Machine-learning hidden symmetries

Ziming Liu and Max Tegmark
Department of Physics, Massachusetts Institute of Technology, Cambridge, USA
(Dated: May 10, 2022)

We present an automated method for finding hidden symmetries, defined as symmetries that become manifest only in a new coordinate system that must be discovered. Its core idea is to quantify asymmetry as violation of certain partial differential equations, and to numerically minimize such violation over the space of all invertible transformations, parametrized as invertible neural networks. For example, our method rediscovers the famous Gullstrand-Painlevé metric that manifests hidden translational symmetry in the Schwarzschild metric of non-rotating black holes, as well as Hamiltonicity, modularity and other simplifying traits not traditionally viewed as symmetries.

arXiv:2109.09721v2 [cs.LG] 6 May 2022

INTRODUCTION

Philip Anderson famously argued that "It is only slightly overstating the case to say that physics is the study of symmetry" [1], and discovering symmetries has proven enormously useful both for deepening understanding and for solving problems more efficiently, in physics [1–3] as well as machine learning [4–10].

Discovering symmetries is useful but hard, because they are often not *manifest* but *hidden*, becoming manifest only after an appropriate coordinate transformation. For example, after Schwarzschild discovered his eponymous black hole metric, it took 17 years until Painlevé, Gullstrand and Lemaître showed that it had hidden translational symmetry: they found that the spatial sections could be made translationally invariant with a clever coordinate transformation, thereby deepening our understanding of black holes [11]. As a simpler example, Fig. 1 shows the same vector field in two coordinates systems where rotational symmetry is manifest and hidden, respectively.

Our results below are broadly applicable because they apply to a very broad definition of symmetry, including not only *invariance* and *equivariance* with respect to arbitrary Lie groups, but also *modularity* and *Hamiltonicity*. If a coordinate transformation is discovered that makes such simplifying properties manifest, this can not only deepen our understanding of the system in question, but also enable an arsenal of more efficient numerical methods for studying it.

Discovering hidden symmetries is unfortunately highly non-trivial, because it involves a search over all smooth invertible coordinate transformations, and has traditionally been accomplished by scientists making inspired guesses. The goal of this *Letter* is to present a machine learning method for automating hidden symmetry discovery. Its core idea is to quantify asymmetry as violation of certain partial differential equations, and to numerically minimize such violation over the space of all invertible transformations, parametrized as invertible neural networks. For example, the neural network automatically learns to transform Fig. 1(b) into Fig. 1(c), thereby making the hidden rotational symmetry manifest. Our method differs from previous work [9, 10, 12–15] that exploits manifest symmetries, partial differential equations or other physical properties to facilitate machine learning, but not the other way around to discover hidden symmetries with machine learning as a tool.

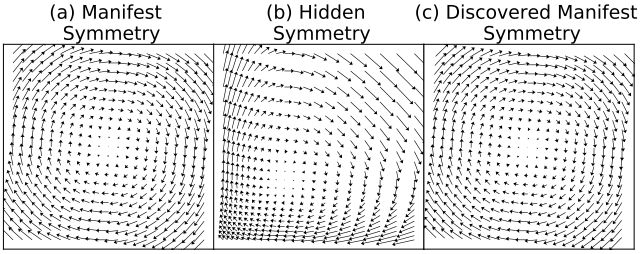

FIG. 1: 1D harmonic oscillator phase space flow vector field $\mathbf{f}(x,p) = (p, -x)$. The rotational symmetry of $\mathbf{f}$ is manifest in (a) and hidden in (b). Our algorithm can reveal the hidden symmetry by auto-discovering the transformation from (b) to (c).

TABLE I: PDE and Losses for Generalized Symmetries

Generalized symmetry	Linear operator $\hat{L}$	Loss ℓ	Examples
Translation invariance	$\hat{L}_j = \partial_j$	$\ell_{\rm TI}$	A,E,F
Lie invariance	$\hat{L}_j = K_j\mathbf{z}\cdot\nabla$	$\ell_{\rm INV}$	E,F
Lie equivariance	$\hat{L}_j = K_j\mathbf{z}\cdot\nabla \pm K_j$	$\ell_{\rm EQV}$	B
Canonical eqvariance	$\hat{L}^{\mathbf{x}}_j = K_j\mathbf{x}\cdot\nabla_{\mathbf{x}} - K^t_j\mathbf{p}\cdot\nabla_{\mathbf{p}} + K^t_j$ $\hat{L}^{\mathbf{p}}_j = K_j\mathbf{x}\cdot\nabla_{\mathbf{x}} - K^t_j\mathbf{p}\cdot\nabla_{\mathbf{p}} - K_j$	$\ell_{\rm CAN}$	C
Hamiltonicity	$\hat{L}_{ij} = -\mathbf{m}^t_i\partial_j + \mathbf{m}^t_j\partial_i$	$\ell_{\rm H}$	A,B,C,D
Modularity	$\hat{L}_{ij} = \mathbf{A}_{ij}\hat{\mathbf{z}}^t_i\partial_j$	$\ell_{\rm M}$	D

In the Method section, we introduce our notation and symmetry definition and present our method for hidden symmetry discovery. In the Results section, we apply our method to classical mechanics and general relativity examples to test its ability to auto-discover hidden symmetries.

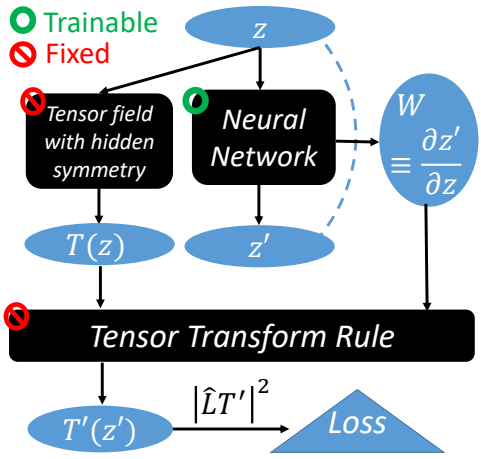

FIG. 2: Schematic workflow of our algorithm for discovering hidden symmetry

METHOD

PDEs encoding generalized symmetries

We seek to discover symmetries in various tensor fields $T(\mathbf{z})$ for $\mathbf{z} \in \mathbb{R}^n$, for example the vector field $\mathbf{f}(\mathbf{z})$ (a rank-1 tensor) defining a dynamical system $\mathbf{z}(t)$ through a vector differential equation $\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z})$, or the metric $g(\mathbf{z})$ (a rank-2 tensor) quantifying spacetime geometry in general relativity. We say that a tensor field T has a *generalized symmetry* if it obeys a linear partial differential equation (PDE) $\hat{L}T = 0$, where $\hat{L}$ is a linear operator that encodes the symmetry generators. This definition covers a broad range of interesting situations, as illustrated by the examples below (see Table I for a summary).

Translational Invariance: A tensor field T is invariant under translation in the $\hat{j^{\text{th}}}$ coordinate direction $\hat{\mathbf{z}}_j$ if $T(\mathbf{z} + a\hat{\mathbf{z}}_i) = T(\mathbf{z})$ for all $\mathbf{z}$ and a, which is equivalent to satisfying the PDE $\partial T/\partial z_j = 0$, corresponding to the linear operator $\hat{L} = \partial_j$.

Lie invariance & equivariance: If $T(\mathbf{z})$ satisfies $T(g\mathbf{z}) = g^n T(\mathbf{z})$ for all elements g of some Lie group $\mathcal{G}$ and an integer $n = -1$, 0 or 1, then we say that T is *invariant* if $n = 0$, and *equivariant* otherwise ($n = 1$ corresponds to a *covariant* (1,0) vector field and $n = -1$ corresponds to a *contravariant* (0,1) vector field) [1]. Taking the derivative on the both sides of the identity $T(e^{K_j a}\mathbf{z}) = e^{nK_j a}T(\mathbf{z})$ with respect to a at $a = 0$ gives the PDEs $\hat{L}_j\,\mathbf{f} = 0$ with $\hat{L}_j \equiv K_j\mathbf{z}\cdot\nabla - nK_j$. Figure 1 (a) and (c) show examples of rotational equivariance.

[1] An (m,n)-tensor means the tensor has m covariant and n contravariant indices, which is a convenient notations especially in general relativity when dealing with metrics. We call a (1,0)-tensor a covariant vector, and a (0,1)-tensor a contravariant vector.

Hamiltonicity (a.k.a. *symplecticity*): A dynamical system $\mathbf{z}(t) \in \mathbb{R}^{2d}$ obeying a vector differential equation $\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z})$ is called *Hamiltonian* or *symplectic* if $\mathbf{f} = \mathbf{M}\nabla H$ for a scalar function H, where

$$\mathbf{M} \equiv \begin{pmatrix} 0 & \mathbf{I} \\ -\mathbf{I} & 0 \end{pmatrix}, \tag{1}$$

and $\mathbf{I}$ is the $d \times d$ identity matrix. Such systems are of great importance in physics, where it is customary to write $\mathbf{z} = (\mathbf{x}, \mathbf{p})$, because the Hamiltonian function $H(\mathbf{z})$ (interpreted as energy) is a conserved quantity under the system evolution $\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z}) = (\dot{\mathbf{x}}, \dot{\mathbf{p}}) = \mathbf{M}\nabla H = (\partial_{\mathbf{p}} H, -\partial_{\mathbf{x}} H)$. Hamiltonicity thus corresponds to $\mathbf{M}^{-1}\mathbf{f}$ being a gradient, *i.e.*, to its generalized curl (the antisymmetric parts of its Jacobian matrix) vanishing. Letting $\mathbf{J} \equiv \nabla\mathbf{f}$ denote the Jacobian of $\mathbf{f}$ ($J_{ij} \equiv f_{i,j}$) and using the fact that $\mathbf{M}^{-1} = \mathbf{M}^t = -\mathbf{M}$ (superscript t denotes transpose), Hamiltonicity is thus equivalent to satisfying the PDEs $\hat{L}_{ij}\mathbf{f} = 0$ where $\hat{L}_{ij}\mathbf{f} = (\mathbf{MJ} + \mathbf{J}^t\mathbf{M})_{ij}$ for all i and j ($\frac{n(n-1)}{2}$ independent ODE's in all), corresponding to $\hat{L}_{ij} = -\mathbf{m}_i^t\partial_j + \mathbf{m}_j^t\partial_i$, where $\mathbf{m}_i$ are the column vectors of $\mathbf{M}$. In other words, although Hamiltonicity is not traditionally thought of as a symmetry, it conveniently meets our generalized symmetry definition and can thus be auto-discovered with our method.

