

Discovering physical concepts with neural networks

Raban Iten,* Tony Metger,* Henrik Wilming, Lidia del Rio, and Renato Renner
ETH Zurich, Wolfgang-Pauli-Str. 27, 8093 Zurich, Switzerland.

(Dated: January 24, 2020)

Despite the success of neural networks at solving concrete physics problems, their use as a general-purpose tool for scientific discovery is still in its infancy. Here, we approach this problem by modelling a neural network architecture after the human physical reasoning process, which has similarities to representation learning. This allows us to make progress towards the long-term goal of machine-assisted scientific discovery from experimental data without making prior assumptions about the system. We apply this method to toy examples and show that the network finds the physically relevant parameters, exploits conservation laws to make predictions, and can help to gain conceptual insights, e.g. Copernicus’ conclusion that the solar system is heliocentric.

Theoretical physics, like all fields of human activity, is influenced by the schools of thought prevalent at the time of development. As such, the physical theories we know may not necessarily be the simplest ones to explain experimental data, but rather the ones that most naturally followed from a previous theory at the time. Both general relativity and quantum theory were built upon classical mechanics — they have been impressively successful in the restricted regimes of the very large and very small, respectively, but are fundamentally incompatible, as reflected by paradoxes such as the black hole information loss [1, 2]. This raises an interesting question: are the laws of quantum physics, and other physical theories more generally, the most natural ones to explain data from experiments if we assume no prior knowledge of physics? While this question will likely not be answered in the near future, recent advances in artificial intelligence allow us to make a first step in this direction. Here, we investigate whether neural networks can be used to discover physical concepts from experimental data.

Previous work. The goal of using machines to help with discovering the physical laws underlying experimental data has been pursued in several contexts (see Appendix B for a more detailed overview and [3–6] for recent reviews). A lot of early work focused on finding mathematical expressions describing a given dataset (see e.g. [7–9]). For example, in [8] an algorithm recovers the laws of motion of simple mechanical systems, like a double pendulum, by searching over a space of mathematical expressions on given input variables. More recently, significant progress was made in extracting dynamical equations from experimental data [10–18]. These methods are highly practical and they were successfully applied to complex physical systems, but require prior knowledge on the systems of interest, for example in the form of knowing what the relevant variables are or that dynamics should be described by differential equations. In certain situations one might not have such prior knowledge or does not want to impose it to allow the machine

to find entirely different representations of the physical system.

Over the last few years, neural networks have become the dominant method in machine learning and they have successfully been used to tackle complex problems in classical as well as quantum physics (see Appendix B for further discussions). Conversely, neural networks may also lead to new insights into how the human brain develops physical intuition from observations [19–25]. Very recently, physical variables were extracted in an unsupervised way from time series data of dynamical systems in [26].

Our goal in this work is to minimize the extent to which prior assumptions about physical systems impose structure on the machine learning system. Eliminating assumptions that may not be satisfied for all physical systems, such as assuming that particles only interact in a pairwise manner, is necessary for the long-term goal of an artificial intelligence physicist (see [27] for recent progress in this direction) that can be applied to any system without a need for adaptations and might eventually contribute to progress in the foundations of physics. Very recently, neural networks were used in this spirit to detect differences between observed data and a reference model [28, 29]. However, there is a tradeoff between generality and performance, and the performance of the machine learning system proposed here — based on autoencoders [30–32] — is not yet comparable to more established approaches that are adapted to specific physical systems.

Modelling the physical reasoning process. This work makes progress towards an interpretable artificial intelligence agent that is unbiased by prior knowledge about physics by proposing to focus on the human physical modelling process itself, rather than on specific physical systems. We formalize a simplified physical modelling process, which we then translate into a neural network architecture. This neural network architecture can be applied to a wide variety of physical systems, both classical and quantum, and is flexible enough to accommodate different additional desiderata on representations of the system that we may wish to impose.

We start by considering a simplified version of the physical modelling process, pictured in Figure 1a. Physi-

* These two authors contributed equally.

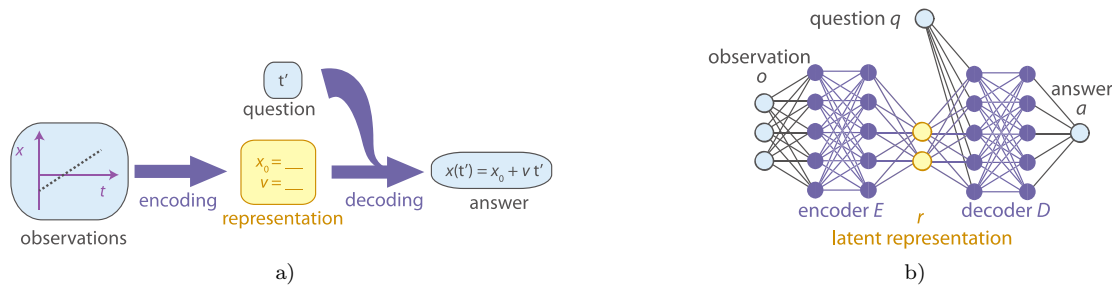


Figure 1. **Learning physical representations.** (a) **Human learning.** A physicist compresses experimental observations into a simple representation (*encoding*). When later asked any question about the physical setting, the physicist should be able to produce a correct answer using only the representation and not the original data. We call the process of producing the answer from the representation *decoding*. For example, the observations may be the first few seconds of the trajectory of a particle moving with constant speed; the representation could be the parameters “speed v ” and “initial position x_0 ” and the question could be “where will the particle be at a later time t' ?” (b) **Neural network structure for *SciNet*.** Observations are encoded as real parameters fed to an encoder (a *feed-forward neural network*, see Appendix D), which compresses the data into a representation (*latent representation*). The question is also encoded in a number of real parameters, which, together with the representation, are fed to the decoder network to produce an answer. (The number of neurons depicted is not representative.)

cists’ interactions with the physical world take the form of experimental observations (e.g. a time series $(t_i, x(t_i))_{i \in \{1, \dots, N\}}$ describing the motion of a particle at constant speed). The models physicists build do not deal with these observations directly, but rather with a representation of the underlying physical state of the observed system (e.g. the two parameters *initial position* and *speed*, (x_0, v)). Which parameters are used is an important part of the model, and we will give suggestions about what makes a good representation below. Finally, the model specifies how to make predictions (i.e., answer questions) based on the knowledge of the physical state of the system (e.g. “where is the particle at time t' ?”). More formally, this physical modelling process can be regarded as an “encoder” $E : \mathcal{O} \rightarrow \mathcal{R}$ mapping the set of possible observations \mathcal{O} to representations \mathcal{R} , followed by a “decoder” $D : \mathcal{R} \times \mathcal{Q} \rightarrow \mathcal{A}$ mapping the sets of all possible representations \mathcal{R} and questions \mathcal{Q} to answers \mathcal{A} .

Network structure. This modelling process can be translated directly into a neural network architecture, which we refer to as *SciNet* in the following (Figure 1b). The encoder and decoder are both implemented as feed-forward neural networks. The resulting architecture, except for the question input, resembles an autoencoder in representation learning [30, 31], and more specifically the architecture in [33]. During the training, we provide triples of the form $(o, q, a_{\text{corr}}(o, q))$ to the network, where $a_{\text{corr}}(o, q) \in \mathcal{A}$ is the correct reply to question $q \in \mathcal{Q}$ given the observation $o \in \mathcal{O}$. The learned parameterization is typically called latent representation [30, 31]. To feed the questions into the neural network, they are encoded into a sequence of real parameters. Thereby, the actual representation of a single question is irrelevant as long as it allows the network to distinguish questions that require different answers.

It is crucial that the encoder is completely free to

choose a latent representation itself, instead of us imposing a specific one. Because neural networks with at least one hidden layer composed of sufficiently many neurons can approximate any continuous function arbitrarily well [34], the fact that the functions E and D are implemented as neural networks does not significantly restrict their generality. However, unlike in an autoencoder, the latent representation need not describe the observations completely; instead, it only needs to contain the information necessary to answer the questions posed.

This architecture allows us to extract knowledge from the neural network: all of the useful information is stored in the representation, and the size of this representation is small compared to the total number of degrees of freedom of the network. This allows us to interpret the learnt representation. Specifically, we can compare *SciNet*’s latent representation to a hypothesized parameterization to obtain a simple map from one to the other. If we do not even have any hypotheses about the system at hand, we may still gain some insights solely from the number of required parameters or from studying the change in the representation when manually changing the input, and the change in output when manually changing the representation (as in e.g. [32]).

Desired properties for a representation. For *SciNet* to produce physically useful representations, we need to formalize what makes a good parameterization of a physical system, i.e., a good latent representation. We stress that this is not a property of a physical system, but a choice we have to make. We will give two possible choices below.

Generally, the latent representation should only store the minimal amount of information that is sufficient to correctly answer all questions in \mathcal{Q} . For *minimal sufficient uncorrelated representations*, we additionally require that the latent neurons be statistically independent from each other for an input sampled at random from

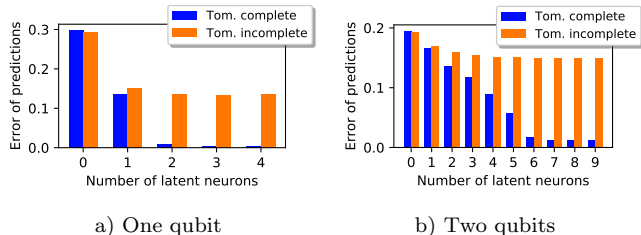


Figure 2. **Quantum tomography.** *SciNet* is given tomographic data for one or two qubits, as shown in part a) and b) of the figure, respectively, and an operational description of a measurement as a question input and has to predict the probabilities of outcomes for this measurement. The plots show the root mean square error of *SciNet*’s measurement predictions for test data as a function of the number of latent neurons. In the tomographically complete case, *SciNet* recovers the number of (real) degrees of freedom required to describe a one and a two qubit state (which are two and six, respectively). Tomographically incomplete data can be recognized, since the prediction error remains high as one increases the number of latent neurons.

the training data, reflecting the idea that physically relevant parameters describe aspects of a system that can be varied independently and are therefore uncorrelated in the experimental data. Under this independence assumption, the network is then motivated to choose a representation that stores different physical parameters in different latent neurons. We formalize these demands in Appendix C and show, using techniques from differential geometry, that the number of latent neurons equals the number of underlying degrees of freedom in the training data that are needed to answer all questions \mathcal{Q} . To implement these requirements in a neural network, we use well-established methods from representation learning, specifically disentangling variational autoencoders [32, 35] (see Appendix D 1 for details).

