# Towards connecting AIxPhysics and Computational Mechanics

Omar Aguilar

**Abstract**

In the past five years, theoretical physicists, working in areas as diverse as biophysics and cosmology, have become increasingly interested in automating scientific discovery using modern machine learning (ML) methods, thereby establishing an interdisciplinary field known as AIxPhysics. In particular, these efforts have focused on implementing symbolic regression (SR) via ML algorithms, that is, inferring closed-form models from data. Although the models obtained from SR via ML are much more interpretable than those of a typical neural network, their generality is limited by the ad hoc nature of SR via ML. A plausible way to overcome this obstacle is to incorporate techniques from Computational Mechanics (CompMech) that are especially suited for measuring model complexity and separating structure from noise. Conversely, SR via ML might benefit CompMech by presenting a system's transition dynamic as a closed-form model; hence, generating interest in *pattern discovery* [1] beyond the complexity science community. Therefore, the purpose of the following is to be a first step towards this exchange by surveying different SR via ML methods, examining their limitations and proposing possible research intersections that can move CompMech closer to the goal of *automated theory building* [2].

## Contents

# 1 History

By the time of Tycho Brahe's death in 1601, he had collected the world's most accurate and comprehensive data on planetary orbits, which he passed onto Johannes Kepler's hands. After 4 years of toil and about 40 failed attempts to fit the Mars data to different oval shapes, Kepler jump-started modern astronomy by characterizing Mars' orbit as an ellipse [3]. This was the birth of *symbolic regression*: Discovering a compact symbolic expression that accurately matches a given data set.

Three centuries and a half later, two schools of thought that automated symbolic regression emerged: The *cognitive* and the *complexity* traditions. The first one, inspired by cognitive psychology, consisted of developing algorithms that resemble *heuristics*, mental shortcuts that humans use to make decisions with limited effort [4]. The second one, based on dynamical systems theory, reduced a chaotic dataset's observed complexity to a compact, algorithmic specification. Despite the successes of these two research lines, it is only in the past few years that the problem of automating symbolic regression has gained vast popularity [5].

Recently, "mainstream" physicists, have become increasingly interested in developing machine learning programs that automate symbolic regression using biological or physics metaphors. These programs are known as *machine scientists*. Although they use some version of the concept of heuristic, to our knowledge, none of them incorporate the achievements of the complexity tradition [7]. This work is a first step towards filling that important gap.

# 2 Symbolic Regression via Machine Learning

## 2.1 Symbolic Regression via Heuristics

In the 1960s, artificial intelligence (AI) pioneers Herbert Simon and Allen Newell put forward the idea that human problem solving is equivalent to doing *heuristic search* through a problem space. Heuristic search consists of iteratively applying operators to transform the state of a problem from its starting state (in problem state space) to its goal state, guided by heuristics that make search tractable [8].

In the 1970s, Pat Langley implemented and extended Herbert Simon's heuristic approach to the problem of scientific discovery. Langley developed a series of programs called Bacon that rediscovered laws from the history of physics and chemistry. The program was named after Francis Bacon, because Langley incorporated many of Bacon's ideas on the nature of scientific reasoning [4]. Here, we describe how the Bacon program rediscovers Kepler's Third Law.

**Kepler's Third Law**
*The cube of a planet's distance from the Sun is proportional to the square of its period*

$$\frac{d^3}{p^2} = c$$

where $d$ is the distance, $p$ is the period, and $c$ is a constant.