Canonical equivariance: We define a Hamiltonian system as canonically equivariant if $\mathbf{z} = (\mathbf{x}, \mathbf{p})$ and the vector field $\mathbf{f} \equiv (\mathbf{f}_{\mathbf{x}}, \mathbf{f}_{\mathbf{p}})$ satisfies $\mathbf{f}_{\mathbf{x}}(g\mathbf{x}, g^{-1}\mathbf{p}) = g^{-t}\mathbf{f}_{\mathbf{x}}$ and $\mathbf{f}_{\mathbf{p}}(g\mathbf{x}, g^{-1}\mathbf{p}) = g\mathbf{f}_{\mathbf{p}}$ for all $g \in \mathcal{G}$. These two equations are equivalent to the PDEs $\hat{L}_j^{\mathbf{x}}\mathbf{f}_{\mathbf{x}} = 0$ and $\hat{L}_j^{\mathbf{p}}\mathbf{f}_{\mathbf{p}} = 0$ with $\hat{L}_j^{\mathbf{x}} = K_j\mathbf{x}\cdot\nabla_{\mathbf{x}} - K_j^t\mathbf{p}\cdot\nabla_{\mathbf{p}} + K_j^t$ and $\hat{L}_j^{\mathbf{p}} = K_j\mathbf{x}\cdot\nabla_{\mathbf{x}} - K_j^t\mathbf{p}\cdot\nabla_{\mathbf{p}} - K_j$. In special cases when the generator K_j is anti-symmetric (*e.g.*, for the rotation group), $\hat{L}_j^{\mathbf{x}} = \hat{L}_j^{\mathbf{p}}$.

Modularity: A dynamical system $\mathbf{z}(t)$ obeying $\dot{\mathbf{z}} = f(\mathbf{z})$ is *modular* if the Jacobian $\mathbf{J} = \nabla\mathbf{f}$ is block-diagonal, which implies that the components of $\mathbf{z}$ corresponding to different blocks evolve independently of each other. More generally, we say that a system is $(n_1 + \cdots + n_k)$-modular if $\mathbf{J}$ vanishes except for blocks of size $n_1, ...n_k$, which we can write as $\mathbf{A}\circ\mathbf{J} = 0$ where $\circ$ denotes element-wise multiplication ($([\mathbf{A}\circ\mathbf{J}])_{ij} = \mathbf{A}_{ij}\mathbf{J}_{ij}$) and the elements of the mask matrix $\mathbf{A}$ equal 1 inside the blocks, vanishing otherwise. Although modularity is typically not viewed as a symmetry, it too thus meets our generalized symmetry definition and can be auto-discovered with our method using the matrix PDE $\mathbf{A}\circ\nabla\mathbf{f} = 0$, corresponding to the linear operators $\hat{L}_{ij} \equiv \mathbf{A}_{ij}\hat{\mathbf{z}}_i^t\partial_j$.

Our algorithm for discovering hidden symmetries

We now describe our algorithm of discovering hidden symmetries. Since $\hat{L}T = 0$ implies manifest symmetry, $|\hat{L}T|^2$ is a natural measure of manifest symmetry viola-

tion. We therefore define the *symmetry loss* as

$$\ell \equiv \frac{\langle|\hat{L}T(\mathbf{z})|^2\rangle}{\langle|\mathbf{z}|^2\rangle^\alpha}, \tag{2}$$

where angle brackets denote averaging over some set of points $\mathbf{z}_i$, and α is chosen so that ℓ is scale-invariant *i.e.*, invariant under a scale transformation $\mathbf{z} \to a\mathbf{z}$, $T \to a^{m-n}T$, $\hat{L} \to a^s\hat{L}$, $\ell \to a^{2(m-n+s-\alpha)}$ if T has m contravariate indices and n covariate indices. Hence we choose $\alpha = m - n + s$. We jointly search for multiple hidden symmetries by using the loss function $\ell = \sum_i \ell_i$ where each i corresponds to one symmetry, denoted by subscripts as in Tab. I.

Discovering hidden symmetry is equivalent to minimizing ℓ over all diffeomorphisms (everywhere differentiable and invertible coordinate transformations), which we parametrize with an invertible neural network. Figure 2 shows the workflow of our algorithm: (1) a neural network transforms $\mathbf{z} \mapsto \mathbf{z}'$ and obtains the transformation's Jacobian $\mathbf{W} \equiv d\mathbf{z}'/d\mathbf{z}$; (2) in parallel with (1), we evaluate the known tensor field T at $\mathbf{z}$; (3) we jointly feed $\mathbf{W}(\mathbf{z})$ and $T(\mathbf{z})$ into a module which implements the tensor transformation rule and gives $T'(\mathbf{z}')$; (4) we compute the symmetry loss of $\ell(T')$. Note that only the neural network is trainable, while both the tensor field with hidden symmetry $T(\mathbf{z})$ and tensor transformation rule are hard-coded in the workflow. We update the neural network with back-propagation to find the coordinate transformation $\mathbf{z} \mapsto \mathbf{z}'$ that minimizes ℓ. If the resulting ℓ is effectively zero, a hidden symmetry has been discovered.

Neural network training and symbolic regression

We parametrize the coordinate transformation $\mathbf{z} \mapsto \mathbf{z}'$ as $\mathbf{z}' = \mathbf{z} + \mathbf{f}_{\rm NN}(\mathbf{z})$, where $\mathbf{f}_{\rm NN}$ is a fully connected neural network with two hidden layers containing 400 neurons each. We use a silu activation function [16] rather than the popular ReLU alternative, because our method requires activation functions to be twice differentiable (since the loss function involves first derivatives of output with respect to input via the Jacobian $\mathbf{W}$). Derivatives of PDE losses and Jacobians are calculated with automatic differentiation and backpropagation. The invertibility of the mapping $\mathbf{z} \to \mathbf{z}'$ is guaranteed by the fact that $\det \mathbf{W} \to 0$ and the loss function $\ell \to \infty$ if $\mathbf{z} \to \mathbf{z}'$ approaches non-invertibility, as seen in equations (B3)-(B5) in the supplementary material. The supplementary material also provides further technical details on the selection of data points $\mathbf{z}$, neural network initialization and training. If multiple symmetries are tested, the training process is performed in multiple stages: at each stage, we add one more symmetry to the loss function and re-train to convergence.

We then apply AI Feynman, a physics-inspired symbolic regression module, to interpret what the neural network has learned; for details, see Appendix F and [17, 18].

RESULTS

We will now test our algorithm on 6 physics examples, ranging from classical mechanics to general relativity. Table II summarizes these examples, labeled A, B,...,F for easy reference, listing their manifestly non-symmetric equations, their simplifying coordinate transformations, their transformed and manifestly symmetric equations, and their discovered hidden symmetries. As we will see, all the symmetries we had hidden in our test examples were rediscovered by our algorithm. The only example is the transformation for A, where the problem is so simple that an infinite family of transformations give equal symmetry.

Warmup examples

To build intuition for how our method works, we first apply it to the simple warmup examples A, B and C, corresponding to systems involving free motion, harmonic oscillation or Kepler problem whose simplicity has been obfuscated by various coordinate transformations. For our examples, we consider a hidden symmetry to have been tentatively discovered if its corresponding loss drops below a threshold $\epsilon = 10^{-3}$ [2]. If that happens, we apply the AI Feynman symbolic regression package [17, 18] to try to discover a symbolic approximation of the learned transformation $\mathbf{f}_{\rm NN}$ that makes the symmetry loss zero. As can be seen in Tab. II and Fig. 3, all hidden symmetries are successfully discovered together with the coordinate transformations that reveal them. This includes not only traditional hidden symmetries such as translational invariance (example A) and rotational equivariance (example B), but also Hamiltonicity and modularity.

A-C were toy examples in the sense that we had hidden the symmetries by deliberate obfuscation. In contrast, the value of our algorithm lies in its ability to discover symmetries hidden not by people but by nature, as in example D (the linearized double pendulum). We see that our method auto-discovers both hamiltonicity (by finding the correct conjugate momentum variables) and modularity (by auto-discovering the two normal modes),

[2] How low the loss should be to warrant interpretation as a symmetry discovery depends on both training accuracy and data noise; see appendix G for details. For our examples, we found $\epsilon = 10^{-3}$ to be an small enough for symbolic regression to be able to discover the exact formula, after which the symmetry loss drops to exactly zero.