Alternatively, for situations where the physically relevant parameters can change, either over time or by some time-independent update rule, we might prefer a representation with a simple such update rule. We explain below how this requirement can be enforced.

Results. To demonstrate that *SciNet* helps to recover relevant concepts in physics by providing the relevant physical variables, both in quantum- and classical-mechanical settings, we consider four toy examples from different areas of physics. In summary, we find: (i) given a time series of the positions of a damped pendulum, *SciNet* can predict future positions with high accuracy and it uses the relevant parameters, namely frequency and damping factor, separately in two of the latent neurons (and sets the activation of unnecessary latent neurons to zero); (ii) *SciNet* finds and exploits conservation laws: it uses the total angular momentum to predict the motion of two colliding particles; (iii) given measurement data from a simple quantum experiment, *SciNet*

can be used to determine the dimension of the underlying unknown quantum system and to decide whether a set of measurements is tomographically complete, i.e., whether it provides full information about the quantum state; (iv) given a time series of the positions of the Sun and Mars as observed from Earth, *SciNet* switches to a heliocentric representation — that is, it encodes the data into the angles of the two planets as seen from the Sun. The results show that *SciNet* finds, without having been given any prior information about the specific physical systems, the same quantities that we use in physics textbooks to describe the different settings. We also show that our results are robust against noise in the experimental data. To illustrate our approach, we will now describe two of these settings in some depth. For detailed descriptions of the four different settings, the data generation, interpretation and additional background information, we refer to Appendix E.

In all our examples, the training data we use is operational and could be generated from experiments, i.e., the correct answer is the one observed experimentally. Here, we use simulations instead because we only deal with classical and quantum mechanics, theories whose predictions are experimentally well tested in the relevant regimes. One might think that using simulated data would restrict *SciNet* to rediscovering the theory used for data generation. However, in particular for quantum mechanics, we are interested in finding conceptually different formulations of the theory with the same predictions.

Quantum state tomography. In quantum mechanics, it is not trivial to construct a simple representation of the state of a quantum system from measurement data, a task called quantum tomography [36]. In the following, we will show that *SciNet* finds representations of arbitrary (pure) one- and two-qubit states. To ensure that no prior knowledge about quantum physics is required to collect the measurement data, we assume an operational setting in which we have access to two devices in a lab, where one device can create (many copies of) a quantum system in a certain state depending on the chosen parameters of the device. The other device performs binary measurements on the quantum system. The input to *SciNet* consists of the outcome probabilities of a random fixed set of “reference measurements” on quantum systems in the unknown quantum state ψ . As a question input, we provide a parameterization of a measurement ω (one may think of the setting of the dials and buttons of the measurement device). *SciNet* has to predict the outcome probability of the measurement ω on a quantum system in the state ψ . We train *SciNet* with different pairs (ω, ψ) for one and two qubits. The results are shown in Figure 2. Training different networks with different numbers of latent neurons, we can observe how the quality of the predictions (after training has been completed) improves as we allow for more parameters in the representation of ψ . This allows us to gain relevant information, without previous hypotheses about the nature of this representation (for example, whether it is a

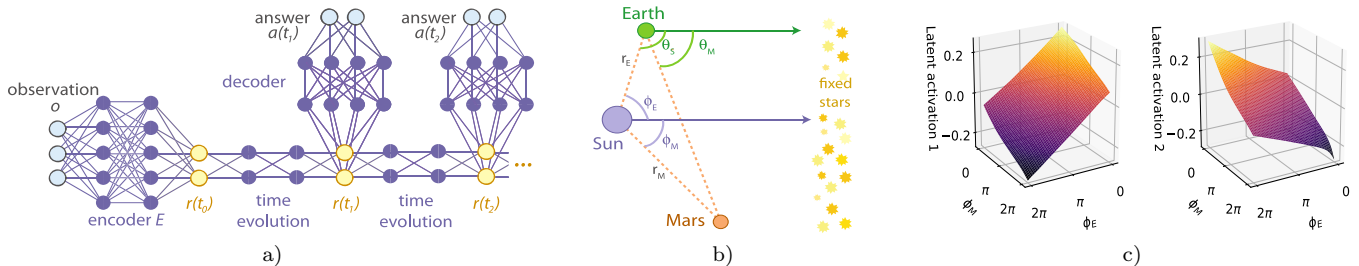


Figure 3. **Heliocentric model of the solar system.** *SciNet* is given the angles of the Sun and Mars as seen from Earth at an initial time t_0 and has to predict these angles for later times. **(a) Recurrent version of *SciNet* for time-dependent variables.** Observations are encoded into a simple representation $r(t_0)$ at time t_0 . Then, the representation is evolved in time to $r(t_1)$ and a decoder is used to predict $a(t_1)$, and so on. In each (equally spaced) time step, the same time evolution network and decoder network are applied. **(b) Physical setting.** The heliocentric angles ϕ_E and ϕ_M of the Earth and Mars are observed from the Sun; the angles θ_S and θ_M of the Sun and Mars are observed from Earth. All angles are measured relative to the fixed star background. **(c) Representation learned by *SciNet*.** The activations $r_{1,2}(t_0)$ of the two latent neurons at time t_0 (see Figure 3a) are plotted as a function of the heliocentric angles ϕ_E and ϕ_M . The plots show that the network stores and evolves parameters that are linear combinations of the heliocentric angles.

vector in a Hilbert space).

If the reference measurements are tomographically complete, meaning that they are sufficient to reconstruct a complete representation of the underlying quantum system, the plots in Figure 2 show a drop in prediction error when the number of latent neurons is increased up to two and six for the cases of one and two qubits, respectively [37]. This is in accordance with the number of degrees of freedom required to describe a one- or a two-qubit state in our current theory of quantum mechanics. For the case where the set of measurements is tomographically incomplete, it is not possible for *SciNet* to predict the outcome of the final measurement perfectly regardless of the number of latent neurons. This means that purely from operational data, we can make a statement about the tomographic completeness of measurements and about the number of degrees of freedom of the underlying unknown quantum system.

Enforcing a simple time evolution. As mentioned above, if the physically relevant parameters can change, we can enforce a representation that has a simple update rule. For illustration, we will consider time evolution here, but more general update rules are possible. To accommodate changing physical parameters, we need to extend the latent representation as shown in Figure 3a. Instead of a single latent representation with a decoder attached to it, we now have many latent representations that are generated from the initial representation by a time evolution network. Each representation has a decoder attached to it to produce an answer to a question. Because we only want the parameters, but not the physical model, to change in time, all time evolution steps and decoders are identical, i.e., they implement the same function. The encoder, time evolution network, and decoder are trained simultaneously. To enforce parameters with a simple time evolution, we restrict the time evolution network to implementing very simple functions, such as addition of a constant [38].

Heliocentric solar system. In the 16th century, Copernicus used observations of the positions of different planets on the night sky (Figure 3b) to hypothesize that the Sun, and not the Earth, is at the centre of our solar system. This heliocentric view was confirmed by Kepler at the start of the 17th century based on astronomic data collected by Brahe, showing that the planets move around the Sun in simple orbits. Here, we show that *SciNet* similarly uses heliocentric angles when forced to find a representation for which the time evolution of the variables takes a very simple form, a typical requirement for time-dependent variables in physics.

The observations given to *SciNet* are angles $\theta_M(t_0)$ of Mars and $\theta_S(t_0)$ of the Sun as seen from Earth at a starting time t_0 (which is varied during training). The time evolution network is restricted to addition of a constant (the value of which is learned during training). At each time step i , *SciNet* is asked to predict the angles as seen from Earth at the time t_i using only its representation $r(t_i)$. Because this question is constant, we do not need to feed it to the decoder explicitly.

We train *SciNet* with randomly chosen subsequences of weekly (simulated) observations of the angles θ_M and θ_S within Copernicus' lifetime (3665 observations in total). For our simulation, we assume circular orbits of Mars and Earth around the Sun. Figure 3c shows the learned representation and confirms that *SciNet* indeed stores a linear combination of *heliocentric angles*. We stress that the training data only contains angles observed from Earth, but *SciNet* nonetheless switches to a heliocentric representation.

Conclusion. In this work, we have shown that *SciNet* can be used to recover physical variables from experimental data in various physical toy settings. The learnt representations turned out to be the ones commonly used in physics textbooks, under the assumption of uncorrelated sampling. In future work we plan to extend our approach to data where the natural underlying parameters

are correlated in the training distribution. The separation of these parameters in the representation found by *SciNet* requires the development of further operational criteria for disentangling latent variables. In more complex scenarios, the methods introduced here may lead to entirely novel representations, and extracting human physical insight from such representations remains challenging. This could be addressed using methods from symbolic regression [39] to obtain analytical expressions for the encoder and decoder maps, or for a map between a hypothesized and the actual representation. Alternatively, methods such as the ones presented in [40, 41] could help to improve the interpretability of the representation. Following this direction, it might eventually become possible for neural networks to produce insights expressed in our mathematical language.

Acknowledgments. We would like to thank Alessandro Achille, Serguei Belousov, Ulrich Eberle, Thomas Frerix, Viktor Gal, Thomas Häner, Maciej Koch-Janusz, Aurelien Lucchi, Ilya Nemenman, Joseph M. Renes, Andrea Rocchetto, Norman Sieroka, Ernest Y.-Z. Tan, Jinzhao Wang and Leonard Wossnig for helpful discussions. We acknowledge support from the Swiss National Science Foundation through SNSF project No. 200020_165843, the Swiss National Supercomputing Centre (CSCS) under project ID da04, and through the National Centre of Competence in Research *Quantum Science and Technology* (QSIT). LdR and RR furthermore acknowledge support from the FQXi grant *Physics of the observer*. TM acknowledges support from ETH Zürich and the ETH Foundation through the *Excellence Scholarship & Opportunity Programme*.

The source code and the training data are available at: <https://github.com/eth-nn-physics/nn.physical.concepts>. See also Appendix A for the implementation details. *SciNet* worked well on all tested examples, i.e., we did not post-select examples based on whether *SciNet* worked or not.

Appendix A: Implementation

The neural networks used in this work are specified by the input size for question and observation input, the output size, the number of latent neurons, and the sizes of encoder and decoder. The number of neurons in the encoder and decoder are not expected to be important for the results, provided the encoder and decoder are large enough to not constrain the expressivity of the network (and small enough to be efficiently trainable). The training is specified by the number of training examples, the batch size, the number of epochs, the learning rate and the value of the parameter β (see section D 1). To test the networks, we use a number of previously unseen test samples. We give the values of these parameters for the

examples presented in the main text (and described in detail in Section E) in Table I and Table II.