**Three Heuristics for discovering Kepler's Third Law**

1. If the values of a numerical term are constant, then infer that the term always has that value.

2. If the values of two numerical terms increase together, then consider their ratio.

3. If the values of one term increase as those of another decrease, then consider their product.

**Example: Recovering Kepler's Third Law from moon data**

1. Gather some data selecting different values for the nominal variable (moon), obtaining the values of the numerical terms ($d$ and $p$). Consider 4 moons labeled $A$, $B$, $C$ and $D$ that each possess a period $p$ and a distance to the sun $d$

| moon | d | p |
|---|---|---|
| A | 5.67 | 1.77 |
| B | 8.67 | 3.57 |
| C | 14.00 | 7.16 |
| D | 24.67 | 16.69 |

2. Since the values of columns $d$ and $p$ increase together, we apply Heuristic 2 and get $d/p$. We define Term-1 as $d/p$.

| moon | d | p | d/p |
|---|---|---|---|
| A | 5.67 | 1.77 | 3.20 |
| B | 8.67 | 3.57 | 2.43 |
| C | 14.00 | 7.16 | 1.96 |
| D | 24.67 | 16.69 | 1.48 |

3. Since the values of column $d$ increase as those of column $d/p$ decrease, we apply Heuristic 3 and get $d^2/p$. We define Term-2 as $d^2/p$.

| moon | d | p | d/p | $d^2/p$ |
|---|---|---|---|---|
| A | 5.67 | 1.77 | 3.20 | 18.15 |
| B | 8.67 | 3.57 | 2.43 | 21.04 |
| C | 14.00 | 7.16 | 1.96 | 27.40 |
| D | 24.67 | 16.69 | 1.48 | 36.46 |

4. Since the values of column $d^2/p$ increase as those of column $d/p$ decrease, we apply Heuristic 3 and get $d^3/p^2$. We define Term-3 as $d^3/p^2$, which is the product of Term-1 and Term-2.

| moon | d | p | d/p | $d^2/p$ | $d^3/p^2$ |
|---|---|---|---|---|---|
| A | 5.67 | 1.77 | 3.20 | 18.15 | 58.15 |
| B | 8.67 | 3.57 | 2.43 | 21.04 | 51.06 |
| C | 14.00 | 7.16 | 1.96 | 27.40 | 53.61 |
| D | 24.67 | 16.69 | 1.48 | 36.46 | 53.89 |

5. Apply Heuristic 1. The statement that Term-3 is constant across moons is equivalent to Kepler's Third Law [4].

## 2.2 Symbolic Regression via Genetic Algorithms

In the 1960s, John Holland invented genetic algorithms (GAs), a search optimization approach inspired by natural selection and genetic mutation. GAs is a method that transforms a population of "chromosomes" (e.g., strings of 1s and 0s) into a new population by combining "natural selection" with "genetic" operators such as crossover and mutation. Each chromosome is made of "genes" (e.g., bits), where each gene is an ocurrance of a particular "allele" (e.g., 0 or 1) [9]. Here, we provide more detail on each of these operators.

**Three operators of GAs**

1. *Selection operator:* Chooses fitter chromosomes (e.g., strings) to be allowed to reproduce
2. *Crossover operator:* Exchanges subparts of two chromosomes (e.g., subparts of two strings), imitating biological recombination
3. *Mutation operator:* Randomly changes the allele values (e.g., 0 or 1) of some locations in the chromosome [9]

Genetic programming (GP) is a special type of genetic algorithms (GA) that represents "chromosomes" as trees, rather than strings (See Figure. 1 (a)). GP is specially suited for representing subparts of mathematical expressions. Thus, GP can be used as a tool for performing symbolic regression on a population of mathematical expressions. Here we detail an example of how to do symbolic regression via GP.

**Genetic Programming Procedure for doing Symbolic Regression**

1. Generate a population of $N$ possible mathematical solutions to the given problem with different fitness values
2. Select two individual solutions at random from current population
3. Use crossover and mutation to produce two new individual solutions, which are assigned to the next generation (See Figures. 1 (b) and (c))
4. Repeat steps 2 and 3 $N/2$ times to produce a new generation of $N$ mathematical solutions
5. Repeat step 1 to create the next generation [9]

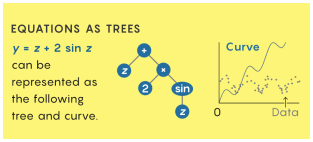


Figure 1 (a): Equation $y = z + 2\sin(z)$ is decomposed into its operators and operands and represented in tree form. Sketches by Kozou Sakai from Quanta magazine [6]

