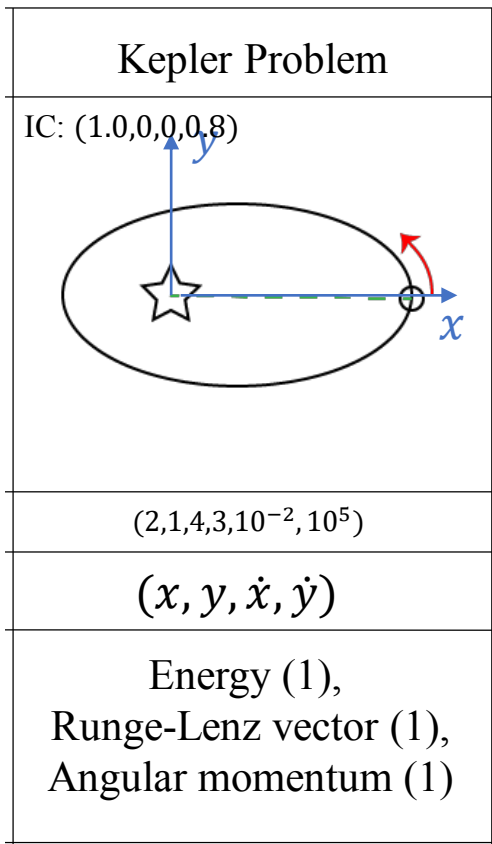
Model	Harmonic Oscillator	Kepler Problem	Double Pendulum	Magnetic Mirror	Three-Body Problem
Illustration	IC: (0,1) 	IC: (1.0,0,0,0.8) 	IC: $(\frac{\pi}{2}, \frac{\pi}{2}, 0, 0)$ 	IC: $(0,0,\sqrt{2},\sqrt{2})$ 	IC: $i = 1: (300.13, 100, 35, 0)$ $i = 2: (300, 200, -255, 0)$ $i = 3: (350, 300, 220, 0)$ 
$(d, k, 2kd, m, \epsilon, N)$	$(1, 1, 2, 1, 10^{-2}, 10^3)$	$(2, 1, 4, 3, 10^{-2}, 10^5)$	$(2, 2, 8, 1, 10^{-3}, 10^6)$	$(2, 1, 4, 1, 10^{-3}, 10^5)$	$(2, 3, 12, 6, 10^{-3}, 2 \times 10^5)$
state	(x, \dot{x})	(x, y, \dot{x}, \dot{y})	$(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$	$(\rho, z, \dot{\rho}, \dot{z})$	$(x_i, y_i, \dot{x}_i, \dot{y}_i) \ i=1,2,3$
Conser- vation Law	Energy (1)	Energy (1), Runge-Lenz vector (1), Angular momentum (1)	Energy (1)	Energy (1)	Center of Mass (2), Momentum (2), Energy (1), Angular Momentum (1)