TABLE II: Physical Systems studied

ID	Name	Original dynamics $\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z})$ or metric $g(\mathbf{z})$	Transformation $\mathbf{z} \mapsto \mathbf{z}'$	Symmetric dynamics $\dot{\mathbf{z}}' = \mathbf{f}'(\mathbf{z}')$ or metric $g'(\mathbf{z}')$	Manifest Symmetries
A	1D Uniform Motion	$\frac{d}{dt}\begin{pmatrix}a\\b\end{pmatrix} = \begin{pmatrix}\frac{1}{2}(a+b)\ln(\frac{a-b}{2})\\ \frac{1}{2}(a+b)\ln(\frac{a-b}{2})\end{pmatrix}$	$\begin{pmatrix}a\\b\end{pmatrix} = \begin{pmatrix}e^{\frac{1}{2}x}+e^{\frac{1}{2}p}\\ e^{\frac{1}{2}x}-e^{\frac{1}{2}p}\end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix}x\\p\end{pmatrix} = \begin{pmatrix}p\\0\end{pmatrix}$	Hamiltonicity 1D Translational invariance
B	1D Harmonic Oscillator	$\frac{d}{dt}\begin{pmatrix}a\\b\end{pmatrix} = \begin{pmatrix}(1+a)\ln(1+b)\\ -(1+b)\ln(1+a)\end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix}a\\b\end{pmatrix} = \begin{pmatrix}e^{\frac{1}{2}x}-1\\ e^{\frac{1}{2}p}-1\end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix}x\\p\end{pmatrix} = \begin{pmatrix}p\\-x\end{pmatrix}$	Hamiltonicity SO(2)-equivariance
C	2D Kepler	$\frac{d}{dt}\begin{pmatrix}a\\b\\c\\d\end{pmatrix} = \begin{pmatrix}(1+a)\ln(1+b)\\ -\frac{(1+b)\ln(1+a)}{8(\ln^2(1+a)+\ln^2(1+c))^{3/2}}\\ (1+c)\ln(1+d)\\ -\frac{(1+d)\ln(1+c)}{8(\ln^2(1+a)+\ln^2(1+c))^{3/2}}\end{pmatrix}$	$\begin{pmatrix}a\\b\\c\\d\end{pmatrix} = \begin{pmatrix}e^{\frac{1}{2}x}-1\\ e^{\frac{1}{2}p_x}-1\\ e^{\frac{1}{2}y}-1\\ e^{\frac{1}{2}p_y}-1\end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix}x\\p_x\\y\\p_y\end{pmatrix} = \begin{pmatrix}p_x\\ -x/r^3\\ p_y\\ -y/r^3\end{pmatrix}$	Hamiltonicity Can SO(2)-equivariance
D	Double Pendulum	$\frac{d}{dt}\begin{pmatrix}\theta_1\\ \dot\theta_1\\ \theta_2\\ \dot\theta_2\end{pmatrix} = \begin{pmatrix}\dot\theta_1\\ -\frac{(m_1+m_2)g}{m_1 l}\theta_1 + \frac{m_2 g}{m_1 l}\theta_2\\ \dot\theta_2\\ \frac{(m_1+m_2)g}{m_1 l}\theta_1 - \frac{(m_1+m_2)g}{m_1 l}\theta_2\end{pmatrix}$	$\begin{pmatrix}\theta_1\\ \theta_2\\ \dot\theta_1\\ \dot\theta_2\end{pmatrix} = \begin{pmatrix}-1&1&0&0\\ a&a&0&0\\ 0&0&-1&1\\ 0&0&a&a\end{pmatrix}\begin{pmatrix}\theta_+\\ \theta_-\\ \dot\theta_+\\ \dot\theta_-\end{pmatrix}$ $a = \sqrt{\frac{m_1+m_2}{m_2}}$	$\frac{d}{dt}\begin{pmatrix}\theta_+\\ \dot\theta_+\\ \theta_-\\ \dot\theta_-\end{pmatrix} = \begin{pmatrix}\dot\theta_+\\ -\omega_+^2\theta_+\\ \dot\theta_-\\ -\omega_-^2\theta_-\end{pmatrix}$ $\omega_\pm^2 = \frac{(m_1+m_2)g}{m_1 l}(1 \pm \sqrt{\frac{m_2}{m_1+m_2}})$	Hamiltonicity (2 + 2)-Modularity
E	Expanding universe & empty space	$\mathbf{g} = \begin{pmatrix}1&0&0&0\\ 0&-(r^2+\frac{kx^2}{1-kr^2})\frac{t^2}{r^2}&-\frac{kxyt^2}{1-kr^2}&-\frac{kxzt^2}{1-kr^2}\\ 0&-\frac{kxyt^2}{1-kr^2}&-(r^2+\frac{ky^2}{1-kr^2})\frac{t^2}{r^2}&-\frac{kyzt^2}{1-kr^2}\\ 0&-\frac{kxzt^2}{1-kr^2}&-\frac{kyzt^2}{1-kr^2}&-(r^2+\frac{kz^2}{1-kr^2})\frac{t^2}{r^2}\end{pmatrix}$	$\begin{pmatrix}t'\\ x'\\ y'\\ z'\end{pmatrix} = \begin{pmatrix}t\sqrt{1+r^2}\\ tx\\ ty\\ tz\end{pmatrix}$	$\mathbf{g} = \begin{pmatrix}1&0&0&0\\ 0&-1&0&0\\ 0&0&-1&0\\ 0&0&0&-1\end{pmatrix}$	SO(3,1)-Invariance 4D Translational Invariance
F	Schwarzchild black hole & GP metric	$\mathbf{g} = \begin{pmatrix}1-\frac{2M}{r}&0&0&0\\ 0&-1-\frac{2Mx^2}{(r-2M)r^2}&-\frac{2Mxy}{(r-2M)r^2}&-\frac{2Mxz}{(r-2M)r^2}\\ 0&-\frac{2Mxy}{(r-2M)r^2}&-1-\frac{2My^2}{(r-2M)r^2}&-\frac{2Myz}{(r-2M)r^2}\\ 0&-\frac{2Mxz}{(r-2M)r^2}&-\frac{2Myz}{(r-2M)r^2}&-1-\frac{2Mz^2}{(r-2M)r^2}\end{pmatrix}$	$\begin{pmatrix}t'\\ x'\\ y'\\ z'\end{pmatrix} = \begin{pmatrix}t+2M\left[2u+\ln\frac{u-1}{u+1}\right]\\ x\\ y\\ z\end{pmatrix}$ $u \equiv \sqrt{\frac{r}{2M}}$	$\mathbf{g} = \begin{pmatrix}1-\frac{2M}{r}&-\sqrt{\frac{2M}{r}}\frac{x}{r}&-\sqrt{\frac{2M}{r}}\frac{y}{r}&-\sqrt{\frac{2M}{r}}\frac{z}{r}\\ -\sqrt{\frac{2M}{r}}\frac{x}{r}&-1&0&0\\ -\sqrt{\frac{2M}{r}}\frac{y}{r}&0&-1&0\\ -\sqrt{\frac{2M}{r}}\frac{z}{r}&0&0&-1\end{pmatrix}$	SO(3)-Invariance 3D Translational Invariance

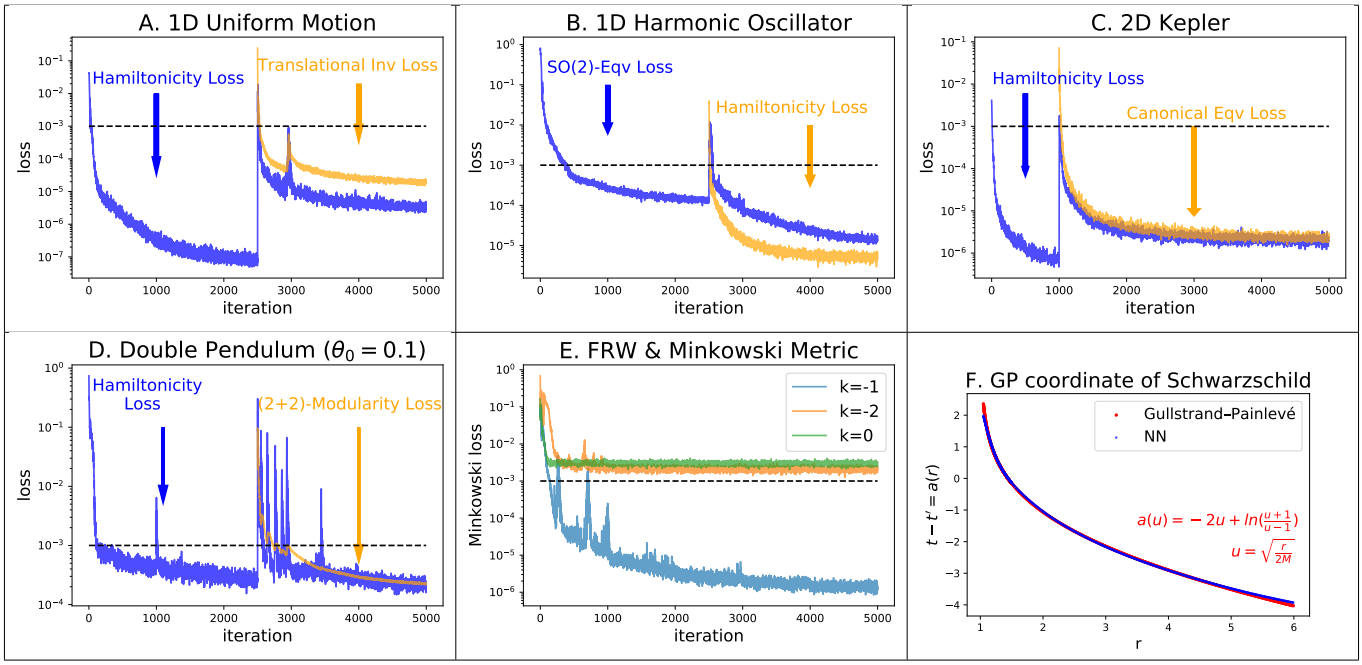

FIG. 3: All hidden symmetries in six tested systems are discovered by our algorithm. The last figure shows that the neural network accurately learns the Gullstrand-Painlevé transformation.

even though neither of these symmetries were manifest in the most obvious physical coordinates (the pendulum angles and angular velocities) [19].

General relativity examples

As a first general relativity (GR) application (example E), we consider the Friedmann-Robertson-Walker (FRW) metric[3] describing a homogeneous and isotropic expanding universe with negative spatial curvature ($k = 1$) and cosmic scale factor evolution $a(t) = t$. A GR expert will realize that its Riemann tensor vanishes everywhere, so that there must exist a coordinate transformation revealing this to be simply empty space in disguise, with Poincaré symmetry (Lorenz symmetry and 4D translational symmetry). Discovering this transformation is

[3] We use $r \equiv \sqrt{x^2+y^2+z^2}$ for brevity in Tab. II, but not in the neural network, which actually takes (t, x, y, z) as inputs.

non-trivial, and is sometimes assigned as a homework problem in graduate courses.