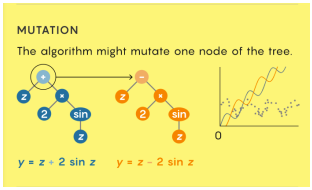


Figure 1 (b): Operator + of equation $y = z + 2\sin(z)$ is mutated into operator $-$. New equation is $y = z - 2\sin(z)$

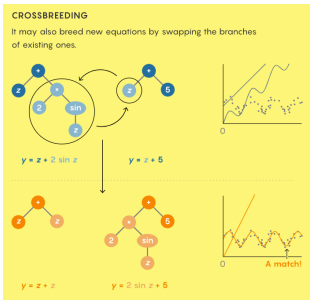


Figure 1 (c): Branch $2\sin(z)$ of equation $y = z + 2\sin(z)$ and branch $z$ of equation $y = z + 5$ are crossbred. New equations are $y = z + z$ and $y = 2\sin(z) + 5$

### 2.3 Symbolic Regression via Physics Metaphors

In the past five years, there has been an increasing interest among physicists on using modern machine learning methods for automating scientific discovery. However, they have identified two main difficulties with using modern neural networks for this purpose [10]:

1. All parts of the data are not generated by just one mechanism
2. Big models are neither interpretable nor generalizable

To address these difficulties, some physicists have decided to combine information theory and physics metaphors. The AI Physicist, developed by Tailin Wu and Max Tegmark, is the archetypical example of this trend [10]. On the information side, Wu and Tegmark leverage the concept of the *description length* of some data, which is the number of bits required to describe such data. On the physics side, they re-define a *theory* $\mathcal{T}$ as the 2-tuple of a prediction function $\mathbf{f}$ and domain $c$ where this function is valid $\mathcal{T} = (\mathbf{f}, c)$ and use four physics metaphors for finding such theories. Here, we provide further detail:

**Description lengths of different types of objects**
Approximate forms of description lengths of integers, rational numbers, real numbers, vectors and functions:

1. DL of natural number $n$: $\log_2(n)$
2. DL of integer number $m$: $\log_2(1+|m|)$
3. DL of rational number $q = \frac{m}{n}$: $\log_2[(1+|m|)n]$
4. DL of vector $\mathbf{p}$: $\sum_i \mathcal{L}_d(p_i)$
5. DL of function $f(\mathbf{x};\mathbf{p})$: $\mathcal{L}_d(\mathbf{p}) + k\log_2 n$ where $n$ basis functions appear $k$ times

**Physics metaphors**

1. *Divide-and-Conquer:* Fit small models one by one, and gradually organize them, instead of fitting a single big one to all the data
2. *Occam's Razor:* Minimize the total description length of a theory (number of bits required to describe that theory)
3. *Unification:* Unify learned theories by introducing parameters
4. *Life-long learning:* Store learned theories in a theory hub and try them on future problems [10]

Since the first two strategies are the least intuitive to understand, we will focus on them:

**Divide-and-Conquer**
Consider an object moving in an environment divided in regions that are each of them governed by different physical laws (See Figure 2).

Conventional neural networks would learn the trajectory of such object by learning a function $\mathbf{f}$ mapping $\mathbf{x}_t \rightarrow \mathbf{y}_t$ by parameterizing $\mathbf{f}$ by some parameter vector $\theta$ that is adjusted to minimize a loss:

$$\mathcal{L} \equiv \sum_t l[f(x_t), y_t]$$

where $f(x_t)$ is the predicted function, $y_t$ is the target, and $l$ is some non-negative distance function quantifying how far each prediction is from the target, usually satisfying $l(y_t, y_t) = 0$. In contrast, the AI Physicist, employs

a Divide-and-Conquer approach that discovers many theories that specialize on different regions. Mathematically, the Divide-and-conquer approach consists of minimizing the generalized-mean loss:

$$\mathcal{L}_\gamma = \sum_t (\frac{1}{M} \sum_{i=1}^{M} l[f_i(x_t), y_t]^\gamma)^{\frac{1}{\gamma}}$$

When $\gamma < 0$, the loss $\mathcal{L}_\gamma$ will be dominated by whichever prediction function $\mathbf{f}_i$ fits each data point best [10].