The source code, all details about the network structure and training process, and pre-trained *SciNets* are available at

<https://github.com/eth-nn-physics/nn.physical.concepts>.

The networks were implemented using the Tensorflow library [42]. For all examples, the training process only takes a few hours on a standard laptop.

Appendix B: Detailed comparison with previous work

Neural networks have become a standard tool to tackle problems where we want to make predictions without following a particular algorithm or imposing structure on the available data (see for example [44–46]) and they have been applied to a wide variety of problems in physics. For example, in condensed matter physics and generally in many-body settings, neural networks have proven particularly useful to characterize phase transitions (see [3] and references therein) and to learn local symmetries [47].

In quantum optics, automated search techniques and reinforcement-learning based schemes have been used to generate new experimental setups [48, 49]. Projective simulation [50] is used in [49] to autonomously discover experimental building blocks with maximum versatility.

Closer to our work, neural networks have also been used to efficiently represent wave functions of particular quantum systems [51–65]. In particular, in [52], variational autoencoders are used to approximate the distribution of the measurement outcomes of a specific quantum state for a fixed measurement basis and the size of the neural network can provide an estimate for the complexity of the state. In contrast, our approach does not focus on an efficient representation of a given quantum state and it is not specifically designed for learning representations of quantum systems. Nevertheless, *SciNet* can be used to produce representations of arbitrary states of simple quantum systems without retraining. This allows us to extract information about the degrees of freedom required to represent any state of a (small) quantum system.

Another step towards extracting physical knowledge in an unsupervised way is presented in [66]. The authors show how the relevant degrees of freedom of a system in classical statistical mechanics can be extracted under the assumption that the input is drawn from a Boltzmann distribution. They make use of information theory to guide the unsupervised training of restricted Boltzmann machines, a class of probabilistic neural networks, to approximate probability distributions.

A different line of work has focused on using neural networks and other algorithmic techniques to better un-

Example	Observation input size	Question input size	# Latent neurons	Output size	Encoder	Decoder
Pendulum	50	1	3	1	[500, 100]	[100, 100]
Collision	30	16	1	2	[150, 100]	[100, 150]
One qubit	10	10	0-5	1	[100, 100]	[100, 100]
Two qubits	30	30	0-9	1	[300, 100]	[100, 100]
Solar system 2	2	0	2	2	[100, 100]	[100, 100]

Table I. Parameters specifying the network structure. The first four examples use the network structure depicted in Figure 1, whereas the last example uses the structure shown in Figure 3a. The notation $[n_1, n_2]$ is used to describe the number of neurons n_1 and n_2 in the first and the second hidden layer of the encoder (or the decoder), respectively.

Example	Batch size	Learning rate	β	# Epochs	# Training samples	# Test samples
Pendulum	512	10^{-3}	10^{-3}	1000	95000	5000
Collision	500	$(5 \cdot 10^{-4}, 10^{-4})$	0	(100, 50)	490000	10000
One qubit	512	$(10^{-3}, 10^{-4})$	10^{-4}	(250, 50)	95000	5000
Two qubits	512	$(10^{-3}, 10^{-4})$	10^{-4}	(250, 50)	490000	10000
Solar system	256 – 2048	$10^{-5} - 10^{-4}$	0.001 – 0.1	15000	95000	5000

Table II. Parameters specifying the training process. For training with two phases, the notation (p_1, p_2) refers to the parameters in the first and second phase, respectively. The last example uses five training phases specified in detail in [43].

derstand how humans are able to gain an intuitive understanding of physics [19–25, 67–69].

Very recently, physical variables were extracted in an unsupervised way from time series data of dynamical systems [26]. The network structure used in [26] is built on interaction networks [70–72] and it is well adapted to physical systems consisting of several objects interacting in a pair-wise manner. The prior knowledge included in the network structure allows the network to generalise to situations that differ substantially from those seen during training.

In the last few years, significant progress was made in extracting dynamical equations from experimental data [10–18], which is known to be an NP-hard problem [73]. Where the most of these works search for dynamical models in the input data, in [12–14] neural networks are used to find a new set of variables such that the evolution of the new variables is approximately linear (motivated by Koopman operator theory). Our example given in Section E 4 uses a similar network structure as the one used in [12–14], which corresponds to a special case of *SciNet* with a trivial question input and a latent representation that is evolved in time. The concept of evolving the system in the latent representation has also been used in machine learning to extract the relevant features from video data [74]. A further step towards an artificial intelligence physicist was taken in [27], where data from complex environments is automatically separated into parts corresponding to systems that can be explained by simple physical laws. The machine learning system then tries to unify the underlying “theories” found for the different parts of the data.

Appendix C: Minimal representations

Here, we describe some of the theoretical considerations that went into designing *SciNet* and helping it to find useful representations that encode physical principles. Given a data set, it is generally a complex task to find a simple representation of the data that contains all the desired information. *SciNet* should recover such representations by itself; however, we encourage it to learn “simple” representations during training. To do so, we have to specify the desired properties of a representation. In this, our approach follows the spirit of several works on representation learning theory [30–32, 35, 75, 76].

For the theoretical analysis, we introduce some additional structure on the data that is required to formulate the desired properties of a representation. We consider real-valued data, which we think of as being sampled from some unknown probability distribution. In other words, we assign random variables to the observations O , the questions Q , the latent representation R , and the answers A . We use the convention that a random variable $X = (X_1, \dots, X_{|X|})$ takes samples in $\mathcal{X} \subset \mathbb{R}^{|X|}$, where $|X|$ denotes the dimension of the ambient space of X . In particular, $|R|$ will correspond to the number of neurons in the latent representation.

We require the following properties for an *uncorrelated (sufficient) representation* R (defined by an encoder mapping $E : \mathcal{O} \rightarrow \mathcal{R}$) for the data described by the triple (O, Q, a_{corr}) , where we recall that the function $a_{\text{corr}} : \mathcal{O} \times \mathcal{Q} \rightarrow \mathcal{A}$ sends an observation $o \in \mathcal{O}$ and a question $q \in \mathcal{Q}$ to the correct answer $a \in \mathcal{A}$.

1. **Sufficient (with smooth decoder):** There exists a smooth map $D : \mathcal{R} \times \mathcal{Q} \mapsto \mathcal{A}$, such that $D(E(o), q) = a_{\text{corr}}(o, q)$ for all possible observations $o \in \mathcal{O}$ and questions $q \in \mathcal{Q}$.
2. **Uncorrelated:** The elements in the set $\{R_1, R_2, \dots, R_{|R|}\}$ are mutually independent.

Property 1 asserts that the encoder map E encodes all information of the observation $o \in \mathcal{O}$ that is necessary to reply to all possible questions $q \in \mathcal{Q}$. We require the decoder to be smooth, since this allows us to give the number of parameters stored in the latent representation a well defined meaning in terms of a dimension (see Section C 1).

Property 2 means that knowing some variables in the latent representation does not provide any information about any other latent variables; note that this depends on the distribution of the observations.

We define a *minimal uncorrelated representation* R as an uncorrelated (sufficient) representation with a minimal number of parameters $|R|$. This formalizes what we consider to be a “simple” representation of physical data.

Without the assumption that the decoder is smooth, it would, in principle, always be sufficient to have a single latent variable, since a real number can store an infinite amount of information. Hence, methods from standard information theory, like the information bottleneck [77–79], are not the right tool to give the number of variables a formal meaning. In Section C 1, we use methods from differential geometry to show that the number of variables $|R|$ in a minimal (sufficient) representation corresponds to the number of relevant degrees of freedom in the observation data required to answer all possible questions.

1. Interpretation of the number of latent variables

Above, we have required that the latent representation should contain a minimal amount of latent variables; we now relate this number to the structure of the given data. Proposition 2 below asserts that the minimal number of latent neurons corresponds to the relevant degrees of freedom in the observed data required to answer all the questions that may be asked.

For simplicity, we describe the data with sets instead of random variables here. Note that the probabilistic structure was only used for Property 2 in Section C, whereas here, we are only interested in the number of latent neurons and not in that they are mutually independent. We therefore consider the triple $(\mathcal{O}, \mathcal{Q}, a_{\text{corr}})$, where \mathcal{O} and \mathcal{Q} are the sets containing the observation data and the questions respectively, and the function $a_{\text{corr}} : (o, q) \mapsto a$ sends an observation $o \in \mathcal{O}$ and a question $q \in \mathcal{Q}$ to the correct reply $a \in \mathcal{A}$.

Intuitively, we say that the triple $(\mathcal{O}, \mathcal{Q}, a_{\text{corr}})$ has dimension at least n if there exist questions in \mathcal{Q} that are able to capture n degrees of freedom from the observation data \mathcal{O} . Smoothness of this “oracle” is a natural requirement, in the sense that we expect the dependence of the answers on the input to be robust under small perturbations. The formal definition follows.

Definition 1 (Dimension of a data set). *Consider a data set described by the triple $(\mathcal{O}, \mathcal{Q}, a_{\text{corr}})$, where $a_{\text{corr}} : \mathcal{O} \times \mathcal{Q} \rightarrow \mathcal{A}$, and all sets are real, $\mathcal{O} \subseteq \mathbb{R}^r$, $\mathcal{Q} \subseteq \mathbb{R}^s$, $\mathcal{A} \subseteq \mathbb{R}^t$. We say that this triple has dimension at least n if there exists an n -dimensional submanifold $\mathcal{O}_n \subseteq \mathcal{O}$ and questions $q_1, \dots, q_k \in \mathcal{Q}$ and a function*

$$f : \mathcal{O}_n \rightarrow \mathcal{A}^k := \overbrace{\mathcal{A} \times \mathcal{A} \times \dots \times \mathcal{A}}^k \\ o \mapsto [a_{\text{corr}}(o, q_1), \dots, a_{\text{corr}}(o, q_k)]$$

such that $f : \mathcal{O}_n \rightarrow f(\mathcal{O}_n)$ is a diffeomorphism.