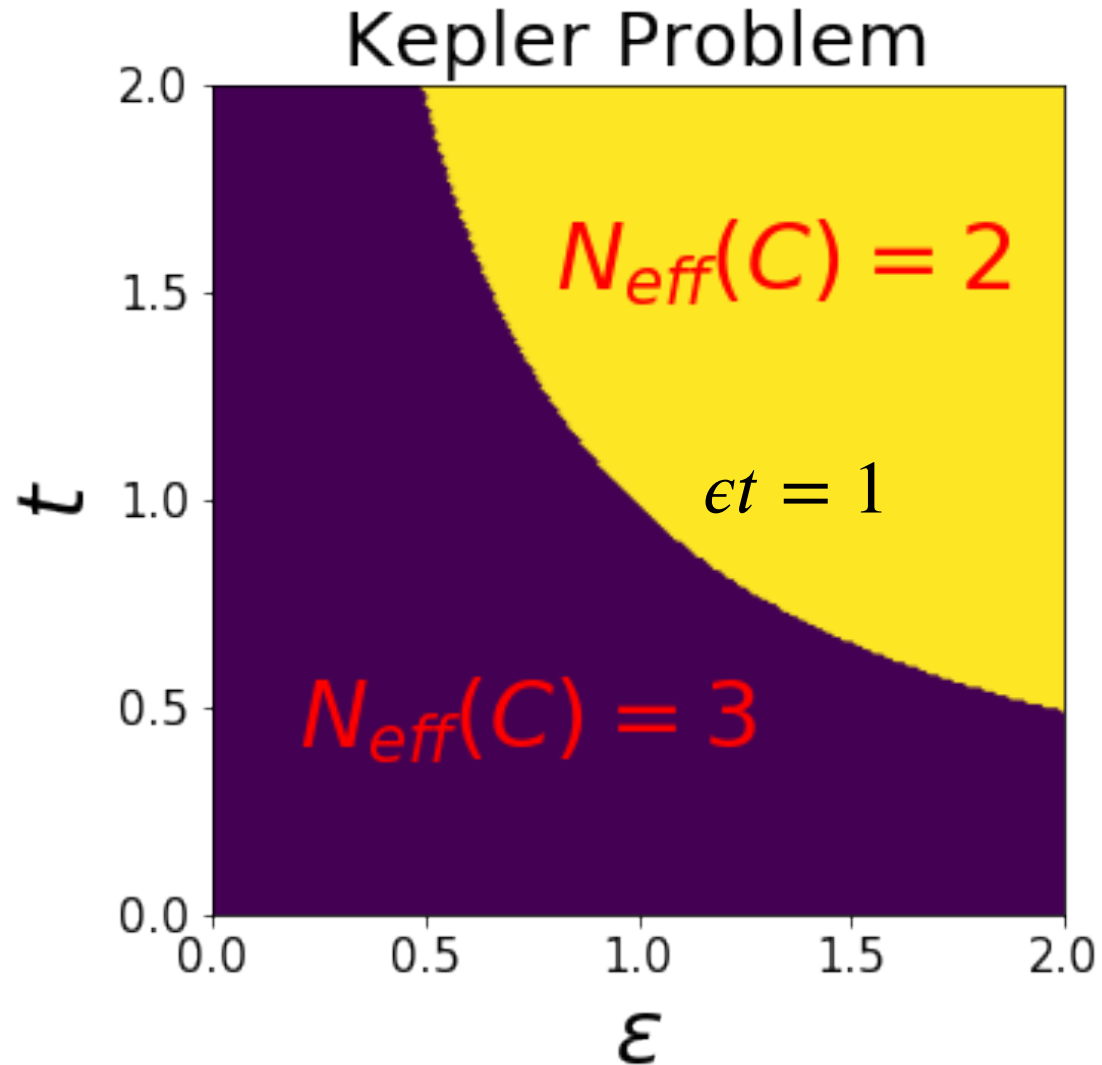
Double pendulum



(b) Double Pendulum



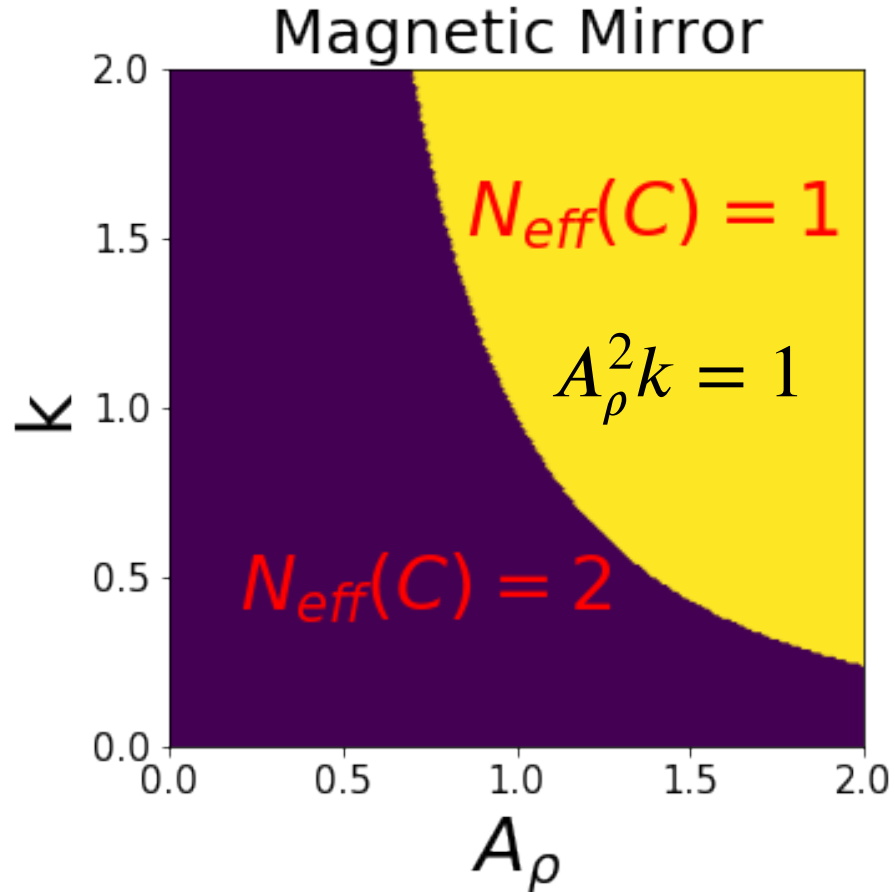
Manifold: $N_{eff}(M) = \sum_i I\left(\lambda_i > \frac{\theta}{n}\right)$, n : total dimension, $\theta = 0.1$
Conservation $N_{eff}(C) = n - N_{eff}(M)$