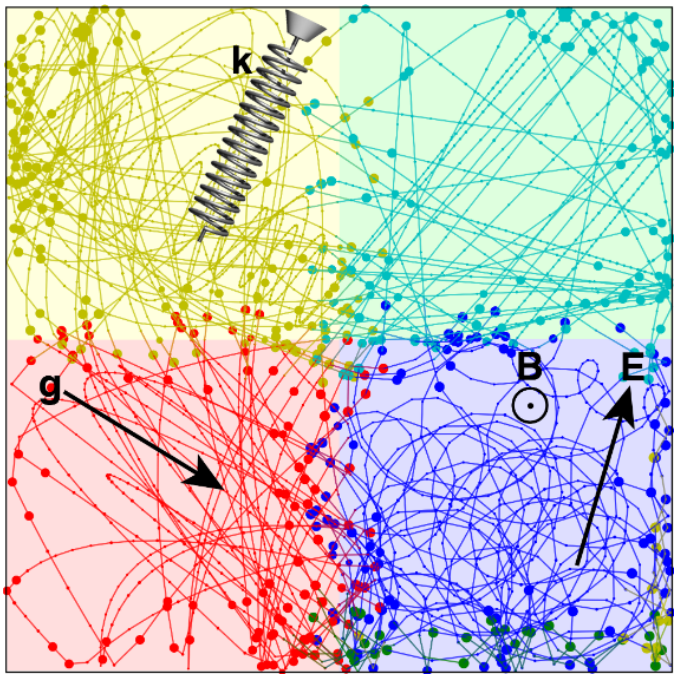


Figure 2: The environment in which a ball moves is divided in four regions, governed by an harmonic potential (upper left quadrant in yellow), a gravitational field (lower left quadrant in red), an electromagnetic field (lower right quadrant in blue) and walls (upper right quadrant in green) [10].

**Occam's Razor**

The AI Physicist uses Occam's Razor to turn theories into compact symbolic theories. For example, the program converts $y = 0.1 + x^{1.9999998}$ into

$y = x^2$. More specifically, the AI Physicist mathematically implements Occam's razor by applying the minimum description length (MDL) principle. This procedure consists of minimizing the description length DL of some theory $\mathcal{T}$ on a dataset $D$, which is given by:

$$\mathrm{DL}(\mathcal{T}, D) = \mathrm{DL}(\mathcal{T}) + \sum_t \mathrm{DL}(u_t)$$

where $\mathrm{DL}(\mathcal{T})$ is the sum of the DLs of the numbers that specify the theory $\mathcal{T}$ and $\sum_t \mathrm{DL}(u_t)$ is the sum of the prediction errors of our theory at time step $t$ [10].

### 2.4 Symbolic Regression on Neural Networks

Symbolic regression has the advantage of inferring models that are both compact and generalizable. However, it has the disadvantage of scaling exponentially with the number of input variables and operators. On the other hand, conventional neural networks efficiently fit complicated models to high-dimensional datasets, but are black boxes that are difficult to generalize and interpret. This begs the question, how can one harness the strengths of both symbolic regression and neural networks? The key idea of Miles Cranmer and his DeepMind collaborators was to train neural networks to model a dataset and then fit symbolic expressions to each part of these neural networks.