It is easy to show that any metric with Poincaré symmetry must be a multiple of the Minkowski metric η, so we define our Poincaré symmetry loss as $\ell \equiv \langle ||T(\mathbf{z}) - \eta||^2 \rangle / \langle ||T(\mathbf{z})||^2 \rangle$. Figure 3 (E) shows that the Minkowski loss drops below 10^{-3}, indicating that the $k = -1$ FRW metric is indeed homomorphic to Minkowski space, while the loss gets stuck above 10^{-3} for $k = -2$ and $k = 0$. Applying the AI Feynman symbolic regression package [17] to the learned transformation $\mathbf{f}_{\rm NN} = (t, x, y, z)$ reveals the exact formula $(x', y', z', t') = (tx, ty, tz, t\sqrt{1 + x^2 + y^2 + z^2})$, which gives vanishing loss.

We now turn to studying the spacetime of a non-rotating black hole described by the Schwarzschild metric (without loss of generality, we set $2M = 1$). This problem proved so difficult that it took physicists 17 years to clear up the misconception that something singular occurs at the event horizon, until it was finally revealed that the apparent singularity at $r = 2M$ was merely caused by a poor choice of coordinates [20–23], just as the z-axis is merely a coordinate singularity in spherical coordinates. Our method auto-discovers hidden translational symmetry in the spatial coordinates (x, y, z), revealed by the coordinate transformation $t' = t + 2M\left[2u + \ln\frac{u-1}{u+1}\right]$, where $u \equiv \sqrt{r/2M}$, which is auto-discovered by applying AI Feynman [17] to the learned transformation $\mathbf{f}_{\rm NN}$ (see Fig. 3, panel F). Since both the original and target metrics have the SO(3) (rotational) spatial symmetry, our neural network parametrizes the coordinate transformation $(x, y, z, t) \to (x', y', z', t')$ via a two-dimensional transformation $(r, t) \to (r', t')$, where $r \equiv \sqrt{x^2 + y^2 + z^2}$. This transforms the Schwarzschild metric into the famous Gullstrand-Painlevé metric [22, 23], which is seen to be perfectly regular at the event horizon and can be intuitively interpreted simply as flat space flowing inward with the classical escape velocity [11, 24].

CONCLUSIONS

We have presented a machine-learning algorithm for auto-discovering hidden symmetries, and shown it to be effective for a series of examples from classical mechanics and general relativity. Our symmetry definition is very broad, corresponding to the data satisfying a differential equation, which encompasses both traditional invariance and equivariance as well as Hamiltonicity and modularity.

Our work is linked to Noether's theorem [25], which states that a continuous symmetry leaving the Lagrangian invariant corresponds to a conservation law. The Lagrangian is a scalar, a special case of the tensors of this paper. If we rewrite the dynamical equations in the form of Euler-Lagrange equations, then the invariance of the Lagrangian under a symmetry group is equivalent to the equivariance of the dynamical equation under the same symmetry group, both of which imply the same conservation laws.

In future work, it will be interesting to seek hidden symmetries in data from both experiments and numerical simulations. Although our examples involved no more than two symmetries at once, it is straightforward to auto-search for a whole library of common symmetries, adopting the best-fitting one and recursively searching for more hidden symmetries until all are found.

Currently, our method can only search for symmetries from a list of candidates pre-specified by the user, and cannot search for unknown symmetries. In future work, it will be interesting to enable search also for unknown symmetries, *e.g.*, by making the Lie generators trainable. In other words, if there is *any* differential equation that a suitably transformed dataset satisfies, one would seek to auto-discover both the transformation and the differential equation.

Acknowledgement We thank the Center for Brains, Minds, and Machines (CBMM) for hospitality. This work was supported by The Casey and Family Foundation, the Foundational Questions Institute, the Rothberg Family Fund for Cognitive Science and IAIFI through NSF grant PHY-2019786.

[1] P. W. Anderson, More is different, Science **177**, 393 (1972), https://science.sciencemag.org/content/177/4047/393.full.pdf.
[2] D. J. Gross, Symmetry in physics: Wigner's legacy, Physics Today **48**, 46 (1995).
[3] D. J. Gross and F. Wilczek, Asymptotically free gauge theories. i, Physical Review D **8**, 3633 (1973).
[4] T. Cohen and M. Welling, Group equivariant convolutional networks, in *International conference on machine learning* (PMLR, 2016) pp. 2990–2999.
[5] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds, arXiv preprint arXiv:1802.08219 (2018).
[6] F. B. Fuchs, D. E. Worrall, V. Fischer, and M. Welling, Se (3)-transformers: 3d roto-translation equivariant attention networks, arXiv preprint arXiv:2006.10503 (2020).
[7] R. Kondor, Z. Lin, and S. Trivedi, Clebsch-gordan nets: a fully fourier space spherical convolutional neural network, arXiv preprint arXiv:1806.09231 (2018).
[8] V. G. Satorras, E. Hoogeboom, and M. Welling, E (n) equivariant graph neural networks, arXiv preprint arXiv:2102.09844 (2021).
[9] G. Kanwar, M. S. Albergo, D. Boyda, K. Cranmer, D. C. Hackett, S. Racanière, D. J. Rezende, and P. E. Shanahan, Equivariant flow-based sampling for lattice gauge theory, Phys. Rev. Lett. **125**, 121601 (2020).
[10] D. Boyda, G. Kanwar, S. Racanière, D. J. Rezende, M. S. Albergo, K. Cranmer, D. C. Hackett, and P. E. Shana-

han, Sampling using SU(n) gauge equivariant flows, Phys. Rev. D **103**, 074504 (2021).
[11] C. Misner, K. Thorne, and J. Wheeler, *Gravitation* (Princeton University Press, 2017).
[12] M. Raissi, P. Perdikaris, and G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics **378**, 686 (2019).
[13] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, and P. Gentine, Enforcing analytic constraints in neural networks emulating physical systems, Phys. Rev. Lett. **126**, 098302 (2021).
[14] M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, Lagrangian neural networks, arXiv preprint arXiv:2003.04630 (2020).
[15] Z. Liu, B. Wang, Q. Meng, W. Chen, M. Tegmark, and T.-Y. Liu, Machine-learning nonconservative dynamics for new-physics detection, Phys. Rev. E **104**, 055302 (2021).
[16] S. Elfwing, E. Uchibe, and K. Doya, Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, Neural Networks **107**, 3 (2018), special issue on deep reinforcement learning.
[17] S.-M. Udrescu, A. Tan, J. Feng, O. Neto, T. Wu, and M. Tegmark, Ai feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity (2020), arXiv:2006.10782 [cs.LG].
[18] S.-M. Udrescu and M. Tegmark, Ai feynman: A physics-inspired method for symbolic regression, Science Advances **6**, 10.1126/sciadv.aay2631 (2020).
[19] H. Goldstein, C. Poole, and J. Safko, Classical mechanics (2002).
[20] R. Penrose, Gravitational collapse and space-time singularities, Phys. Rev. Lett. **14**, 57 (1965).
[21] M. D. Kruskal, Maximal extension of schwarzschild metric, Phys. Rev. **119**, 1743 (1960).
[22] P. Painlevé, La mécanique classique et la théorie de la relativité, C. R. Acad. Sci. (Paris) **173**, 677 (1921).
[23] A. Gullstrand, Allgemeine lösung des statischen einkörrperproblems in der einsteinschen gravitationstheorie, Arkiv för Matematik, Astronomi och Fysik **16 (8)**, 1 (1922).
[24] A. J. S. Hamilton and J. P. Lisle, The river model of black holes, American Journal of Physics **76**, 519–532 (2008).
[25] E. Noether, Invariante variationsprobleme, Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse **1918**, 235 (1918).

Supplementary material

Appendix A: Neural network training details

Preparing data Our default method for generating training data is to draw the sample points of $\mathbf{z}$ as i.i.d. normalized Gaussians, *i.e.*, $\mathbf{z} \sim N(0, \mathbf{I}_N)$. However, there are three issues to note: (1) For the double pendulum, since we focus on the small angle regime, we instead use the narrowed the Gaussian distribution $(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) \sim N(0, 0.1^2)$. (2) To avoid the $r = 2M = 1$ singularity of the Schwarzchild metric, we draw radius its r from a uniform distribution $U(1.1, 6)$ (F); (3) For the two general relativity examples E and F, the time variable is sampled from a uniform distribution $U[0, 3]$.

The ADAM optimizer [**?**] is employed to train the neural network. The output $\mathbf{z}'$ is computed as $\mathbf{z}' = \mathbf{z} + \mathbf{f}_{\rm NN}(\mathbf{z})$ rather than $\mathbf{z}' = \mathbf{f}_{\rm NN}(\mathbf{z})$ to ensure $\mathbf{W} \equiv \frac{\partial \mathbf{z}'}{\partial \mathbf{z}} \approx \mathbf{I}$ at initialization, so that $\mathbf{W}^{-1}$ is well-conditioned and avoids training instabilities. For examples A-D, we train the neural network for 2,000 epochs with learning rate 10^{-3}; For examples E and F, we train for 1,000 epochs three times while annealing the learning rate as $\{5 \times 10^{-3}, 10^{-3}, 2 \times 10^{-4}\}$.