Proposition 2 (Minimal representation for *SciNet*). *A (sufficient) latent representation for data described by a triple $(\mathcal{O} \subset \mathbb{R}^r, \mathcal{Q} \subset \mathbb{R}^s, a_{\text{corr}} : \mathcal{O} \times \mathcal{Q} \rightarrow \mathcal{A} \subset \mathbb{R}^t)$ of dimension at least n requires at least n latent variables.*

Proof. By assumption, there is an n -dimensional submanifold $\mathcal{O}_n \subset \mathcal{O}$ and k questions q_1, \dots, q_k such that $f : \mathcal{O}_n \rightarrow \mathcal{I}_n := f(\mathcal{O}_n)$ is a diffeomorphism. We prove the statement by contradiction: assume that there exists a (sufficient) representation described by an encoder $E : \mathcal{O} \rightarrow \mathcal{R}_m \subset \mathbb{R}^m$ with $m < n$ latent variables. By sufficiency of the representation, there exists a smooth decoder $D : \mathcal{R}_m \times \mathcal{Q} \rightarrow \mathcal{A}$ such that $D(E(o), q) = a_{\text{corr}}(o, q)$ for all observations $o \in \mathcal{O}$ and questions $q \in \mathcal{Q}$. We define the smooth map

$$\tilde{D} : \mathcal{R}_m \rightarrow \mathcal{A}^k \\ r \mapsto [D(r, q_1), \dots, D(r, q_k)],$$

and denote the pre-image of \mathcal{I}_n by $\tilde{\mathcal{R}}_m := \tilde{D}^{-1}(\mathcal{I}_n)$.

By sufficiency of the representation, the restriction of the map \tilde{D} to $\tilde{\mathcal{R}}_m$ denoted by $\tilde{D}|_{\tilde{\mathcal{R}}_m} : \tilde{\mathcal{R}}_m \rightarrow \mathcal{I}_n$ is a smooth and surjective map. However, by Sard’s theorem (see for example [80]), the image $\tilde{D}(\tilde{\mathcal{R}}_m)$ is of measure zero in \mathcal{I}_n , since the dimension of the domain $\tilde{\mathcal{R}}_m \subset \mathbb{R}^m$ is at most m , which is smaller than the dimension n of the image \mathcal{I}_n . This contradicts the surjectivity of $\tilde{D}|_{\tilde{\mathcal{R}}_m}$ and finishes the proof. \square

We can consider an autoencoder as a special case of *SciNet*, where we ask always the same question and expect the network to reproduce the observation input. Hence, an autoencoder can be described by a triple $(\mathcal{O}, \mathcal{Q} = \{0\}, a_{\text{corr}} : (o, 0) \mapsto o)$. As a corollary of Proposition 2, we show that in the case of an autoencoder, the required number of latent variables corresponds to the “relevant” number of degrees of freedom that describe the observation input. The relevant degrees of freedom,

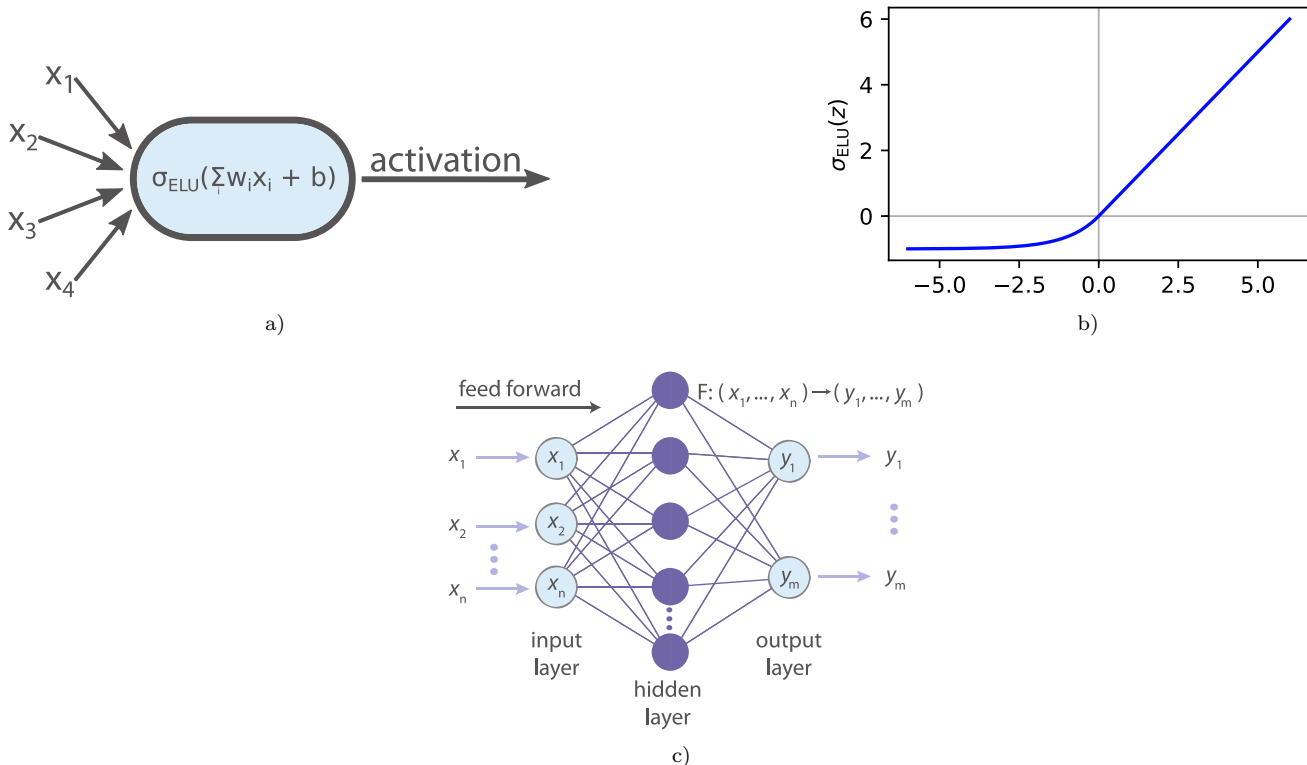


Figure 4. **Neural networks.** (a) Single artificial neuron with weights w_i , bias b and ELU activation function σ_{ELU} . The inputs to the neuron are denoted by x_1, \dots, x_4 . (b) ELU activation function for $\alpha = 1$. (c) Fully connected (feed-forward) neural network with 3 layers. The network as a whole can be thought of as a function mapping the inputs (x_1, \dots, x_n) to the output (y_1, \dots, y_m) .

which are called (*hidden*) *generative factors* in this context in representation learning (see for example [32]), may be described by the dimension of the domain of a smooth nondegenerate data generating function H , defined as follows.

Definition 3. We say that a smooth function $H : \mathcal{G} \subset \mathbb{R}^d \rightarrow \mathbb{R}^r$ is nondegenerate if there exists an open subset $\mathcal{N}_d \subset \mathcal{G}$ such that the restriction $H|_{\mathcal{N}_d} : \mathcal{N}_d \rightarrow H(\mathcal{N}_d)$ of H on \mathcal{N}_d is a diffeomorphism.

One may think of H as sending a small dimensional representation of the data onto a manifold in a high dimensional space of observations.

Corollary 4 (Minimal representation for an autoencoder). Let $H : \mathcal{G} \subset \mathbb{R}^d \rightarrow \mathcal{O} \subset \mathbb{R}^r$ be a smooth, nondegenerate and surjective (data generating) function, and let us assume that \mathcal{G} is bounded. Then the minimal sufficient representation for data described by a triple $(\mathcal{O}, \mathcal{Q} = \{0\}, a_{\text{corr}} : (o, 0) \mapsto o)$ contains d latent variables.

Proof. First, we show the existence of a (sufficient) representation with d latent variables. We define the encoder mapping (and hence the representation) by $E :$

$o \mapsto \text{argmin}[H^{-1}(\{o\})] \in \mathcal{G}$, where the minimum takes into account only the first vector entry.[81] We set the decoder equal to the smooth map H . By noting that $D(E(o), 0) = o$ for all $o \in \mathcal{O}$, this shows that d latent variables are sufficient.

Let us now show that there cannot exist a representation with less than d variables. By definition of a nondegenerate function H , there exists an open subset $\mathcal{N}_d \subset \mathcal{G}$ in \mathbb{R}^d such that $H|_{\mathcal{N}_d} : \mathcal{N}_d \rightarrow H(\mathcal{N}_d)$ is a diffeomorphism. We define the function $f : o \in H(\mathcal{N}_d) \mapsto a_{\text{corr}}(o, 0) \in \mathcal{I}$, where $\mathcal{I} = H(\mathcal{N}_d)$. Since f is the identity map and hence a diffeomorphism, the data described by the triple $(\mathcal{O}, \mathcal{Q} = \{0\}, a_{\text{corr}} : (o, 0) \mapsto o)$ has dimension at least d . By Proposition 2, we conclude that at least d latent variables are required. \square

Appendix D: Neural networks

For a detailed introduction to artificial neural networks and deep learning, see for example [44]. Here we give a very short overview of the basics.

a. Single artificial neuron. The building blocks of neural networks are single neurons (Figure 4a). We can think of a neuron as a map that takes several real inputs x_1, \dots, x_n and provides an output $\sigma(\sum_i w_i x_i + b)$, according to an *activation function* $\sigma : \mathbb{R} \rightarrow \mathbb{R}$, where the weights $w_i \in \mathbb{R}$ and the bias $b \in \mathbb{R}$ are tunable parameters. The output of the neuron is itself sometimes denoted by *activation*, and there are different possible choices for the activation function. For the implementation of the examples in this paper, we use the exponential linear unit (ELU) [82], depicted in Figure 4b. The ELU is defined for a parameter $\alpha > 0$ as

$$\sigma_{\text{ELU}}(z) = \begin{cases} z & \text{for } z > 0, \\ \alpha(e^z - 1) & \text{for } z \leq 0. \end{cases}$$

b. Neural network. A (feed-forward) neural network is created by arranging neurons in layers and forwarding the outcomes of the neurons in the i -th layer to neurons in the $(i + 1)$ -th layer (see Figure 4c). The network as a whole can be viewed as a function $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with x_1, \dots, x_n corresponding to the activations of the neurons in the first layer (which is called *input layer*). The activations of the input layer form the input for the second layer, which is a *hidden layer* (since it is neither an input nor an output layer). In the case of a fully connected network, each neuron in the $(i + 1)$ -th layer receives the activations of all neurons in the i -th layer as input. The activations of the m neurons in the last layer, which is called *output layer*, are then interpreted as the output of the function F . It can be shown that neural networks are *universal*, in the sense that any continuous function can be approximated arbitrarily well by a feedforward network with just one hidden layer by using sufficiently many hidden neurons. For a mathematical statement of the result, see [83, 84]. A visualization is given in [44].

c. Training. The weights and the biases of the neural network are not tuned by hand; instead, they are optimized using training samples, i.e., known input-output-pairs $(x, F^*(x))$ of the function F^* that we would like to approximate. We may think of a neural network as a class of functions $\{F_\theta\}_\theta$, parametrized by θ , which contains the weights and biases of all the neurons in the network. A cost function $C(x, \theta)$ measures how close the output $F_\theta(x)$ of the network is to the desired output $F^*(x)$ for an input x . For example, a common choice for the cost function is $C(x, \theta) = \|F^*(x) - F_\theta(x)\|_2^2$.