**Method for combining symbolic regression and neural networks**

1. Create a neural network with a separable internal modular functions so you have smaller latent spaces.[1]

    - Choose Graph Networks (GNs) as they have inductive biases that are optimal for learning models of interacting particles
    - GNs are structured into three distinct modular functions whose structure is similar to that of physics of interacting particles:
        (a) Edge model $\phi^e$: Maps pair of nodes to a message vector. For the Newtonian dynamics example, this model is analogous to a force on a particle due to another.
        (b) Node model $\phi^v$: Takes receiving node and summed message vector to compute updated node. For the Newtonian

[1] A latent space is a representation of compressed data in which similar data points are closer together in space

dynamics example, this model is analogous to the acceleration on a particle.

(c) Global model $\phi^u$: Aggregates all messages and updated nodes

2. Train these functions on each of the latent spaces

3. Approximate internal functions with symbolic regression.

4. Replace internal functions of neural networks with the symbolic expressions. You end up with symbolic model that is equivalent to neural network (See Figure 3).

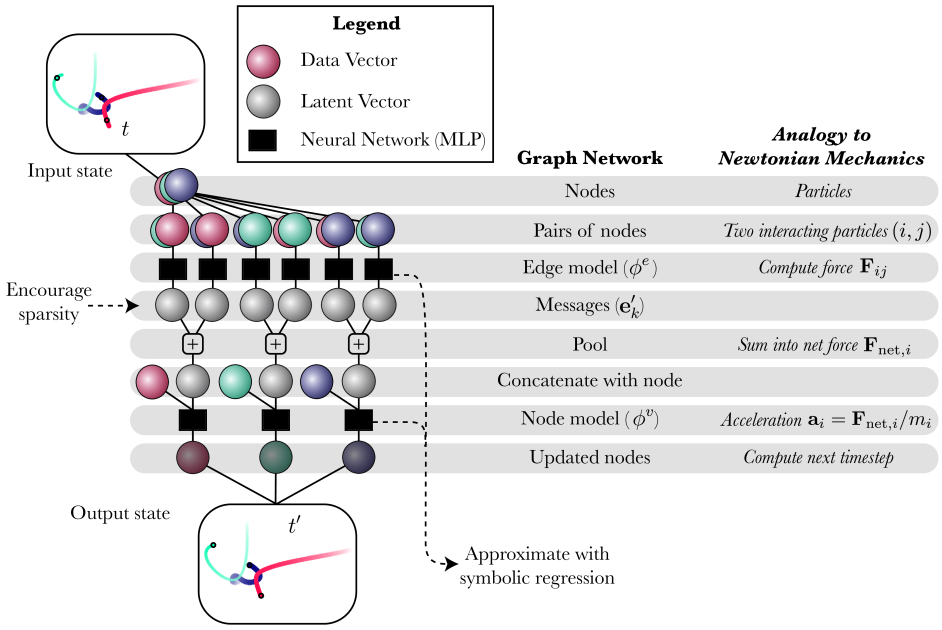


Figure 3: Illustration of the internal structure of graph networks. The rough equivalence between this architecture and physical frameworks allows us to interpret learned formulas in terms of existing physics. [10].

This framework is used to rediscover Newtonian dynamics and to discover the overdensity of a dark matter halo from properties of halos nearby, which is a completely novel property. Here, we show an illustration of these procedures [11].