Appendix B: Tensor transformation rules

Although tensor transformation rules are well-known, we list the relevant ones here for the convenience of any reader wishing to implement our method. We consider a coordinate vector $\mathbf{z} \in \mathbb{R}^N$ and a tensor field $T(\mathbf{z})$. Under the coordinate transformation $\mathbf{z} \to \mathbf{z}'$, the transformation rule for the tensor field $T(\mathbf{z}) \to T'(\mathbf{z}')$ depends on the type of T. In general relativity, a contravariant (1,0) vector v^i and a covariant (0,1) vector v_i transforms as

$$\begin{aligned} v'^i &= \frac{\partial z'^i}{\partial z^j} v^j = \mathbf{W}^i{}_j v^j, \\ v'_i &= \frac{\partial z^j}{\partial z'^i} v_j = (\mathbf{W}^{-1})^j{}_i v_j = (\mathbf{W}^{-T})_i{}^j v_j, \end{aligned} \tag{B1}$$

where $\mathbf{W}^i{}_j \equiv \frac{\partial z'^i}{\partial z^j}$. More generally, an (m, n) tensor $T^{i_1 \cdots i_m}_{j_1 \cdots j_n}$ transforms as

$$T'^{i'_1 \cdots i'_m}_{j'_1 \cdots j'_n} = \mathbf{W}^{i'_1}_{i_1} \cdots \mathbf{W}^{i'_m}_{i_m} (\mathbf{W}^{-1})^{j_1}_{j'_1} \cdots (\mathbf{W}^{-1})^{j_n}_{j'_n} T^{i_1 \cdots i_m}_{j_1 \cdots j_n} \tag{B2}$$

In this paper, we are interested in these specific cases:

- $(1, 0)$ vector: The transformation rule is $\mathbf{f} \to \mathbf{f}' = \mathbf{W}\mathbf{f}$, and the dynamical system $\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z})$ where $\dot{\mathbf{z}} \equiv \frac{d\mathbf{z}}{dt}$ falls in this category.
- $(0, 2)$ tensor: The transformation rule is $\mathbf{g} \to \mathbf{g}' = \mathbf{W}^{-T}\mathbf{g}\mathbf{W}^{-1}$, and the metric tensor $g_{\mu\nu}$ from General Relativity lies in this category.

We define the first-order differentiation of T wrt $\mathbf{z}$ as $\mathbf{J} \equiv \nabla T$ or $J^{i_1 \cdots i_m}_{j_1 \cdots j_n l} \equiv \partial_l T^{i_1 \cdots i_m}_{j_1 \cdots j_n}$. Written as such, $\mathbf{J}$ is not a $(m, n+1)$ tensor because its transformation rule is

$$\begin{aligned} J'^{i'_1 \cdots i'_m}_{j'_1 \cdots j'_n l'} &\equiv \partial'_{l'} T'^{i'_1 \cdots i'_m}_{j'_1 \cdots j'_n} = (\mathbf{W}^{-T})_{l'}{}^l \partial_l (\mathbf{W}^{i'_1}_{i_1} \cdots \mathbf{W}^{i'_m}_{i_m} (\mathbf{W}^{-1})^{j_1}_{j'_1} \cdots (\mathbf{W}^{-1})^{j_n}_{j'_n} T^{i_1 \cdots i_m}_{j_1 \cdots j_n}) \\ &= \sum_{a=1}^{m} (\mathbf{W}^{-T})_{l'}{}^l (\partial_l \mathbf{W}^{i'_a}_{i_a}) (\prod_{b=1, b\neq a}^{m} \mathbf{W}^{i'_b}_{i_b}) (\prod_{c=1}^{n} (\mathbf{W}^{-1})^{j_c}_{j'_c}) T^{i_1 \cdots i_m}_{j_1 \cdots j_n} \\ &+ \sum_{a=1}^{n} (\mathbf{W}^{-T})_{l'}{}^l (\prod_{b=1}^{m} \mathbf{W}^{i'_b}_{i_b}) (\partial_l (\mathbf{W}^{-1})^{j_a}_{j'_a}) (\prod_{c=1, c\neq a}^{n} (\mathbf{W}^{-1})^{j_c}_{j'_c}) T^{i_1 \cdots i_m}_{j_1 \cdots j_n} \\ &+ (\mathbf{W}^{-T})_{l'}{}^l \mathbf{W}^{i'_1}_{i_1} \cdots \mathbf{W}^{i'_m}_{i_m} (\mathbf{W}^{-1})^{j_1}_{j'_1} \cdots (\mathbf{W}^{-1})^{j_n}_{j'_n} \partial_l T^{i_1 \cdots i_m}_{j_1 \cdots j_n} \end{aligned} \tag{B3}$$

$(1, 0)$ vector $\mathbf{f}$:

$$\begin{aligned}\mathbf{J}'^{i}_{\ j} &\equiv \partial'_j \mathbf{f}'^i \\ &= (\mathbf{W}^{-T})_j^{\ k} \partial_k (\mathbf{W}^i_{\ l} \mathbf{f}^l) \\ &= (\mathbf{W}^{-T})^k_{\ j} (\partial_k \mathbf{W}^i_{\ l}) \mathbf{f}^l + (\mathbf{T}^{-T})_j^{\ k} \mathbf{W}^i_{\ l} \partial_k \mathbf{f}^l\end{aligned} \tag{B4}$$

$(0,2)$ tensor $\mathbf{g}$:

$$\begin{aligned}\mathbf{J}'_{ijk} &\equiv \partial'_k g'_{ij} \\ &= (\mathbf{W}^{-T})_k^{\ l} \partial_l ((\mathbf{W}^{-T})_i^{\ m} g_{mn} (\mathbf{W}^{-1})^n_{\ j}) \\ &= (\mathbf{W}^{-T})_k^{\ l} \partial_l (\mathbf{W}^{-T})_i^{\ m} g_{mn} (\mathbf{W}^{-1})^n_{\ j} + (\mathbf{W}^{-T})_k^{\ l} (\mathbf{W}^{-T})_i^{\ m} \partial_l g_{mn} (\mathbf{W}^{-1})^n_{\ j} + (\mathbf{W}^{-T})_k^{\ l} (\mathbf{W}^{-T})_i^{\ m} g_{mn} \partial_l (\mathbf{W}^{-1})^n_{\ j}\end{aligned} \tag{B5}$$

Appendix C: Deriving PDEs forr generalized symmetry

Here we present more detailed derivations of the partial differential equations (PDEs) corresponding to Hamiltonicity, Lie equivariance, Lie invariance and canonical equivariance. The PDEs for translational invariance and modularity are obvious given the definitions.

1. Hamiltonicity

Hamiltonian systems (a.k.a. *symplectic systems*) can be written with the state $\mathbf{z} \in \mathbb{R}^{2d}$ as the concatenation of coordinates $\mathbf{x} \in \mathbb{R}^d$ and momenta $\mathbf{p} \in \mathbb{R}^d$. The Hamiltonian $H(\mathbf{z})$ is a conserved quantity and governs the system evolution as $(\dot{\mathbf{x}}, \dot{\mathbf{p}}) = (\partial_{\mathbf{p}} H, -\partial_{\mathbf{x}} H)$, or more compactly, $\dot{\mathbf{z}} = \mathbf{M}\nabla H$ where $\mathbf{M}$ is the (2,0) symplectic matrix $((\mathbf{0}, \mathbf{I}_d), (-\mathbf{I}_d, \mathbf{0}))$. $\mathbf{f} \equiv \mathbf{M}\nabla H$ is a (1,0) vector. $\mathbf{M}$ satisfies $\mathbf{M}^{-1} = \mathbf{M}^T = -\mathbf{M} = -\mathbf{M}^{-T}$. We observe that

$$\mathbf{J}_{ij} = \partial_j f^i = \partial_j (\mathbf{M}\nabla H)^i = \partial_j (\mathbf{M}^{ik} \partial_k H) = \mathbf{M}^{ik} \partial_j \partial_k H \tag{C1}$$

Left multiplying by $(\mathbf{M}^{-1})_{mi}$ on both sides and utilizing $(\mathbf{M}^{-1})_{mi} \mathbf{M}^{ik} = \delta_m^{\ k}$ gives

$$(\mathbf{M}^{-1}\mathbf{J})_{mj} = \delta_m^{\ k} \partial_j \partial_k H = \partial_j \partial_m H. \tag{C2}$$

Since the right hand side $\partial_j \partial_m H = \partial_m \partial_j H$ is symmetric, $\mathbf{M}^{-1}\mathbf{J}$ is symmetric too, i.e.,

$$(\mathbf{M}^{-1}\mathbf{J})^T = \mathbf{M}^{-1}\mathbf{J} \Longrightarrow \mathbf{J}^T\mathbf{M} + \mathbf{M}\mathbf{J} = 0 \tag{C3}$$

Note that the converse is also true: if $\mathbf{MJ}$ is symmetric, then there exists a scalar field H such that $\mathbf{f} = \mathbf{M}\nabla H$. We first introduce a lemma here:

Lemma 1. *A vector field $\boldsymbol{f}$ can be written as the gradient of a scalar field ϕ, i.e. $\boldsymbol{f} = \nabla\phi$, if and only if $\mathbf{J} \equiv \nabla \boldsymbol{f}$ is symmetric.*

Proof. $\Longrightarrow$: $\mathbf{J} \equiv \nabla\mathbf{f} = \nabla\nabla\phi$, i.e. the Hessian (the matirx of second derivatives) of ϕ, which is symmetric.
$\Longleftarrow$: $\mathbf{J}_{ij} = \mathbf{J}_{ji}$ means that $\partial_i \mathbf{f}_j = \partial_j \mathbf{f}_i$. Consider the loop integral

$$\oint \mathbf{f} \cdot d\mathbf{z} = \frac{1}{2} \sum_i \sum_j \iint (\partial_i f_j - \partial_j f_i) d\mathbf{z}_i d\mathbf{z}_j = 0 \tag{C4}$$

The first equality is due to Green's theorem. This Vanishing loop integral is equivalent to saying $\mathbf{f}$ is a gradient field [**?**]. □

Using this lemma, the fact that $\mathbf{MJ} \equiv \mathbf{M}\nabla\mathbf{f} = \nabla(\mathbf{Mf})$ is symmetric is equivalent to saying there exists a scalar function H such that $\mathbf{Mf} = -\nabla H \Leftrightarrow \mathbf{f} = \mathbf{M}\nabla H$.