$$F(r) = \frac{A}{r^{2+\epsilon}}$$

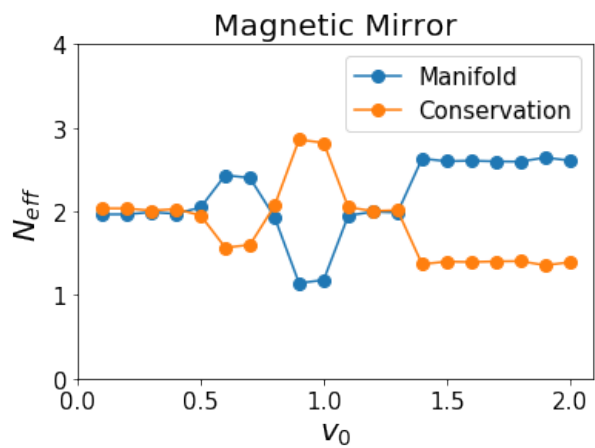
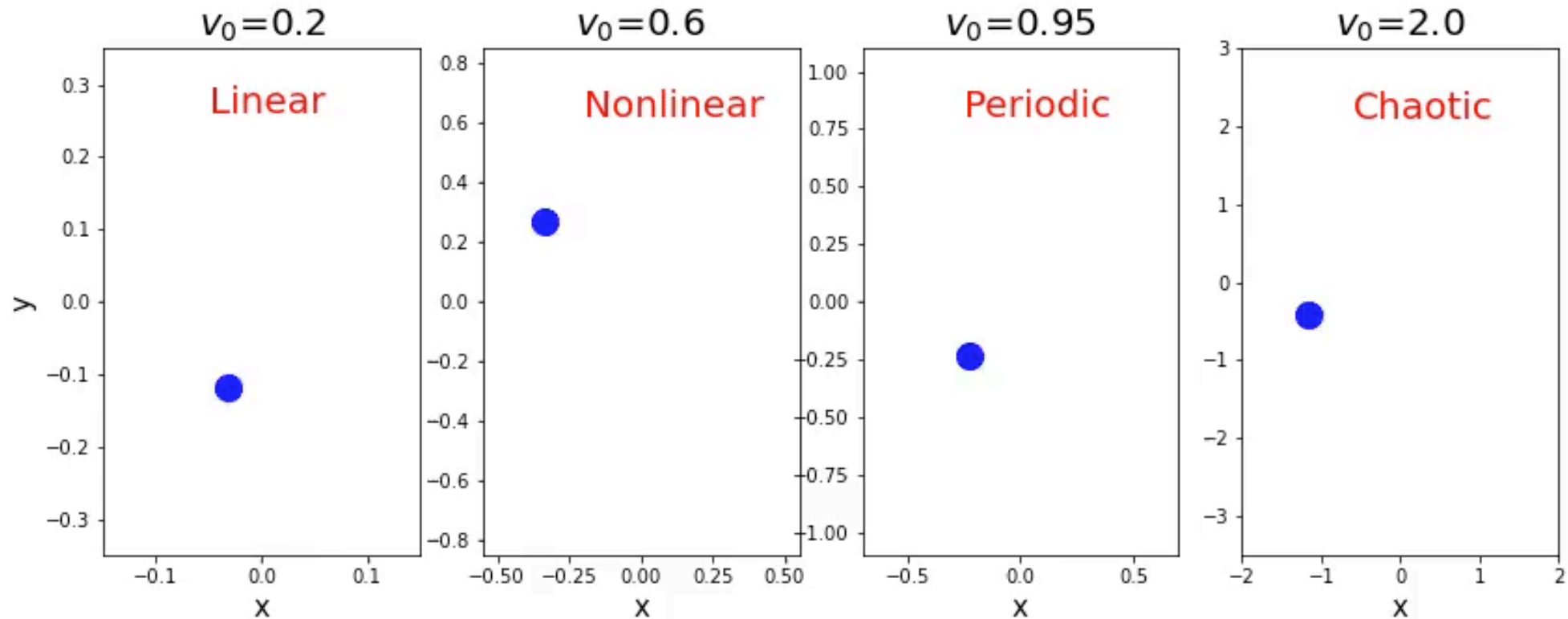
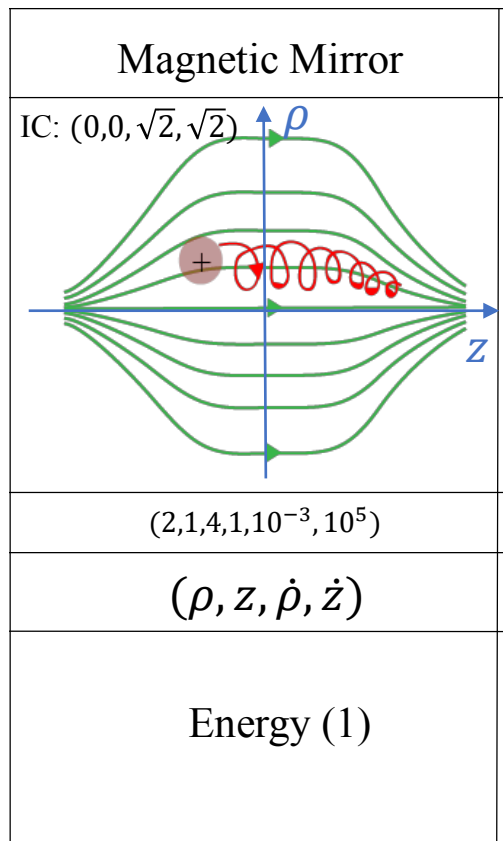
Manifold: $N_{eff}(M) = \sum_i I\left(\lambda_i > \frac{\theta}{n}\right)$, n : total dimension, $\theta = 0.1$