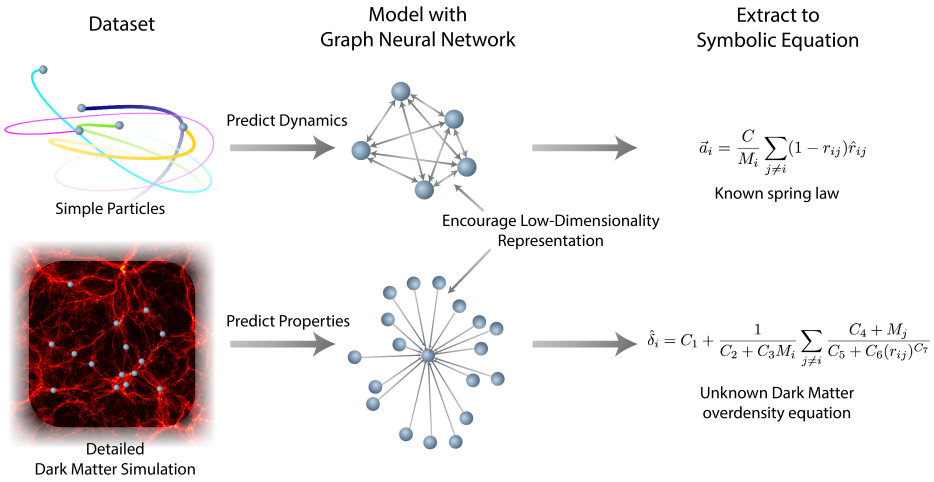


Figure 4: Illustration of procedure for rediscovering known spring law and uknown dark matter overdensity equation [11].

# 3 Limitations of SR via ML

## 3.1 Model complexity is not-well defined

Machine scientists define the complexity of a model as an ad hoc function of the number and the type of operators and operands used, which leads to procedures for inferring optimal equations that are hard to systematize. For example, Wu and Tegmark define the complexity of an expression as the description length of different types of objects using approximate formulas [10]. This definition requires us to insert a strong bias of what a "type" is and to trust rough measures of complexity (e.g., the description length of a rational number $m/n$ is $\mathrm{DL}(m/n) = \log((1+|m|)n)$). In practice, these crude measures of complexity lead researchers to be careless about calculational mistakes. For example, Udrescu and Tegmark calculate the complexity of the classical kinetic energy as $\mathrm{DL}(m*v*v/2) = \mathrm{DL}(2) + k\log_2 n = \log_2 3 + 6\log_2 4 \approx 13.6$ bits. According to their complexity definitions, since $1/2$ is a rational number, the first term of the description length should be $\log_2((1+|1|)2) = \log_2(4) = 2$. However, they treat $1/2$ as an integer and compute $\log_2(1+|2|) = \log_2(3) = 1.58$ bits [12]. Therefore, arbitrary and crude measures of complexity lead to inefficient methods for inferring optimal equations.

Furthermore, arbitrary definitions of complexity are based on the ap-

parent complexity the modeler perceives, which is hard to generalize. For instance, Cranmer and collaborators rely intuitively on the complexity that they perceive some operators have. They weight the complexity of $\wedge$, exp, log, IF$(,,)$ as three times that of $+$, $-$, $*$, / because the authors perceive the operators are "more complex" [11]. Why not two or four times? They provide no explanation. One of the issues with this assumption is that choosing weights arbitrarily leads to no solid reason for why a model should be preferred with respect to another one. Furthermore, different weights may be required for different types of problems. Thus, arbitrary definitions of complexity may lead to good enough models, but their lack of rigour limits their generality.

### 3.2 Difficulty in inferring expressions from noisy data

Machine scientists struggle with inferring equations when noisy data is present. The Bayesian machine scientist is the cutting-edge machine scientist for inferring non-linear differential equations when noise is present and it still fails for high noise. For instance, the Bayesian machine scientist struggles with high noise when recovering Rossler differential equations [13]:

$$\begin{aligned}\dot{x} &= -y - z + \epsilon_x \\ \dot{y} &= x + 0.2y + \epsilon_y \\ \dot{z} &= 0.2 + z(x - 5.7) + \epsilon_z\end{aligned}$$

where $\epsilon$ is a Gaussian random variable with $\sigma_\epsilon = 1$ (low-noise case) or $\sigma_\epsilon = 5$ (high-noise case).

For the low-noise case $\sigma = 1$, the Bayesian machine scientist identifies exactly $\dot{x}$, $\dot{y}$ and $\dot{z}$. For the high-noise case $\sigma = 5$, the Bayesian machine scientist identifies exactly $\dot{x}$, but neither $\dot{y}$ nor $\dot{z}$. The most plausible models are simplified versions of the true models for $\dot{y}$. That is, we get $\dot{y} = x$ instead of $\dot{y} = x + 0.2y$ and $\dot{z} = z(x - a_0)$ instead of $\dot{z} = 0.2 + z(x - 5.7)$. In practice, the most plausible models are almost identical to the true ones. We observe that the machine scientist automatically decreases the complexity of the preferred models as the quality of data decreases [13].