The weights and biases of a network are initialized at random [44]. To then update the parameters θ , the gradient $\vec{\nabla}_\theta C(x, \theta)$ is computed and averaged over all training samples x . Subsequently, θ is updated in the negative gradient direction — hence the name *gradient descent*. In practice, the average of the gradient over all training samples is often replaced by an average over a smaller subset of training samples called a *mini-batch*; then, the algorithm is called *stochastic gradient descent*. The backpropagation algorithm is used to perform a gradient descent step efficiently (see [44] for details).

1. Variational autoencoders

The implementation of *SciNet* uses a modified version of so-called variational autoencoders (VAEs) [32, 35]. The standard VAE architecture does not include the question input used by *SciNet* and tries to reconstruct the input from the representation instead of answering a question. VAEs are one particular architecture used in the field of representation learning [30]. Here, we give a short overview over the goals of representation learning and the details of VAEs.

a. Representation learning. The goal in representation learning is to map a high-dimensional input vector x to a lower-dimensional representation $z = (z_1, z_2, \dots, z_d)$, commonly called the *latent vector*. [85] The representation z should still contain all the relevant information about x . In the case of an autoencoder, z is used to reconstruct the input x . This is motivated by the idea that the better the (low-dimensional) representation is, the better the original data can be recovered from it. Specifically, an autoencoder uses a neural network (*encoder*) to map the input x to a small number of latent neurons z . Then, another neural network (*decoder*) is used to reconstruct an estimate of the input, that is $z \mapsto \tilde{x}$. During training, the encoder and decoder are optimized to maximize the reconstruction accuracy and reach $\tilde{x} \approx x$.

b. Probabilistic encoder and decoder. Instead of considering deterministic maps $x \mapsto z$ and $z \mapsto \tilde{x}$, we generalize to conditional probability distributions $p(z|x)$ for the encoder and $p(\tilde{x}|z)$ for the decoder. This is motivated by the Bayesian view that the most informative statement the encoder can output is a description of a probability distribution over all latent vectors, instead of outputting a single estimate. The same reasoning holds for the decoder. We use the notation $z \sim p(z)$ to indicate that z is picked at random according to the distribution p .

We cannot treat the general case analytically, so we make restricting assumptions to simplify the setting. First we assume that the input can be perfectly compressed and reconstructed by an encoder and decoder which are both neural networks, that is we assume that the ideal distributions $p(z|x)$ and $p(\tilde{x}|z)$ that reach $x = \tilde{x}$ are members of parametric families $\{p_\phi(z|x)\}_\phi$ and $\{p_\theta(\tilde{x}|z)\}_\theta$, respectively. We further assume that it is possible to achieve this with a latent representation where each neuron is independent of the others, $p_\phi(z|x) = \prod_i p_{\phi_i}(z_i|x)$. If these distributions turn out hard to find for a given dimension d of the latent representation, we can try to increase the number of neurons of the representation to disentangle them. Finally, we make one more simplifying assumption, which is justified *a posteriori* by good results: that we can reach a good approximation of $p(z|x)$ by using only independent normal distributions for each latent neuron, $p_\phi(z_i|x) = \mathcal{N}(\mu_i, \sigma_i)$, where μ_i is the mean and σ_i the variance. We can think of the encoder as mapping x to the vectors $\mu = (\mu_1, \dots, \mu_d)$

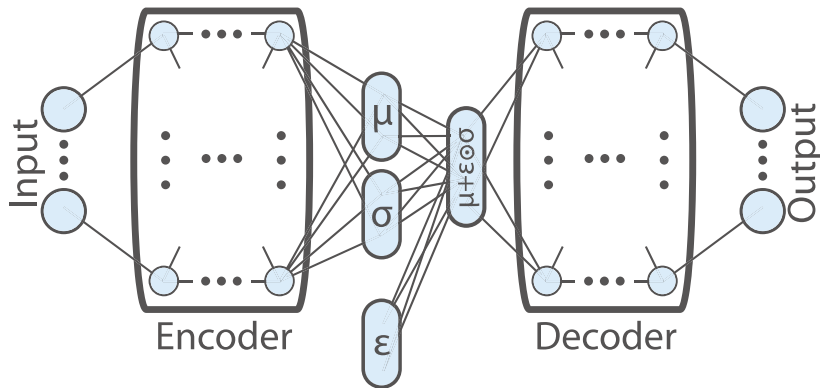


Figure 5. Network structure for a variational autoencoder. The encoder and decoder are described by conditional probability distributions $p(z|x)$ and $p(x|z)$ respectively. The output distribution of the encoder are the parameters μ_i and $\log(\sigma_i)$ for independent Gaussian distributions $z_i \sim \mathcal{N}(\mu_i, \sigma_i)$ of the latent variables. The reparameterization trick is used to sample from the latent distribution.

and $\sigma = (\sigma_1, \dots, \sigma_d)$.

The optimal settings for ϕ and θ are then learned as follows, see Figure 5:

1. The encoder with parameters (weights and biases) ϕ maps an input x to $p_\phi(z|x) = \mathcal{N}[(\mu_1, \dots, \mu_d), (\sigma_1, \dots, \sigma_d)]$.
2. A latent vector z is sampled from $p_\phi(z|x)$.
3. The decoder with parameters (weights and biases) θ maps the latent vector z to $p_\theta(\tilde{x}|z)$.
4. The parameters ϕ and θ are updated to maximize the likelihood of the original input x under the decoder distribution $p_\theta(\tilde{x}|z)$.

c. Reparameterization trick. The operation that samples a latent vector z from $p_\phi(z|x)$ is not differentiable with respect to the parameters ϕ and θ of the network. However, differentiability is necessary to train the network using stochastic gradient descent. This issue is solved by the reparameterization trick introduced in [35]: if $p_\phi(z_i|x)$ is a Gaussian with mean μ_i and standard deviation σ_i , we can replace the sampling operation using an auxiliary random number $\varepsilon_i \sim \mathcal{N}(0, 1)$. Then, a sample of the latent variable $z_i \sim \mathcal{N}(\mu_i, \sigma_i)$ can be generated by $z_i = \mu_i + \sigma_i \varepsilon_i$. Sampling ε_i does not interfere with the gradient descent because ε_i is independent of the trainable parameters ϕ and θ . Alternatively, one can view this way of sampling as injecting noise into the latent layer [76].

d. β -VAE cost function. A computationally tractable cost function for optimizing the parameters ϕ and θ was derived in [35]. This cost function was extended in [32] to encourage independency of the latent variables z_1, \dots, z_d (or to encourage “disentangled” representations, in the language of representation learning). The cost function in [32] is known as the β -VAE cost

function,

$$C_\beta(x) = - \left[\mathbb{E}_{z \sim p_\phi(z|x)} \log p_\theta(x|z) \right] + \beta D_{\text{KL}} [p_\phi(z|x) \| h(z)],$$

where the distribution $h(z)$ is a prior over the latent variables, typically chosen as the unit Gaussian[86], $\beta \geq 0$ is a constant, and D_{KL} is the Kullback-Leibler (KL) divergence, which is a quasi-distance[87] measure between probability distributions,

$$D_{\text{KL}} [p(z) \| q(z)] = \sum_z p(z) \log \left(\frac{p(z)}{q(z)} \right).$$

Let us give an intuition for the motivation behind the β -VAE cost function. The first term is a log-likelihood factor, which encourages the network to recover the input data with high accuracy. It asks “for each z , how likely are we to recover the original x after the decoding?” and takes the expectation of the logarithm of this likelihood $p_\theta(x|z)$ (other figures of merit could be used here in an alternative to the logarithm) over z sampled from $p_\phi(z|x)$, in order to simulate the encoding. In practice, this expectation is often estimated with a single sample, which works well enough if the mini-batches are chosen sufficiently large [35].

The second term encourages disentangled representations, and we can motivate it using standard properties of the KL divergence. Our goal is to minimize the amount of correlations between the latent variables z_i : we can do this by minimizing the distance $D_{\text{KL}} [p(z) \| \prod_i p(z_i)]$ between $p(z)$ and the product of its marginals. For any other distribution with independent z_i , $h(z) = \prod_i h(z_i)$, the KL divergence satisfies

$$D_{\text{KL}} \left[p(z) \| \prod_i p(z_i) \right] \leq D_{\text{KL}} [p(z) \| h(z)].$$

The KL divergence is furthermore jointly convex in its

Box 1: Time evolution of a damped pendulum (Section E 1)

Problem:: Predict the position of a one-dimensional damped pendulum at different times.

Physical model:: Equation of motion: $m\ddot{x} = -\kappa x - b\dot{x}$.

$$\text{Solution: } x(t) = A_0 e^{-\frac{b}{2m}t} \cos(\omega t + \delta_0), \text{ with } \omega = \sqrt{\frac{\kappa}{m}} \sqrt{1 - \frac{b^2}{4m\kappa}}.$$

Observation:: Time series of positions: $o = [x(t_i)]_{i \in \{1, \dots, 50\}} \in \mathbb{R}^{50}$, with equally spaced $t_i \in [0, 5]s$. Mass $m = 1\text{kg}$, amplitude $A_0 = 1\text{m}$ and phase $\delta_0 = 0$ are fixed; spring constant $\kappa \in [5, 10] \text{kg/s}^2$ and damping factor $b \in [0.5, 1] \text{kg/s}$ are varied between training samples.

Question:: Prediction times: $q = t_{\text{pred}} \in [0, 10]s$.

Correct answer:: Position at time t_{pred} : $a_{\text{corr}} = x(t_{\text{pred}}) \in \mathbb{R}$.

Implementation:: Network depicted in Figure 1 with three latent neurons (see Table II and Table I for the details).