2. Lie Equivariance

By definition, a vector field $\mathbf{f}(\mathbf{z})$ is equivariant under a Lie group $\mathcal{G}$ if $\mathbf{f}(g\mathbf{z}) = g\mathbf{f}(\mathbf{z})$ for every group element $g \in \mathcal{G}$. The element g can be expressed in terms of generators K_i $(i = 1, \cdots, d)$ such that $g = \exp(\theta_i K_i)$. When θ_i are small, $g \approx I + \theta_i K_i$, so the following equalities hold to first order in a Taylor expansion in θ:

$$\begin{aligned} \mathbf{f}(g\mathbf{z}) &= \mathbf{f}(\mathbf{z} + \theta_i K_i \mathbf{z}) = \mathbf{f}(\mathbf{z}) + \theta_i \nabla \mathbf{f}(\mathbf{z}) K_i \mathbf{z} \\ g\mathbf{f}(\mathbf{z}) &= \mathbf{f}(\mathbf{z}) + \theta_i K_i \mathbf{f}(\mathbf{z}) \\ \mathbf{f}(g\mathbf{z}) - g\mathbf{f}(\mathbf{z}) &= \theta_i (\nabla \mathbf{f}(\mathbf{z}) K_i \mathbf{r} - K_i \mathbf{f}(\mathbf{z})) \\ \frac{\partial(\mathbf{f}(g\mathbf{z}) - g\mathbf{f}(\mathbf{z}))}{\partial \theta_i} &= \nabla \mathbf{f}(\mathbf{z}) K_i \mathbf{z} - K_i \mathbf{f}(\mathbf{z}) = \mathbf{J}(\mathbf{z}) K_i \mathbf{z} - K_i \mathbf{f}(\mathbf{z}) \end{aligned} \tag{C5}$$

Hence the equivariance PDE and loss are

$$\begin{aligned} &\hat{L}_i \mathbf{f} = \mathbf{J} K_i \mathbf{z} - K_i \mathbf{f} = \mathbf{0}, \\ &\ell_{\text{eqv}} = \frac{1}{N_g} \sum_{j=1}^{N_g} \sum_{i=1}^{N_s} (||\mathbf{J}(\mathbf{z}^{(i)}) K_j \mathbf{z}^{(i)} - K_j \mathbf{f}(\mathbf{z}^{(i)})||_2^2) / (\sum_{i=1}^{N_s} ||\mathbf{z}^{(i)}||_2^2), \end{aligned} \tag{C6}$$

where the aim of the denominator $||\mathbf{f}||_2^2$ is to make the loss function dimensionless, i.e., a (0,0) tensor. Eq. (C6) holds for all Lie groups – one just needs to insert generators K_i to obtain the corresponding loss for any Lie group. For the Lorentz group $SO(3,1)$, for example, there are $N_g = 6$ Lie generators in total, where first three correspond to boosting, and last three to spatial rotations.

$$\begin{aligned} K_1 &= \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, K_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, K_3 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \\ K_4 &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, K_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}, K_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \end{aligned} \tag{C7}$$

3. Lie Invariance

Following the notation above, for any (m, n)-tensor field T:

$$T(g\mathbf{z}) = T(\mathbf{z}) + \theta_i \nabla T(\mathbf{z}) K_i \mathbf{z} \tag{C8}$$

Lie invariance requests that

$$T(g\mathbf{z}) = T(\mathbf{z}) \Longrightarrow \nabla T(\mathbf{z}) K_i \mathbf{z} = 0 \ \forall i \tag{C9}$$

Similarly we can construct the loss function as

$$\ell_{\text{inv}} = \frac{1}{N_g} \sum_{j=1}^{N_g} \sum_{i=1}^{N_s} (||\mathbf{J}(\mathbf{z}^{(i)}) K_j \mathbf{z}^{(i)}||_F^2) / (\sum_{i=1}^{N_s} ||\mathbf{z}^{(i)}||_2^2). \tag{C10}$$

Here $\mathbf{J} = \nabla T$ is reshaped as a matrix of size $N^{m+n} \times N$ where the l-th column is the vectorization of $\partial_l T^{i_1 \cdots i_m}_{j_1 \cdots j_n}$. Note that translational invariance can also be included in this framework by setting $K_j = \partial_j$.

4. Canonical Equivariance

Canonical equivariance requires that

$$\begin{aligned} &\mathbf{f}_{\mathbf{x}}(g\mathbf{x}, g^{-1}\mathbf{p}) = g^{-1} \mathbf{f}_{\mathbf{x}}(\mathbf{x}, \mathbf{p}), \\ &\mathbf{f}_{\mathbf{p}}(g\mathbf{x}, g^{-1}\mathbf{p}) = g \mathbf{f}_{\mathbf{x}}(\mathbf{x}, \mathbf{p}), \end{aligned} \tag{C11}$$

for all $g \in \mathcal{G}$. For a Lie group $\mathcal{G}$, $g = \exp(\theta_j K_j)$, where K_j are generators. Note that $g^{-t} = \exp(-\theta_j K_j^t)$. Differentiating with respect to θ_j in Eq. (C11) gives

$$\begin{aligned} K_j \mathbf{x} \cdot \nabla_{\mathbf{x}} \mathbf{f}_{\mathbf{x}} - K_j^t \mathbf{p} \cdot \nabla_{\mathbf{p}} \mathbf{f}_{\mathbf{x}} &= -K_j^t \mathbf{f}_{\mathbf{x}}, \\ K_j \mathbf{x} \cdot \nabla_{\mathbf{x}} \mathbf{f}_{\mathbf{p}} - K_j^t \mathbf{p} \cdot \nabla_{\mathbf{p}} \mathbf{f}_{\mathbf{p}} &= K_j \mathbf{f}_{\mathbf{p}} \end{aligned} \tag{C12}$$

Appendix D: Physical systems

For convenience, we here review the well-known equations of the physical systems we test our method on in the paper.

1. Double pendulum

The dynamics of a double pendulum can be described by two angles (θ_1, θ_2) and corresponding angular velocities $(\dot{\theta}_1, \dot{\theta}_2)$:

$$\frac{d}{dt}\begin{pmatrix}\theta_1\\ \theta_2\\ \dot{\theta}_1\\ \dot{\theta}_2\end{pmatrix} = \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \frac{m_2 l_1 \dot{\theta}_1^2 \sin(\theta_2-\theta_1)\cos(\theta_2-\theta_1) + m_2 g \sin\theta_2 + m_2 l_2 \dot{\theta}_2^2 \sin(\theta_2-\theta_1) - (m_1+m_2) g \sin\theta_1}{(m_1+m_2) l_1 - m_2 l_1 \cos^2(\theta_2-\theta_1)} \\ \frac{-m_2 l_2 \dot{\theta}_2^2 \sin(\theta_2-\theta_1) + (m_1+m_2)(g\sin\theta_1\cos(\theta_2-\theta_1) - l_1 \dot{\theta}_1^2 \sin(\theta_2-\theta_1) - g\sin\theta_2)}{(m_1+m_2) l_1 - m_2 l_1 \cos^2(\theta_2-\theta_1)} \end{pmatrix} \tag{D1}$$

The canonical coordinates corresponding to $(\dot{\theta}_1, \dot{\theta}_2)$ are (p_1, p_2), given by

$$\begin{aligned} p_1 &= (m_1+m_2) l_1^2 \dot{\theta}_1 + m_2 l_1 l_2 \dot{\theta}_2 \cos(\theta_1-\theta_2), \\ p_2 &= m_2 l_2^2 \dot{\theta}_2 + m_2 l_1 l_2 \dot{\theta}_1 \cos(\theta_1-\theta_2). \end{aligned} \tag{D2}$$

The dynamical equations (B1) can be rewritten in terms of $(\theta_1, \theta_2, p_1, p_2)$:

$$\frac{d}{dt}\begin{pmatrix}\theta_1\\ \theta_2\\ p_1\\ p_2\end{pmatrix} = \begin{pmatrix} \frac{l_2 p_1 - l_1 p_2 \cos(\theta_1-\theta_2)}{l_1^2 l_2 (m_1 + m_2 \sin^2(\theta_1-\theta_2))} \\ \frac{l_1 (m_1+m_2) p_2 - l_1 m_2 p_1 \cos(\theta_1-\theta_2)}{l_1 l_2^2 m_2 (m_1 + m_2 \sin^2(\theta_1-\theta_2))} \\ -(m_1+m_2) g l_1 \sin\theta_1 - C_1 + C_2 \\ -m_2 g l_2 \sin\theta_2 + C_1 - C_2 \end{pmatrix}, \text{ where } \begin{pmatrix}C_1\\ C_2\end{pmatrix} = \begin{pmatrix} \frac{p_1 p_2 \sin(\theta_1-\theta_2)}{l_1 l_2 (m_1 + m_2 \sin^2(\theta_1-\theta_2))} \\ \frac{l_2^2 m_2 p_1^2 + l_1^2 (m_1+m_2) p_2^2 - l_1 l_2 m_2 p_1 p_2 \cos(\theta_1-\theta_2)}{2 l_1^2 l_2^2 (m_1 + m_2 \sin^2(\theta_1-\theta_2))^2} \end{pmatrix} \tag{D3}$$