## 4 Further work: Possible research intersections

How can SR via ML benefit CompMech? The answer is not self-evident and the reason for this is that $\epsilon$-machines provide the minimally predictive

causal states and their transition dynamic, which carry more predictive information than closed-form symbolic expressions. However, inferring a closed-form symbolic expression that is (approximately) equivalent to the transition dynamic can lead to interpretable models that non-complexity physicists might find particularly useful for research in biophysics, dark matter, etc. Thus, inspired by SR on NNs, we can apply SR to the transition dynamic itself. Moreover, we can take the inspiration further by replacing Graph Networks with *local causal states*, which are a physics-based unsupervised learning approach for discovering and describing coherent structures [14]. A possible criticism of applying SR via ML to CompMech is that it is not necessary as now there is an equivalence between Reproducing-Kernel Hilbert Space $\epsilon$-machines and stochastic differential equations [15]. However, the value of applying SR via ML lies on the fact that it can potentially bridge the *cognitive* and *complexity* traditions. In summation, SR via ML can aid CompMech to be more accessible to a wider audience of physicists.

## 5 Acknowledgements

I would like to thank Jim P. Crutchfield and Mikhael Semaan for their guidance and patience with me throughout the year. I feel incredibly lucky that I could be part of the fascinating and welcoming UC Davis Complexity Science community. I look forward to deepen my exploration of the intersection between SR via ML and Computational Mechanics and keep working towards *automated theory building*.

## 6 References

## References

[1] C. R. Shalizi and J. P. Crutchfield. Pattern Discovery and Computational Mechanics. *arXiv preprint arXiv: cs/0001027*, 2000.

[2] J. P. Crutchfield. The dreams of theory. *WIRES Comp. Stat.*, 6(March/April): 75–79, 2014.

[3] A. Koyré. The Astronomical Revolution: Copernicus-Kepler-Borelli. *Routledge*, 2013

[4] P. Langley. *Scientific Discovery: Computational Explorations of the Creative Processes*. MIT Press, Cambridge, Massachusetts, 1987

[5] J. P. Crutchfield and B. S. McNamara. Equations of motion from a data series. *Complex Systems*, 1:417 – 452, 1987.

[6] C. Wood. Powerful 'Machine Scientists' Distill the Laws of Physics From Raw Data. Quanra Magazine, 2022.

[7] W. La Cava et al. Contemporary Symbolic Regression Methods and their Relative Performance. *arXiv preprint arXiv:2107.14351v1*, 2021.

[8] H. Simon and A. Newell. Human problem solving: The state of the theory in 1970. *American Psychologist*, 26(2), 145–159, 1971.

[9] M. Mitchell. An introduction to Genetic Algorithms. MIT Press, 1999.

[10] T. Wu and M. Tegmark. Toward an AI Physicist for Unsupervised Learning. *arXiv preprint arXiv:1810.10525v4*, 2019.

[11] M. Cranmer et al. Discovering Symbolic Models from Deep Learning with Inductive Biases. *arXiv preprint arXiv:2006.11287v2*, 2020.

[12] S. M. Udrescu and M. Tegmark. AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity. *arXiv preprint arXiv:2006.10782v2*, 2020.

[13] R. Guimera et al. A Bayesian machine scientist to aid in the solution of challenging scientific problems. *Science Advances*, 2020.

[14] A. Rupe and J.P. Crutchfield. A Local Causal States and Discrete Coherent Structures. *arXiv preprint arXiv:1801.00515v1*, 2018.

[15] N. Brodu and J.P. Crutchfield. Discovering Causal Structure with Reproducing-Kernel Hilbert Space $\epsilon$-Machines. *arXiv preprint arXiv:2011.14821v2*, 2021.