Key findings::

- *SciNet* predicts the positions $x(t_{\text{pred}})$ with a root mean square error below 2% (with respect to the amplitude $A_0 = 1\text{m}$) (Figure 6a).
- *SciNet* stores κ and b in two of the latent neurons, and does not store any information in the third latent neuron (Figure 6b).

arguments, which implies

$$\begin{aligned} D_{\text{KL}} \left[\sum_x p(x) p_{\theta}(z|x) \| h(z) \right] \\ \leq \sum_x p(x) D_{\text{KL}} [p_{\theta}(z|x) \| h(z)]. \end{aligned}$$

Combining this with the previous inequality, we obtain

$$\begin{aligned} D_{\text{KL}} \left[p(z) \| \prod_i p(z_i) \right] \\ \leq \mathbb{E}_{x \sim p(x)} D_{\text{KL}} [p(z|x) \| h(z)]. \end{aligned}$$

The term on the right hand side corresponds exactly to the second term in the cost function, since in the training we try to minimize $\mathbb{E}_{x \sim p(x)} C_{\beta}(x)$. Choosing a large parameter β also penalizes the size of latent representation z , motivating the network to learn an efficient representation. For an empirical test of the effect of large β see [32], and for another theoretical justification using the information bottleneck approach see [76].

To derive an explicit form of C_{β} for a simple case, we again assume that $p_{\phi}(z|x) = \mathcal{N}(\mu, \sigma)$. In addition, we assume that the decoder output $p_{\theta}(\tilde{x}|z)$ is a multivariate Gaussian with mean \hat{x} and fixed covariance matrix $\hat{\sigma} = \frac{1}{\sqrt{2}} \mathbb{1}$. With these assumptions, the β -VAE cost function can be explicitly written as

$$C_{\beta}(x) = \|\hat{x} - x\|_2^2 - \frac{\beta}{2} \left(\sum_i \log(\sigma_i^2) - \mu_i^2 - \sigma_i^2 \right) + C. \quad (\text{D1})$$

The constant terms C do not contribute to the gradients used for training and can therefore be ignored.

Appendix E: Details about the physical examples

In the following, we give some more information about the four examples of physical systems to which we applied *SciNet* and which were mentioned in the main text.

1. Damped pendulum

We consider a simple example from classical physics, the damped pendulum, described in Box 1. The time evolution of the system is given by the differential equation $-\kappa x - b\dot{x} = m\ddot{x}$, where κ is the *spring constant*, which determines the frequency of the oscillation, and b is the *damping factor*. We keep the mass m constant (it is a scaling factor that could be absorbed by defining $\kappa' = \kappa/m$ and $b' = b/m$), such that κ and b are the only variable parameters. We consider the case of weak damping here, where the solution to the equation of motion is given in Box 1.

We choose a network structure for *SciNet* with 3 latent neurons. As an input, we provide a time series of positions of the pendulum and we ask *SciNet* to predict the position at a future time (see Box 1 for details). The accuracy of the predictions given by *SciNet* after training is illustrated in Figure 6a.

Without being given any physical concepts, *SciNet* learns to extract the two relevant physical parameters from (simulated) time series data for the x -coordinate of the pendulum and to store them in the latent representation. As shown in Figure 6b, the first latent neuron depends nearly linearly on b and is almost independent of κ , and the second latent neuron depends only on κ , again almost linearly. Hence, *SciNet* has recovered the same time-independent parameters b and κ that are used

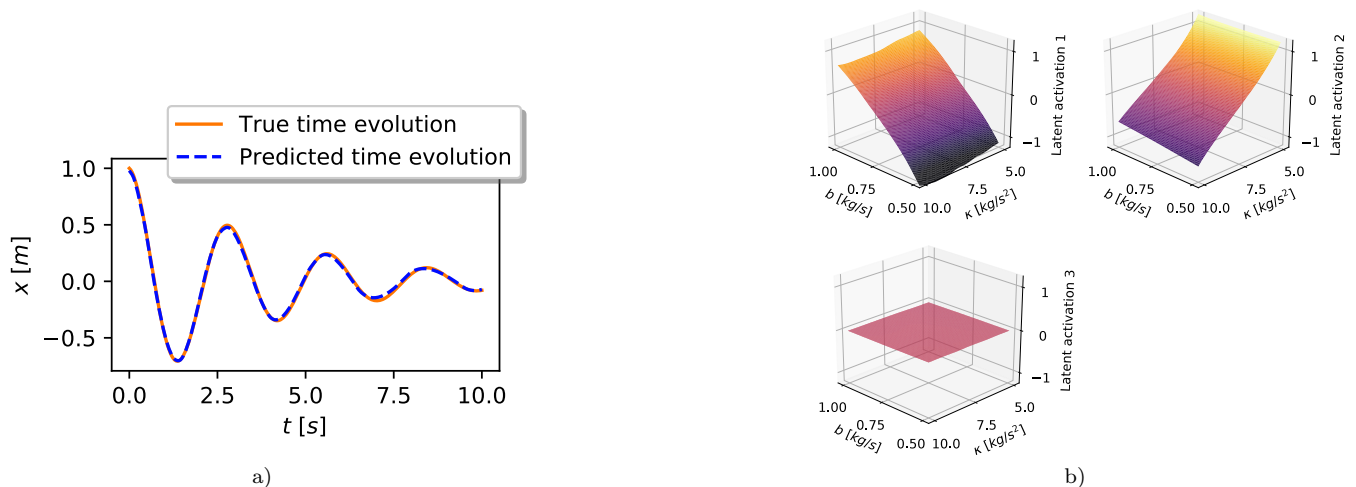


Figure 6. **Damped pendulum.** *SciNet* is fed a time series of the trajectory of a damped pendulum. It learns to store the two relevant physical parameters, frequency and damping, in the representation, and makes correct predictions about the pendulum’s future position. **(a) Trajectory prediction of *SciNet*.** Here, the spring constant is $\kappa = 5\text{kg/s}^2$ and the damping factor is $b = 0.5\text{kg/s}$. *SciNet*’s prediction is in excellent agreement with the true time evolution. **(b) Representation learned by *SciNet*.** The plots show the activations of the three latent neurons of *SciNet* as a function of the spring constant κ and the damping factor b . The first two neurons store the damping factor and spring constant, respectively. The activation of the third neuron is close to zero, suggesting that only two physical variables are required. On an abstract level, learning that one activation can be set to a constant is encouraged by searching for uncorrelated latent variables, i.e., by minimizing the common information of the latent neurons during training.

by physicists. The third latent neuron is nearly constant and does not provide any additional information — in other words, *SciNet* recognized that two parameters suffice to encode this situation.

2. Conservation of angular momentum

One of the most important concepts in physics is that of conservation laws, such as conservation of energy and angular momentum. While their relation to symmetries makes them interesting to physicists in their own right, conservation laws are also of practical importance. If two systems interact in a complex way, we can use conservation laws to predict the behaviour of one system from the behaviour of the other, without studying the details of their interaction. For certain types of questions, conserved quantities therefore act as a compressed representation of joint properties of several systems.

We consider the scattering experiment shown in Figure 7 and described in Box 2, where two point-like particles collide. Given the initial angular momentum of the two particles and the final trajectory of one of them, a physicist can predict the trajectory of the other using conservation of total angular momentum.

To see whether *SciNet* makes use of angular momentum conservation in the same way as a physicist would do, we train it with (simulated) experimental data as described in Box 2, and add Gaussian noise

to show that the encoding and decoding are robust. Indeed, *SciNet* does exactly what a physicist would do and stores the total angular momentum in the latent representation (Figure 7b). This example shows that *SciNet* can recover conservation laws, and suggests that they emerge naturally from compressing data and asking questions about joint properties of several systems.

3. Representation of qubits

Quantum state tomography is an active area of research [36]. Ideally, we look for a *faithful representation* of the state of a quantum system, such as the wave function: a representation that stores all information necessary to predict the probabilities of the outcomes for arbitrary measurements on that system. However, to specify a faithful representation of a quantum system it is not necessary to perform all theoretically possible measurements on the system. If a set of measurements is sufficient to reconstruct the full quantum state, such a set is called *tomographically complete*.

Here we show that, based only on (simulated) experimental data and without being given any assumptions about quantum theory, *SciNet* recovers a faithful representation of the state of small quantum systems and can make accurate predictions. In particular, this allows us to infer the dimension of the system and distinguish tomographically complete from incomplete measurement

Box 2: Two-body collision with angular momentum conservation (Section E 2)

Problem:: Predict the position of a particle fixed on a rod of radius r (rotating about the origin) after a collision at the point $(0, r)$ with a free particle (in two dimensions, see Figure 7a).

Physical model:: Given the total angular momentum before the collision and the velocity of the free particle after the collision, the position of the rotating particle at time t'_{pred} (after the collision) can be calculated from angular momentum conservation: $J = m_{\text{rot}}r^2\omega - rm_{\text{free}}(\mathbf{v}_{\text{free}})_x = m_{\text{rot}}r^2\omega' - rm_{\text{free}}(\mathbf{v}'_{\text{free}})_x = J'$.

Observation::

Time series of both particles before the collision: $o = [(t_i^{\text{rot}}, \mathbf{q}_{\text{rot}}(t_i^{\text{rot}})), (t_i^{\text{free}}, \mathbf{q}_{\text{free}}(t_i^{\text{free}}))]_{i \in \{1, \dots, 5\}}$, with times t_i^{rot} and t_i^{free} randomly chosen for each training sample. Masses $m_{\text{rot}} = m_{\text{free}} = 1\text{kg}$ and the orbital radius $r = 1\text{m}$ are fixed; initial angular velocity ω , initial velocity \mathbf{v}_{free} , the first component of $\mathbf{q}_{\text{free}}(0)$ and final velocity $\mathbf{v}'_{\text{free}}$ are varied between training samples. Gaussian noise ($\mu = 0, \sigma = 0.01\text{m}$) is added to all position inputs.

Question:: Prediction time and position of free particle after collision: $q = (t'_{\text{pred}}, [t'_i, \mathbf{q}'_{\text{free}}(t'_i)]_{i \in \{1, \dots, 5\}})$.

Correct answer:: Position of rotating particle at time t'_{pred} : $a_{\text{corr}} = \mathbf{q}'_{\text{rot}}(t'_{\text{pred}})$.

Implementation:: Network depicted in Figure 1 with one latent neuron (see Table II and Table I for the details).

Key findings::

- *SciNet* predicts the position of the rotating particle with root mean square prediction error below 4% (with respect to the radius $r = 1\text{m}$).
- *SciNet* is resistant to noise.
- *SciNet* stores the total angular momentum in the latent neuron.

sets. Box 3 summarizes the setting and the results.