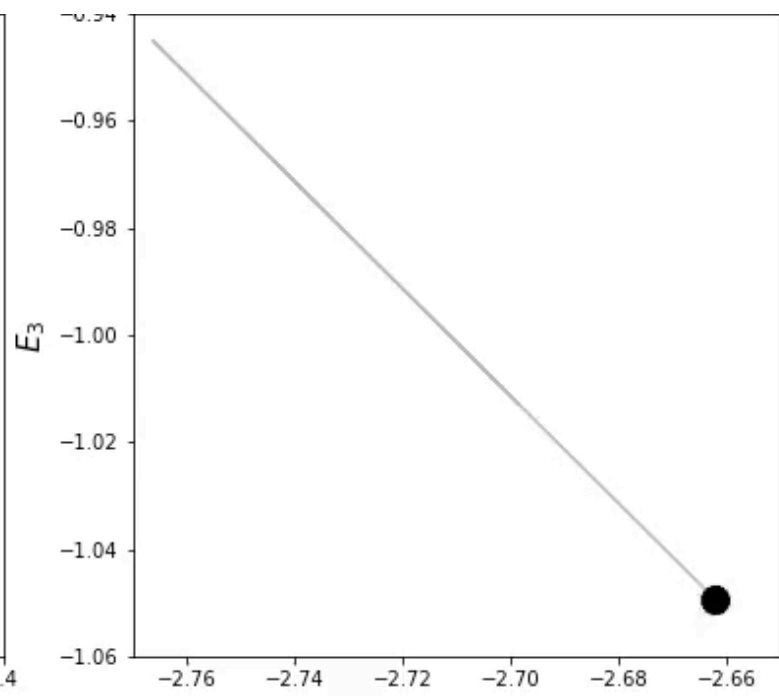
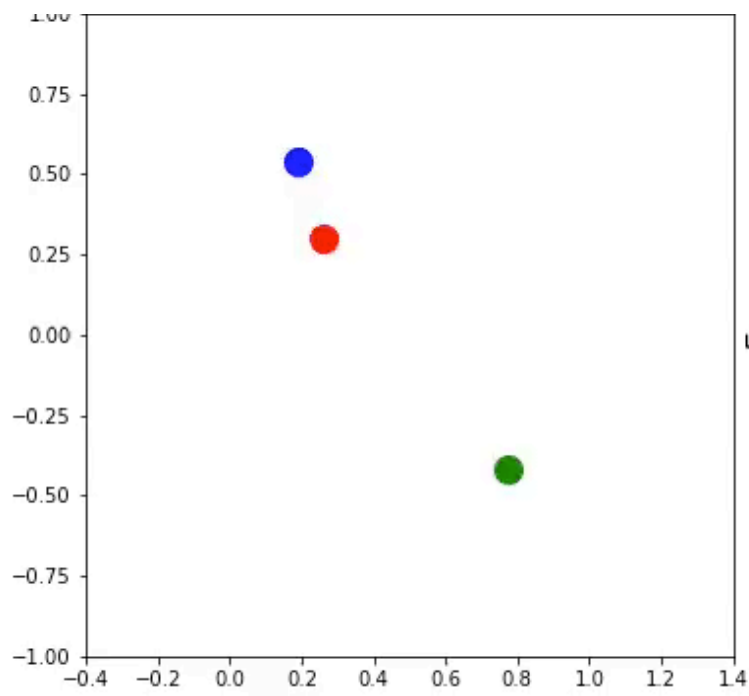
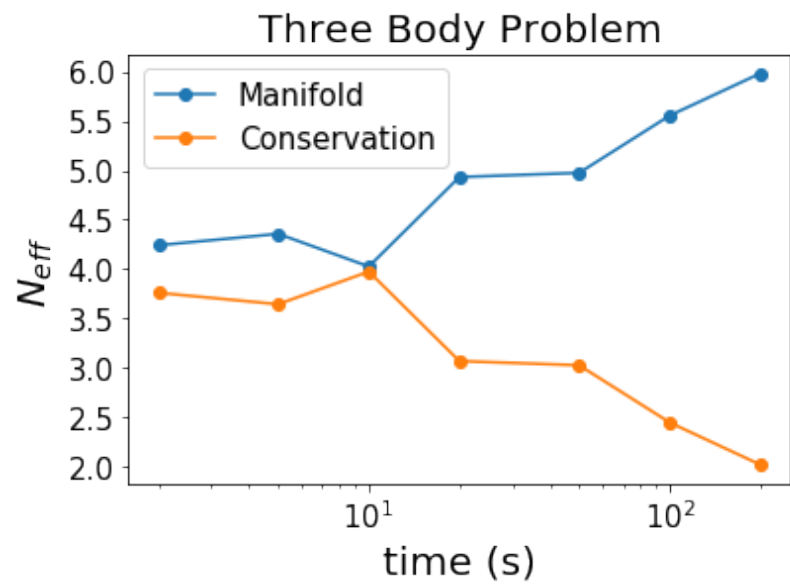
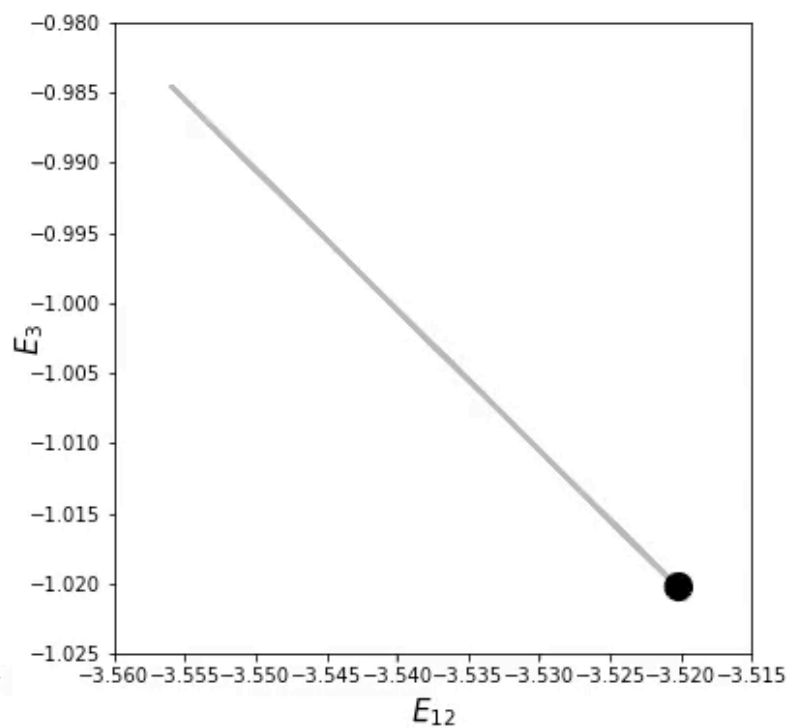