In the small angle regime $\theta_1, \theta_2 \ll 1$, after using the approximations $\sin\theta_1 \approx \theta_1$, $\sin\theta_2 \approx \theta_2$, $\cos(\theta_1-\theta_2) \approx 1$, $\dot{\theta}_1^2 \approx 0$, $\dot{\theta}_2^2 \approx 0$ and setting $l_1 = l_2 = l$, (E1)-(E3) simplify to

$$\frac{d}{dt}\begin{pmatrix}\theta_1\\ \theta_2\\ \dot{\theta}_1\\ \dot{\theta}_2\end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{(m_1+m_2)g}{m_1 l} & \frac{m_2 g}{m_1 l} & 0 & 0 \\ \frac{(m_1+m_2)g}{m_1 l} & -\frac{(m_1+m_2)g}{m_1 l} & 0 & 0 \end{pmatrix} \begin{pmatrix}\theta_1\\ \theta_2\\ \dot{\theta}_1\\ \dot{\theta}_2\end{pmatrix}, \tag{D4}$$

$$\begin{aligned} p_1 &= (m_1+m_2) l^2 \dot{\theta}_1 + m_2 l^2 \dot{\theta}_2, \\ p_2 &= m_2 l^2 \dot{\theta}_2 + m_2 l^2 \dot{\theta}_1, \end{aligned} \tag{D5}$$

$$\frac{d}{dt}\begin{pmatrix}\theta_1\\ \theta_2\\ p_1\\ p_2\end{pmatrix} = \begin{pmatrix} 0 & 0 & \frac{1}{m_1 l^2} & -\frac{1}{m_1 l^2} \\ 0 & 0 & -\frac{1}{m_1 l^2} & \frac{m_1+m_2}{l^2 m_2 m_1} \\ -(m_1+m_2) g l & 0 & 0 & 0 \\ 0 & -m_2 g l & 0 & 0 \end{pmatrix} \begin{pmatrix}\theta_1\\ \theta_2\\ p_1\\ p_2\end{pmatrix}, \tag{D6}$$

$$\begin{pmatrix}\theta_1\\ \dot{\theta}_1\\ \theta_2\\ \dot{\theta}_2\end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{m_1 l^2} & 0 & -\frac{1}{m_1 l^2} \\ 0 & 0 & 1 & 0 \\ 0 & -\frac{1}{m_1 l^2} & 0 & \frac{m_1+m_2}{m_1 m_2 l^2} \end{pmatrix} \begin{pmatrix}x_1\\ p_1\\ x_2\\ p_2\end{pmatrix}, \tag{D7}$$

From Eq. (E4), we can write θ_1 and θ_2 as linear combinatitons of two **normal modes** $\theta_\pm(t)$ that oscillate independently with frequencies $\omega_\pm$:

$$\begin{aligned} \omega_\pm^2 &= \frac{g}{m_1 l}\left[m_1+m_2 \pm \sqrt{(m_1+m_2)m_2}\right], \\ \theta_1 = \theta_+ - \theta_-, &\quad \theta_2 = -\sqrt{\frac{m_1+m_2}{m_2}}(\theta_+ + \theta_-) \end{aligned} \tag{D8}$$

2. Friedmann–Robertson–Walker (FRW) metric

The Friedmann–Robertson–Walker (FRW) metric in the spherical coordinate and in the cartesian coordiante are:

$$\begin{aligned} \mathbf{g}^{\text{FRW}}_{\text{spherical}}(t,r,\theta,\phi) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -\frac{a(t)^2}{1-kr^2} & 0 & 0 \\ 0 & 0 & -a(t)^2 r^2 & 0 \\ 0 & 0 & 0 & -a(t)^2 r^2 \sin^2\theta \end{pmatrix}, \\ \mathbf{g}^{\text{FRW}}_{\text{cartesian}}(t,x,y,z) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -(r^2+\frac{kx^2}{1-kr^2})\frac{a(t)^2}{r^2} & -\frac{kxya(t)^2}{1-kr^2} & -\frac{kxza(t)^2}{1-kr^2} \\ 0 & -\frac{kxya(t)^2}{1-kr^2} & -(r^2+\frac{ky^2}{1-kr^2})\frac{a(t)^2}{r^2} & -\frac{kyza(t)^2}{1-kr^2} \\ 0 & -\frac{kxza(t)^2}{1-kr^2} & -\frac{kyza(t)^2}{1-kr^2} & -(r^2+\frac{kz^2}{1-kr^2})\frac{a(t)^2}{r^2} \end{pmatrix}. \end{aligned} \tag{D9}$$

When $k=1$ and $a(t)=t$, the FRW metric can be transformed to the Minkowski metric via a global transformation:

$$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} t\sqrt{1+r^2} \\ tx \\ ty \\ tz \end{pmatrix} \tag{D10}$$

3. Schwarzschild metric and GP coordinate

The Schwarzschild metric in spherical coordinates and in Cartesian coordinates are

$$\begin{aligned} \mathbf{g}^{\text{Sch}}_{\text{spherical}}(t,r,\theta,\phi) &= \begin{pmatrix} 1-\frac{2M}{r} & 0 & 0 & 0 \\ 0 & -(1-\frac{2M}{r})^{-1} & 0 & 0 \\ 0 & 0 & -r^2 & 0 \\ 0 & 0 & 0 & -r^2\sin^2\theta \end{pmatrix}, \\ \mathbf{g}^{\text{Sch}}_{\text{cartesian}}(t,x,y,z) &= \begin{pmatrix} 1-\frac{2M}{r} & 0 & 0 & 0 \\ 0 & -1-\frac{2Mx^2}{(r-2M)r^2} & -\frac{2Mxy}{(r-2M)r^2} & -\frac{2Mxz}{(r-2M)r^2} \\ 0 & -\frac{2Mxy}{(r-2M)r^2} & -1-\frac{2My^2}{(r-2M)r^2} & -\frac{2Myz}{(r-2M)r^2} \\ 0 & -\frac{2Mxz}{(r-2M)r^2} & -\frac{2Myz}{(r-2M)r^2} & -1-\frac{2Mz^2}{(r-2M)r^2} \end{pmatrix}. \end{aligned} \tag{D11}$$

The spatial part of the metric diverges at $r=2M$. However, if we transform to the Gullstrand–Painlevé coordinate defined by

$$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} t-2M(-2u+\ln\frac{u+1}{u-1})) \\ x \\ y \\ z \end{pmatrix}, u=\sqrt{\frac{r}{2M}}, \tag{D12}$$

the apparent singularity disappears, and we obtain a spatial metric which, remarkably, is the same as for Euclidean space:

$$\mathbf{g}_{\rm GP}^{\rm Sch} = \begin{pmatrix} 1-\frac{2M}{r} & -\sqrt{\frac{2M}{r}}\frac{x}{r} & -\sqrt{\frac{2M}{r}}\frac{y}{r} & -\sqrt{\frac{2M}{r}}\frac{z}{r} \\ -\sqrt{\frac{2M}{r}}\frac{x}{r} & -1 & 0 & 0 \\ -\sqrt{\frac{2M}{r}}\frac{y}{r} & 0 & -1 & 0 \\ -\sqrt{\frac{2M}{r}}\frac{z}{r} & 0 & 0 & -1 \end{pmatrix} \tag{D13}$$

Appendix E: Underconstrained problems

We saw that, ironically, the only example where our symbolic regression failed to find an exact formula for the discovered coordinate transformation was the very simplest one: the uniform motion of example A:

$$\begin{aligned} \dot{x} &= p \\ \dot{p} &= 0 \end{aligned} \tag{E1}$$

The reason for this is that the final equations with manifest symmetry are so simple that there are infinitely many different transformations that produce it, and our neural network has no incentive to find the simplest one. Specifically, any coordinate transformation of the form $x' = c_1(p)x + c_2(p)$ and $p' = c_3(p)$ preserves both translational invariance because

$$\begin{aligned} &\dot{x}' = c_1(p)\dot{x} + c_1'(p)x\dot{p} + c_2'(p)\dot{p} = c_1'(p)p = c_1'(c_3^{-1}(p'))c_3^{-1}(p') \equiv A(p'), \\ &\dot{p}' = c_3'(p)\dot{p} = 0 \end{aligned} \tag{E2}$$

and Hamiltonicity $H = \int A(p')dp'$.

Appendix F: The AI Feynman Package

The goal of *symbolic regression* is discovering a symbolic expression that accurately matches a given data set. More specifically, we are given a table of numbers whose rows are of the form $\{x_1, \cdots, x_n, y\}$ where $y = f(x_1, \cdots, x_n)$, and our task is to discover the correct symbolic expression for the unknown mystery function f. The symbolic regression problem for mathematical functions has been tackled with a variety of methods, including sparse regression and genetic algorithms.

The *AI Feynman* software improves on these methods by using physics-inspired strategies enabled by neural networks. It uses neural networks to discover hidden simplicity such as symmetry and separability in the data, which enables it to recursively break harder problems into simpler ones with fewer variables. The overall algorithm consists of a series of modules that try to exploit each of the above-mentioned properties. Like a human scientist, it tries many different strategies (modules) in turn, and if it cannot solve the full problem in one fell swoop, it tries to transform it and divide it into simpler pieces that can be tackled separately, recursively re-launching the full algorithm on each piece. These modules include dimensional analysis, polynomial fitting, brute force experssion search, neural-network-based tests and transformations (symmetries and separability).

Note that AI Feynman can handle cases with multiple input variables, but only one output, so we run AI Feynman multiple times to obtain each component of the transformation/transformed equation. We will now illustrate how AI Feynman works on Example E and F.

For Example E (Expanding universe and empty space), this is the transformation to be discovered:

$$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} t\sqrt{1+r^2} \\ tx \\ ty \\ tz \end{pmatrix}$$

We thus need to run AI Feynman four times to learn (t', x', y', z') separately. (1) for t' (the most complicated case), by training a neural network, AI Feynman is first able to first discover $SO(3)$ symmetry, i.e., that t' depends on x, y

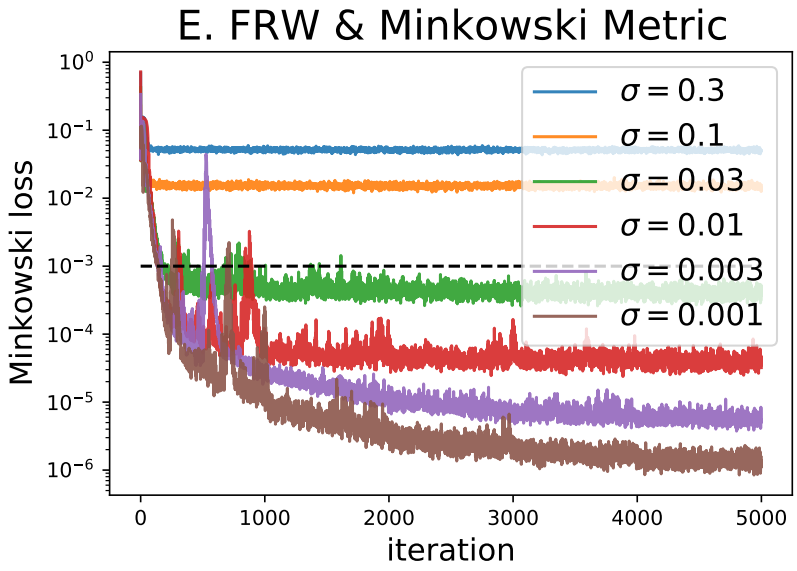

FIG. 4: Training curve for the noisy metric tensor in Example E.

and z only through their combination $r \equiv \sqrt{x^2+y^2+z^2}$. AI Feynman then tries to express t' as a function of t and r. When AI Feynman tries to square the output t'^2 and do polynomial fitting, it succeeds and find that $t'^2 = t^2 + t^2 r^2$. (2) for x', AI Feynman can discover $x' = tx$ immediately by trying polynomial fitting, and the same holds for y' and z'.