A (pure) state on n qubits can be represented by a normalized complex vector $\psi \in \mathbb{C}^{2^n}$, where two states ψ and ψ' are identified if and only if they differ by a global phase factor, i.e., if there exists $\phi \in \mathbb{R}$ such that $\psi = e^{i\phi}\psi'$. The normalization condition and irrelevance of the global phase factor decrease the number of free parameters of a quantum state by two. Since a complex number has two real parameters, a single-qubit state is described by $2 \times 2^1 - 2 = 2$ real parameters, and a state of two qubits is described by $2 \times 2^2 - 2 = 6$ real parameters.

Here, we consider binary projective measurements on n qubits. Like states, these measurements can be described by vectors $\omega \in \mathbb{C}^{2^n}$, with measurement outcomes labeled by 0 for the projection on ω and 1 otherwise. The probability to get outcome 0 when measuring ω on a quantum system in state ψ is then given by $p(\omega, \psi) = |\langle \omega, \psi \rangle|^2$, where $\langle \cdot, \cdot \rangle$ denotes the standard scalar product on \mathbb{C}^{2^n} .

To generate the training data for *SciNet*, we assume that we have one or two qubits in a lab that can be prepared in arbitrary states and we have the ability to perform binary projective measurements in a set \mathcal{M} . We choose n_1 measurements $\mathcal{M}_1 := \{\alpha_1, \dots, \alpha_{n_1}\} \subset \mathcal{M}$ randomly, which we would like to use to determine the state of the quantum system. We perform all measurements in \mathcal{M}_1 several times on the same quantum state ψ to estimate the probabilities $p(\alpha_i, \psi)$ of measuring 0 for the i -th measurement. These probabilities form the observation given to *SciNet*.

To parameterize the measurement ω , whose outcome probabilities should be predicted by *SciNet*, we choose another random set of measurements $\mathcal{M}_2 :=$

$\{\beta_1, \dots, \beta_{n_2}\} \subset \mathcal{M}$. The probabilities $p(\beta_i, \omega)$ are provided to *SciNet* as the question input. We always assume that we have chosen enough measurements in \mathcal{M}_2 such that they can distinguish all the possible measurements $\omega \in \mathcal{M}$, i.e., we assume that \mathcal{M}_2 is tomographically complete.[88] *SciNet* then has to predict the probability $p(\omega, \psi)$ for measuring the outcome 0 on the state ψ when performing the measurement ω .

We train *SciNet* with different pairs (ω, ψ) for one and two qubits, keeping the measurement sets \mathcal{M}_1 and \mathcal{M}_2 fixed. We choose $n_1 = n_2 = 10$ for the single-qubit case and $n_1 = n_2 = 30$ for the two-qubit case. The results are shown in Figure 8.

Varying the number of latent neurons, we can observe how the quality of the predictions improves as we allow for more parameters in the representation of ψ . To minimize statistical fluctuations due to the randomized initialization of the network, each network specification is trained three times and the run with the lowest mean square prediction error on the test data is used.

For the cases where \mathcal{M}_1 is tomographically complete, the plots in Figure 8 show a drop in prediction error when the number of latent neurons is increased up to two or six for the cases of one and two qubits, respectively.[89] This is in accordance with the number of parameters required to describe a one- or a two-qubit state. Thus, *SciNet* allows us to extract the dimension of the underlying quantum system from tomographically complete measurement data, without any prior information about quantum mechanics.

SciNet can also be used to determine whether the measurement set \mathcal{M}_1 is tomographically complete or not. To

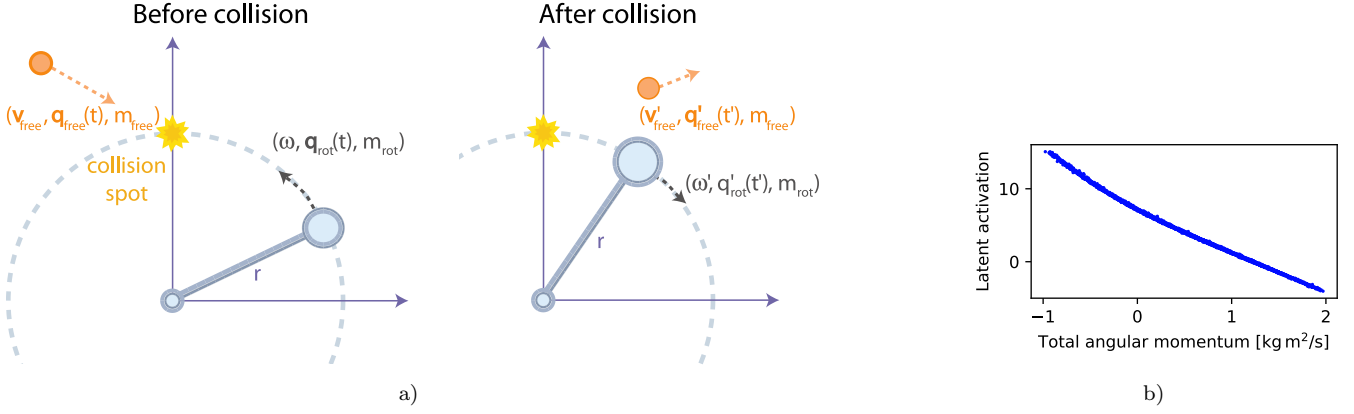


Figure 7. **Collision under conservation of angular momentum.** In a classical mechanics scenario where the total angular momentum is conserved, the neural network learns to store this quantity in the latent representation. **(a) Physical setting.** A body of mass m_{rot} is fixed on a rod of length r (and of negligible mass) and rotates around the origin with angular velocity ω . A free particle with velocity \mathbf{v}_{free} and mass m_{free} collides with the rotating body at position $\mathbf{q} = (0, r)$. After the collision, the angular velocity of the rotating particle is ω' and the free particle is deflected with velocity $\mathbf{v}'_{\text{free}}$. **(b) Representation learned by *SciNet*.** Activation of the latent neuron as a function of the total angular momentum. *SciNet* learns to store the total angular momentum, a conserved quantity of the system.

Box 3: Representation of pure one- and two-qubit states (Section E 3)

Problem:: Predict the measurement probabilities for any binary projective measurement $\omega \in \mathbb{C}^{2^n}$ on a pure n -qubit state $\psi \in \mathbb{C}^{2^n}$ for $n = 1, 2$.

Physical model:: The probability $p(\omega, \psi)$ to measure 0 on the state $\psi \in \mathbb{C}^{2^n}$ performing the measurement $\omega \in \mathbb{C}^{2^n}$ is given by $p(\omega, \psi) = |\langle \omega, \psi \rangle|^2$.

Observation:: Operational parameterization of a state ψ : $o = [p(\alpha_i, \psi)]_{i \in \{1, \dots, n_1\}}$ for a fixed set of random binary projective measurements $\mathcal{M}_1 := \{\alpha_1, \dots, \alpha_{n_1}\}$ ($n_1 = 10$ for one qubit, $n_1 = 30$ for two qubits).

Question:: Operational^F parameterization of a measurement ω : $q = [p(\beta_i, \omega)]_{i \in \{1, \dots, n_2\}}$ for a fixed set of random binary projective measurements $\mathcal{M}_2 := \{\beta_1, \dots, \beta_{n_2}\}$ ($n_2 = 10$ for one qubit, $n_2 = 30$ for two qubits).

Correct answer:: $a_{\text{corr}}(\omega, \psi) = p(\omega, \psi) = |\langle \omega, \psi \rangle|^2$.

Implementation:: Network depicted in Figure 1 with varying numbers of latent neurons (see Table II and Table I for the details).

Key findings::

- *SciNet* can be used to determine the minimal number of parameters necessary to describe the state ψ (see Figure 8) without being provided with any prior knowledge about quantum physics.
- *SciNet* distinguishes tomographically complete and incomplete sets of measurements (see Figure 8).

generate tomographically incomplete data, we choose the measurements in \mathcal{M}_1 randomly from a subset of all binary projective measurements. Specifically, the quantum states corresponding to measurements in \mathcal{M}_1 are restricted to random *real* linear superpositions of k orthogonal states, i.e., to a (real) k -dimensional subspace. For a single qubit, we use a two-dimensional subspace; for two qubits, we consider both two- and three-dimensional subspaces.

Given tomographically incomplete data about a state ψ , it is not possible for *SciNet* to predict the outcome of the final measurement perfectly regardless of the number of latent neurons, in contrast to the tomographically complete case (see Figure 8). Hence, we can deduce from

SciNet's output that \mathcal{M}_1 is an incomplete set of measurements. Furthermore, this analysis provides a qualitative measure for the amount of information provided by the tomographically incomplete measurements: in the two-qubit case, increasing the subspace dimension from two to three leads to higher prediction accuracy and the required number of latent neurons increases.

4. Heliocentric model of the solar system

All the details about this example were already given in the main text and are summarized in Box 4.

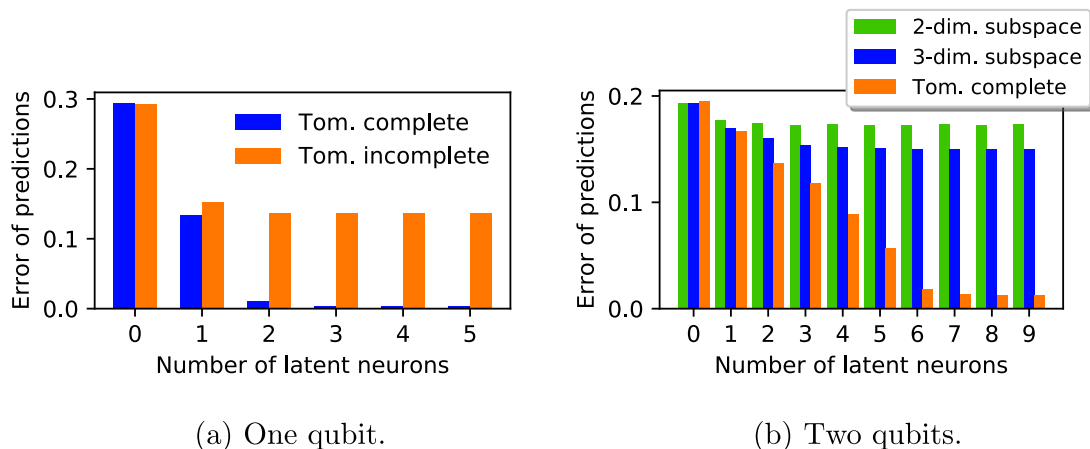


Figure 8. **Quantum tomography (more detailed plots than in the main text)**. *SciNet* is given tomographic data for one or two qubits and an operational description of a measurement as a question input and has to predict the probabilities of outcomes for this measurement. We train *SciNet* with both tomographically complete and incomplete sets of measurements, and find that, given tomographically complete data, *SciNet* can be used to find the minimal number of parameters needed to describe a quantum state (two parameters for one qubit and six parameters for two qubits). Tomographically incomplete data can be recognized, since *SciNet* cannot achieve perfect prediction accuracy in this case, and the prediction accuracy can serve as an estimate for the amount of information provided by the tomographically incomplete set. The plots show the root mean square error of *SciNet*'s measurement predictions for test data as a function of the number of latent neurons.