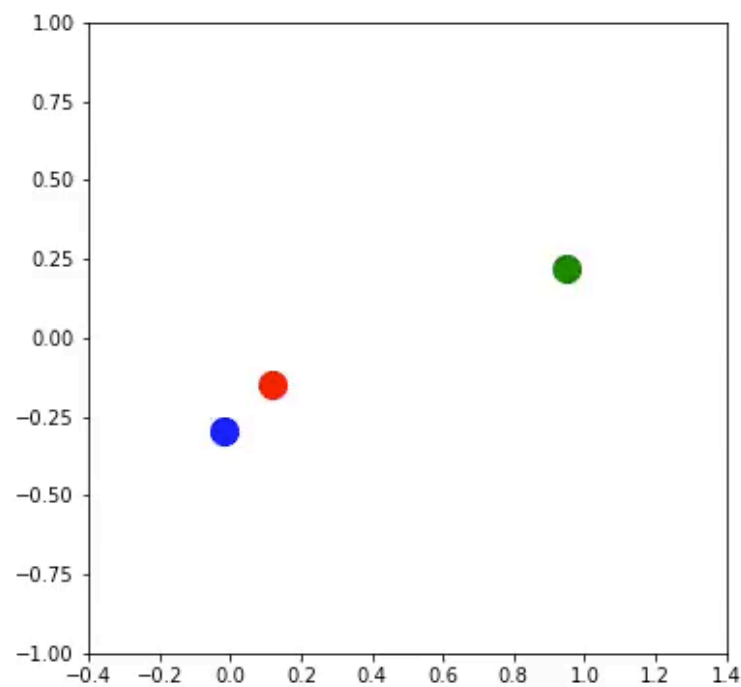
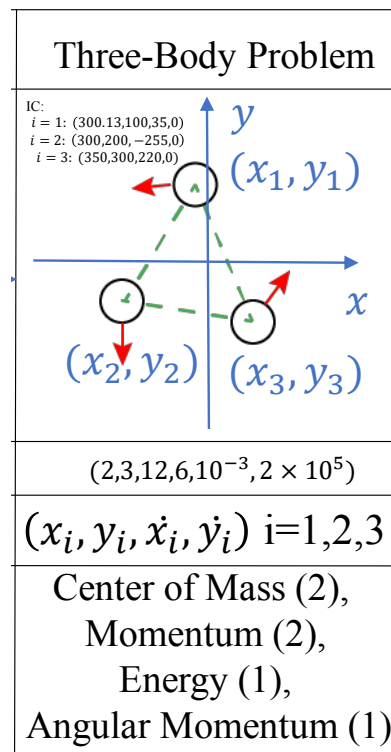
Conservation $N_{eff}(C) = n - N_{eff}(M)$



