Machine Learning Symmetries of Classical Mechanical Systems

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Matthew Blakeney, Theoretical Physics Undergraduate Student, Trinity College Dublin
Supervisor: Prof. Kirk Soodhalter, Trinity College Dublin

LAIDLAW SCHOLARS

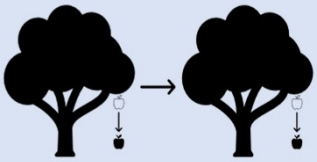
Background

Mechanics is about trying to predict the position/movement of something. That 'something' is called **the system**.

A **Lagrangian,** L, is a quantity of energy that describes the system's state.

What is a symmetry?

In physics, a **symmetry** is some change you can make that won't affect the behaviour of the system.



For example, a tree can be moved horizontally to a different position without affecting the motion of an apple falling out of it.

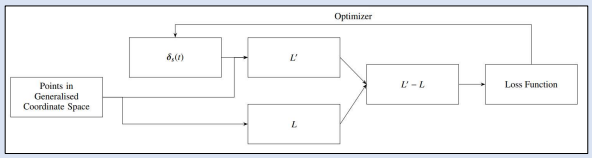
If the difference between a system's Lagrangian before and after a transformation is $L' - L = \frac{d}{dt}\Lambda$, the transformation is a symmetry.

Previous work in machine learning symmetry has searched for different classes of symmetry separately rather than using this general definition.

Why we care about symmetries: Symmetry → Conserved Quantity → Predictions about System's Behaviour

Method

I devised the following process as a way in which symmetry transformations, $\delta_s(t)$, could be discovered using machine learning.



Having come up with an overall process by which to learn symmetries, the real challenge in this project was creating loss functions and ways of parameterising the transformations that would accommodate the process.

Method 1

The first loss function that I created works based on the fact that the following equations are true due to the symmetry of second derivatives:

$$\frac{\partial^2\Lambda}{\partial t\partial q_i} = \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{ki},\dot{\boldsymbol{q}}=0} = \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=\varepsilon,\epsilon_{\dot{q}_k}=0,\dot{q}_k=\delta_{ki}} - \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=\varepsilon,\epsilon_{\dot{q}_k}=0,\dot{q}_k=0} = \frac{\partial^2\Lambda}{\partial q_i\partial t}$$

$$\frac{\partial^2\Lambda}{\partial q_i\partial q_j} = \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{kj},\dot{q}_k=\delta_{ki}} - \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{kj},\dot{q}_k=0}$$
$$= \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{ki},\dot{q}_k=\delta_{kj}} - \Delta\frac{d}{dt}\Lambda\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{ki},\dot{q}_k=0} = \frac{\partial^2\Lambda}{\partial q_j\partial q_i}$$

Then, following loss function is minimized when $L' - L = \frac{d}{dt}\Lambda$:

$$\ell_{symm}(g(\boldsymbol{q},\dot{\boldsymbol{q}},t)) = \sum_{i=1}^{n}\left|\Delta g\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{ki},\dot{\boldsymbol{q}}=0} - \Delta g\Big|_{\epsilon_i=\varepsilon,\epsilon_{\dot{q}_k}=0,\dot{q}_k=\delta_{ki}} + \Delta g\Big|_{\epsilon_i=\varepsilon,\epsilon_{\dot{q}_k}=0,\dot{q}_k=0}\right|$$
$$+ \sum_{i=1}^{n-1}\sum_{m=1}^{n-i}\left|\Delta g\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{k(i+m)},\dot{q}_k=\delta_{ki}} - \Delta g\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{k(i+m)},\dot{q}_k=0}\right.$$
$$\left. - \Delta g\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{ki},\dot{q}_k=\delta_{k(i+m)}} + \Delta g\Big|_{\epsilon_i=0,\epsilon_{\dot{q}_k}=\varepsilon\delta_{ki},\dot{q}_k=0}\right|$$
$$+ \alpha\sum_{j=1}^{10}\left(\left(g\Big|_{\dot{q}_k=1} - g\Big|_{\dot{q}_k=0}\right)\times j + g\Big|_{\dot{q}_k=0} - g\Big|_{\dot{q}_k=j}\right)^2$$

My first method of parameterising transformations allows combinations of rotations and translations to be considered, through the following matrix multiplication:

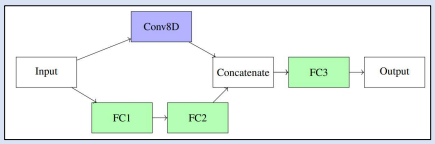
$$\begin{bmatrix}\boldsymbol{q}+\boldsymbol{\delta_s}\\ 1\end{bmatrix} = M\begin{bmatrix}\boldsymbol{q}\\ 1\end{bmatrix}$$

$$M = \begin{bmatrix} 1 & a_1 & a_2 & \cdots & a_{n-1} \\ -a_1 & 1 & a_n & \cdots & a_{2n-2} \\ \vdots & \vdots & 1 & & \vdots \\ \vdots & \vdots & & \ddots & a_{\frac{(n^2+n)}{2}} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

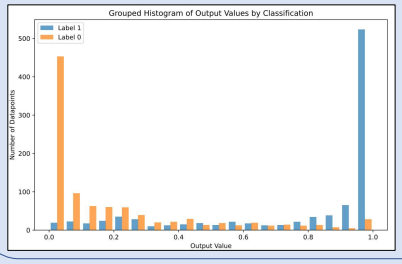
This method is restricted to finding symmetries where Λ is independent of the system's velocities.

Method 2

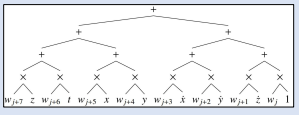
The second loss function that I created was a deep neural network that I trained on a dataset I created of the graphs (in 8-D space) of 190,000 functions. The architecture of this network is depicted below:



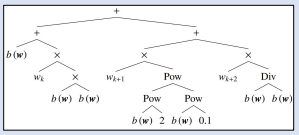
When a new graph is passed to this neural network, it makes a prediction between 0 and 1 as to whether it is a total time derivative. 85.5% accuracy was attained on a test set:



My second method of parameterising transformations involved building branches of an expression tree as follows:



Then these branches were filled in in place of *b(w)* in the following tree:



This method could theoretically find any symmetry of a classical system.

Results

Using Method 1 resulted in correctly identified symmetries. However, as mentioned, this method is unable to find certain kinds of symmetries.

Using Method 2 also resulted in successful symmetry identification, but this application of my process produced noisier results.

Displayed below are the symmetries identified by my process for the Kepler problem (which describes the motion of planets):

δx	δy	δz	Symmetry
ϵy	$-\epsilon x$	0	$x-y$ Rotational Invariance
0	ϵ	$-\epsilon y$	$y-z$ Rotational Invariance
ϵz	0	$-\epsilon x$	$x-z$ Rotational Invariance
ϵL_x	ϵL_y	ϵL_z	Translations $\perp$ to Plane of Motion
ϵL_z	0	$-\epsilon L_x$	SO(4) Symmetry
0	ϵL_z	$-\epsilon L_y$	SO(4) Symmetry
$-\epsilon L_y$	ϵL_x	0	SO(4) Symmetry

Symmetries found by Method 1.

δx	δy	δz	Symmetry
0	$\epsilon\dot{y}$	0	Time Invariance Manifest in y
0	0	$\epsilon\dot{z}$	Time Invariance Manifest in z
$\epsilon\dot{x}$	0	0	Time Invariance Manifest in x

Symmetries found by Method 2.

Conclusions

I was successful in creating a method of finding symmetries of physical systems that doesn't require a prescribed list of classes of symmetries to search for, but rather searches for symmetry based on the general definition of the concept.

My method of finding symmetries is applicable beyond the specific conditions under which I developed it, opening new avenues for further research using similar methods.

My second method gave noisy results with some errors. To improve the accuracy of my symmetry finding method and manifest in reality the theoretical ability to find any symmetry of a classical system, the accuracy of the learned loss function must be improved.

References:
1 Liu, Z. & Tegmark, M. (2022), Machine Learning Hidden Symmetries, *American Physical Society,* 128(18). DOI: 10.1103/PhysRevLett.128.180201

I would like to thank my supervisor, Prof. Kirk Soodhalter, Prof. Chaolun Wu and Christopher Kirwan for helpful discussions over the course of this project and the Laidlaw Foundation for funding the research.