For Example F (Schwarzchild black hole and GP coordinates), this is the transformation to be discovered:

$$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} t + 2M\left[2\sqrt{\frac{r}{2M}} + \ln\frac{\sqrt{\frac{r}{2M}}-1}{\sqrt{\frac{r}{2M}}+1}\right] \\ x \\ y \\ z \end{pmatrix}$$

In our case, $2M = 1$. We again need to run AI Feynman four times to learn (t', x', y', z') separately. (1) Like in the previous example, AI Feynman first discover rotational symmetry, i.e., that t' only depends on the apace coordinates via their combination $r \equiv \sqrt{x^2+y^2+z^2}$. AI Feynman then discover additive modularity, i.e., that $t'(t,r) = g(t)+h(r)$ for some yet-to-be-discovered functions g and h, so the problem break down into two simpler problems. $g(t) = t$ is trivially discovered via polynomial fitting, while $h(r) = \left[2\sqrt{r} + \ln\frac{\sqrt{r}-1}{\sqrt{r}+1}\right]$ is discovered by brute force search over ever-more-complex symbolic formulas. (2) $x' = x$ can be easily discovered by polynomial fitting, as well as $y' = y$ and $z' = z$.

Appendix G: Noisy data

In the main text, we presented results for noise-free data. In an experimental setting, we need to quantify how noise affects affect our results. We construct noisy data by adding Gaussian noise of standard deviation σ to each component of $T(\mathbf{z})$ and rerun Example E (the Expanding universe and Minkowski space) for five different noise levels. FIG. 4 shows that the effect of data noise is simply to add a noise floor to the resulting loss; in this particular example, we see that data noise with $\sigma = 0.01$ still allows the loss to drop significantly below 0.001 and the models without hidden symmetry in Figure 3E.

Appendix H: Full Table II

A more detailed version of Table II is Table III, including (m, n, s, t) indices and results of symbolic discoveries.

TABLE III: Physical Systems studied (Full version of Table II)

ID	Name	Original dynamics $\dot{\mathbf{z}} = \mathbf{f}(\mathbf{z})$ or metric $g(\mathbf{z})$	Transformation $\mathbf{z} \mapsto \mathbf{z}'$	Symmetric dynamics $\dot{\mathbf{z}}' = \mathbf{f}'(\mathbf{z}')$ or metric $g'(\mathbf{z}')$	Manifest Symmetries	(m, n, s, t)	Symbolic sol.?
A	1D Uniform Motion	$\frac{d}{dt}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \frac{1}{2}(a+b)\ln(\frac{a-b}{2}) \\ \frac{1}{2}(a+b)\ln(\frac{a-b}{2}) \end{pmatrix}$	$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} e^{\frac{1}{2}x} + e^{\frac{1}{2}p} \\ e^{\frac{1}{2}x} - e^{\frac{1}{2}p} \end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix} x \\ p \end{pmatrix} = \begin{pmatrix} p \\ 0 \end{pmatrix}$	Hamiltonicity 1D Translational invariance	$(1,0,-1,0)$ $(1,0,-1,0)$	No, Yes
B	1D Harmonic Oscillator	$\frac{d}{dt}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} (1+a)\ln(1+b) \\ -(1+b)\ln(1+a) \end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} e^{\frac{1}{2}x} - 1 \\ e^{\frac{1}{2}p} - 1 \end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix} x \\ p \end{pmatrix} = \begin{pmatrix} p \\ -x \end{pmatrix}$	Hamiltonicity SO(2)-equivariance	$(1,0,-1,0)$ $(1,0,0,1)$	Yes, Yes
C	2D Kepler	$\frac{d}{dt}\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} (1+a)\ln(1+b) \\ -\frac{(1+b)\ln(1+a)}{8(\ln^2(1+a)+\ln^2(1+c))^{3/2}} \\ (1+c)\ln(1+d) \\ -\frac{(1+d)\ln(1+c)}{8(\ln^2(1+a)+\ln^2(1+c))^{3/2}} \end{pmatrix}$	$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} e^{\frac{1}{2}x} - 1 \\ e^{\frac{1}{2}p_x} - 1 \\ e^{\frac{1}{2}y} - 1 \\ e^{\frac{1}{2}p_y} - 1 \end{pmatrix}$	$\frac{d}{dt}\begin{pmatrix} x \\ p_x \\ y \\ p_y \end{pmatrix} = \begin{pmatrix} p_x \\ -x/r^3 \\ p_y \\ -y/r^3 \end{pmatrix}$	Hamiltonicity Can SO(2)-equivariance	$(1,0,-1,0)$ $(1,0,0,1)$	Yes, Yes
D	Double Pendulum	$\frac{d}{dt}\begin{pmatrix} \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{pmatrix} = \begin{pmatrix} \dot{\theta}_1 \\ -\frac{(m_1+m_2)g}{m_1 l}\theta_1 + \frac{m_2 g}{m_1 l}\theta_2 \\ \dot{\theta}_2 \\ \frac{(m_1+m_2)g}{m_1 l}\theta_1 - \frac{(m_1+m_2)g}{m_1 l}\theta_2 \end{pmatrix}$	$\begin{pmatrix} \theta_1 \\ \theta_2 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{pmatrix} = \begin{pmatrix} -1 & 1 & 0 & 0 \\ a & a & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & a & a \end{pmatrix} \begin{pmatrix} \theta_+ \\ \theta_- \\ \dot{\theta}_+ \\ \dot{\theta}_- \end{pmatrix}$ $a = \sqrt{\frac{m_1+m_2}{m_2}}$	$\frac{d}{dt}\begin{pmatrix} \theta_+ \\ \dot{\theta}_+ \\ \theta_- \\ \dot{\theta}_- \end{pmatrix} = \begin{pmatrix} \dot{\theta}_+ \\ -\omega_+^2 \theta_+ \\ \dot{\theta}_- \\ -\omega^2\ \theta_- \end{pmatrix}$ $\omega_\pm^2 = \frac{(m_1+m_2)g}{m_1 l}(1 \pm \sqrt{\frac{m_2}{m_1+m_2}})$	Hamiltonicity $(2+2)$-Modularity	$(1,0,-1,0)$ $(1,0,-1,0)$	Yes, Yes
E	Expanding universe & empty space	$\mathbf{g} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -(r^2 + \frac{kx^2}{1-kr^2})\frac{t^2}{r^2} & -\frac{kxyt^2}{1-kr^2} & -\frac{kxzt^2}{1-kr^2} \\ 0 & -\frac{kxyt^2}{1-kr^2} & -(r^2 + \frac{ky^2}{1-kr^2})\frac{t^2}{r^2} & -\frac{kyzt^2}{1-kr^2} \\ 0 & -\frac{kxzt^2}{1-kr^2} & -\frac{kyzt^2}{1-kr^2} & -(r^2 + \frac{kz^2}{1-kr^2})\frac{t^2}{r^2} \end{pmatrix}$	$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} t\sqrt{1+r^2} \\ tx \\ ty \\ tz \end{pmatrix}$	$\mathbf{g} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$	SO(3,1)-Invariance 4D Translational Invariance	$(0,2,0,-2)$ $(0,2,-1,-3)$	Yes, Yes
F	Schwarzchild black hole & GP metric	$\mathbf{g} = \begin{pmatrix} 1-\frac{2M}{r} & 0 & 0 & 0 \\ 0 & -1-\frac{2Mx^2}{(r-2M)r^2} & -\frac{2Mxy}{(r-2M)r^2} & -\frac{2Mxz}{(r-2M)r^2} \\ 0 & -\frac{2Mxy}{(r-2M)r^2} & -1-\frac{2My^2}{(r-2M)r^2} & -\frac{2Myz}{(r-2M)r^2} \\ 0 & -\frac{2Mxz}{(r-2M)r^2} & -\frac{2Myz}{(r-2M)r^2} & -1-\frac{2Mz^2}{(r-2M)r^2} \end{pmatrix}$	$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} t + 2M\left[2u + \ln\frac{u-1}{u+1}\right] \\ x \\ y \\ z \end{pmatrix}$ $u \equiv \sqrt{\frac{r}{2M}}$	$\mathbf{g} = \begin{pmatrix} 1-\frac{2M}{r} & -\sqrt{\frac{2M}{r}}\frac{x}{r} & -\sqrt{\frac{2M}{r}}\frac{y}{r} & -\sqrt{\frac{2M}{r}}\frac{z}{r} \\ -\sqrt{\frac{2M}{r}}\frac{x}{r} & -1 & 0 & 0 \\ -\sqrt{\frac{2M}{r}}\frac{y}{r} & 0 & -1 & 0 \\ -\sqrt{\frac{2M}{r}}\frac{z}{r} & 0 & 0 & -1 \end{pmatrix}$	SO(3)-Invariance 3D Translational Invariance	$(0,2,0,-2)$ $(0,2,-1,-3)$	Yes, Yes