$$H = \frac{1}{2} \dot{\rho}^2 + \frac{1}{2} \rho^2 + \frac{1}{2} \dot{z}^2 + \frac{1}{2} z^2 + k \rho^2 z^2$$

Note: Amount of time is fixed.
 A_ρ : amplitude in ρ direction





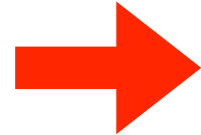
Our focus: Intelligible Intelligence



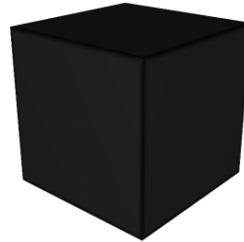
Data

```
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
```

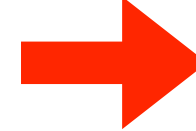
*Neural
network*



Inscrutable
black box model



*Our
focus*



Model we can
understand

$$f(\omega_0, \omega, \theta) = \pi \left(\frac{\omega_0}{\omega} \right)^2 \left[\frac{\omega_0}{\omega} + \frac{\omega}{\omega_0} - \sin^2 \theta \right]$$

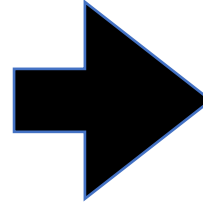
Today I'll discuss auto-discovery of

- ◆ equations (symbolic regression)
- ◆ useful degrees of freedom ("pregression")
- ◆ conserved quantities

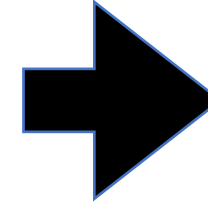
Summary:



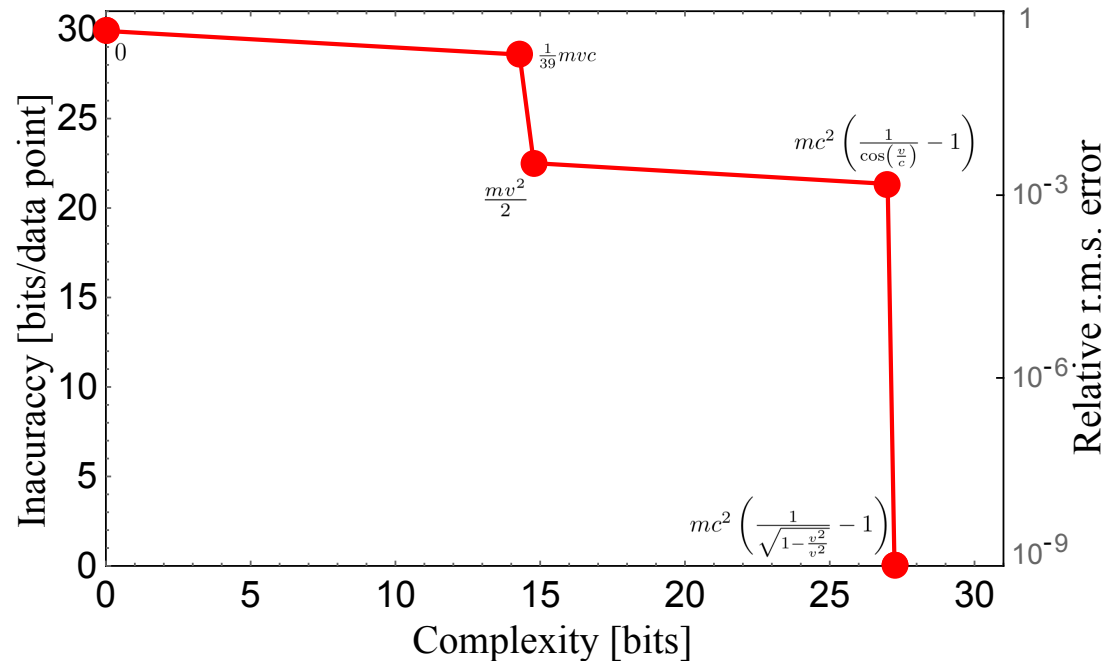
Progression



AI Feynman



$$\begin{aligned}\ddot{x} &= -(x^2 + y^2)x \\ \ddot{y} &= -(x^2 + y^2)y\end{aligned}$$



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

Got fun data we can try this on?