Box 4: Heliocentric model of the solar system (Section E 4)

Problem:: Predict the angles $\theta_M(t)$ and $\theta_S(t)$ of Mars and the Sun as seen from Earth, given initial states $\theta_M(t_0)$ and $\theta_S(t_0)$.

Physical model:: Earth and Mars orbit the Sun with constant angular velocity on (approximately) circular orbits.

Observation:: Initial angles of Mars and the Sun as seen from Earth: $o = (\theta_M(t_0), \theta_S(t_0))$, randomly chosen from a set of weekly (simulated) observations within Copernicus' lifetime (3665 observations in total).

Question:: Implicit.

Correct answer:: Time series $[a(t_1), \dots, a(t_n)] = [(\theta_M(t_1), \theta_S(t_1)), \dots, (\theta_M(t_n), \theta_S(t_n))]$ of $n = 20$ (later in training: $n = 50$) observations, with time steps $t_{i+1} - t_i$ of one week.

Implementation:: Network depicted in Figure 3a with two latent neurons and allowing for time updates of the form $r(t_{i+1}) = r(t_i) + b$ (see Table II and Table I for the details).

Key findings::

- *SciNet* predicts the angles of Mars and the Sun with a root mean square error below 0.4% (with respect to 2π).
- *SciNet* stores the angles ϕ_E and ϕ_M of the Earth and Mars as seen from the Sun in the two latent neurons (see Figure 3c).

Appendix F: Representations of cyclic parameters

Here we explain the difficulty of a neural network to learn representations of cyclic parameters, which was alluded to in the context of the qubit example (Section E 3, see [90, 91] for a detailed discussion relevant to computer vision). In general, this problem occurs if the data \mathcal{O} that we would like to represent forms a closed manifold (i.e., a compact manifold without boundary), such as a circle, a sphere or a Klein bottle. In that case, several coordinate charts are required to describe this manifold.

As an example, let us consider data points lying on the unit sphere $\mathcal{O} = \{(x, y, z) : x^2 + y^2 + z^2 = 1\}$, which we

would like to encode into a simple representation. The data can be (globally) parameterized with spherical coordinates $\phi \in [0, 2\pi)$ and $\theta \in [0, \pi]$ where $(x, y, z) = f(\theta, \phi) := (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta)$. [92] We would like the encoder to perform the mapping f^{-1} , where we define $f^{-1}((0, 0, 1)) = (0, 0)$ and $f^{-1}((0, 0, -1)) = (\pi, 0)$ for convenience. This mapping is not continuous at points on the sphere with $\phi = 0$ for $\theta \in (0, \pi)$. Therefore, using a neural network as an encoder leads to problems, as neural networks, as introduced here, can only implement continuous functions. In practice, the network is forced to approximate the discontinuity in the encoder by a very steep continuous function, which leads to a high error for

points close to the discontinuity.

In the qubit example, the same problem appears. To parameterize a qubit state ψ with two parameters, the Bloch sphere with parameters $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$ is used: the state ψ can be written as $\psi(\theta, \phi) = (\cos(\theta/2), e^{i\phi} \sin(\theta/2))$ (see for example [93] for more details). Ideally, the encoder would perform the map $E : o(\psi(\theta, \phi)) := (|\langle \alpha_1, \psi(\theta, \phi) \rangle|^2, \dots, |\langle \alpha_{N_1}, \psi(\theta, \phi) \rangle|^2) \mapsto (\theta, \phi)$ for some fixed binary projective measurements $\alpha_i \in \mathbb{C}^2$. However, such an encoder is not continuous. Indeed, assuming that the encoder is continuous, leads to the following contradiction:

$$\begin{aligned} (\theta, 0) &= E(o(\psi(\theta, \phi = 0))) \\ &= E(o(\lim_{\phi \rightarrow 2\pi} \psi(\theta, \phi))) \\ &= \lim_{\phi \rightarrow 2\pi} E(o(\psi(\theta, \phi))) \\ &= \lim_{\phi \rightarrow 2\pi} (\theta, \phi) = (\theta, 2\pi), \end{aligned}$$

where we have used the periodicity of ϕ in the second equality and the fact that the Bloch sphere representation and the scalar product (and hence $o(\psi(\theta, \phi))$) as well as the encoder (by assumption) are continuous in ϕ in the third equality.

-
- [1] S. W. Hawking, *Phys. Rev. D* **14**, 2460 (1976).
- [2] J. Preskill, *International Symposium on Black Holes, Membranes, Wormholes, and Superstrings* (1993).
- [3] V. Dunjko and H. J. Briegel, (2017), arXiv:1709.02779.
- [4] R. Roscher, B. Bohn, M. F. Duarte, and J. Garcke, (2019), arXiv:1905.08883.
- [5] I. Alhousseini, W. Chemissany, F. Kleit, and A. Nasrallah, (2019), arXiv:1905.01023.
- [6] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborov, (2019), arXiv:1903.10563.
- [7] J. P. Crutchfield and B. S. McNamara, *Complex Systems* **1**, 417 (1987).
- [8] M. Schmidt and H. Lipson, *Science* **324**, 81 (2009).
- [9] M. Schmidt, R. R. Vallabhajosyula, J. W. Jenkins, J. E. Hood, A. S. Soni, J. P. Wikswo, and H. Lipson, *Physical Biology* **8**, 055011 (2011).
- [10] B. C. Daniels and I. Nemenman, *Nature Communications* **6**, 8133 (2015).
- [11] S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Proceedings of the National Academy of Sciences* **113**, 3932 (2016).
- [12] B. Lusch, J. N. Kutz, and S. L. Brunton, (2017), arXiv:1712.09707.
- [13] N. Takeishi, Y. Kawahara, and T. Yairi, “Learning Koopman invariant subspaces for dynamic mode decomposition,” (2017), arXiv:1710.04340.
- [14] S. E. Otto and C. W. Rowley, (2017), arXiv:1712.01378.
- [15] M. Raissi, (2018), arXiv:1801.06637.
- [16] D. Zhang, L. Guo, and G. E. Karniadakis, (2019), arXiv:1905.01205.
- [17] M. Raissi and G. E. Karniadakis, *Journal of Computational Physics* **357**, 125 (2018).
- [18] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Journal of Computational Physics* **378**, 686 (2019).
- [19] C. Bates, I. Yildirim, J. B. Tenenbaum, and P. W. Battaglia, *Proceedings of the 37th Annual Conference of the Cognitive Science Society, Pasadena, CA*, 172 (2015).
- [20] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum, in *Advances in Neural Information Processing Systems 28*, edited by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Curran Associates, Inc., 2015) pp. 127–135.
- [21] N. R. Bramley, T. Gerstenberg, J. B. Tenenbaum, and T. M. Gureckis, *Cognitive Psychology* **105**, 9 (2018).
- [22] D. Rempe, S. Sridhar, H. Wang, and L. J. Guibas, (2019), arXiv:1901.00466.
- [23] M. Kissner and H. Mayer, (2019), arXiv:1905.09891.
- [24] S. Ehrhardt, A. Monszpart, N. Mitra, and A. Vedaldi, (2018), arXiv:1805.05086.
- [25] T. Ye, X. Wang, J. Davidson, and A. Gupta, (2018), arXiv:1808.10002.
- [26] D. Zheng, V. Luo, J. Wu, and J. B. Tenenbaum, (2018), arXiv:1807.09244.
- [27] T. Wu and M. Tegmark, (2018), arXiv:1810.10525.
- [28] A. De Simone and T. Jacques, *The European Physical Journal C* **79**, 289 (2019).
- [29] R. T. D’Agnolo and A. Wulzer, *Physical Review D* **99**, 015014 (2019).
- [30] Y. Bengio, A. Courville, and P. Vincent, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**, 1798 (2013).
- [31] G. E. Hinton and R. R. Salakhutdinov, *Science* **313**, 504 (2006).
- [32] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, *ICLR* (2017).
- [33] S. M. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, and D. Hassabis, *Science* **360**, 1204 (2018).
- [34] K. Hornik, *Neural Networks* **4**, 251 (1991).
- [35] D. P. Kingma and M. Welling, (2013), arXiv:1312.6114.
- [36] M. Paris and J. Reháček, eds., *Quantum State Estimation*, *Lecture Notes in Physics* (Springer, Berlin, Heidelberg, 2004).
- [37] In the case of a single qubit, there is an additional small improvement in going from two to three latent neurons: this is a technical issue caused by the fact that (finite size) neural networks cannot represent discontinuous functions (see Appendix F). The same likely applies in the case of two qubits, going from 6 to 7 latent neurons.
- [38] For a general system, there might not exist a representation that admits such a simple time evolution. In such a case, one may have to define a complexity measure for the time update rule and search over different rules succes-

- sively increasing the complexity until *SciNet* can achieve good prediction accuracy.
- [39] J. R. Koza, *Statistics and Computing* **4**, 87 (1994).
- [40] M. Krenn, F. Häse, A. Nigam, P. Friederich, and A. Aspuru-Guzik, (2019), arXiv:1905.13741.
- [41] C. Bény, (2019), arXiv:1904.10387.
- [42] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” (2015).
- [43] https://github.com/eth-nn-physics/nn.physical.concepts/blob/copernicus/analysis/copernicus_analysis.ipynb.
- [44] M. A. Nielsen, “Neural networks and deep learning,” (2018).
- [45] Y. LeCun, Y. Bengio, and G. Hinton, *Nature* **521**, 436 (2015).
- [46] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, *Nature* **529**, 484 (2016).
- [47] A. Decelle, V. Martin-Mayor, and B. Seoane, (2019), arXiv:1904.07637.
- [48] M. Krenn, M. Malik, R. Fickler, R. Lapkiewicz, and A. Zeilinger, *Phys. Rev. Lett.* **116**, 090405 (2016).
- [49] A. A. Melnikov, H. P. Nautrup, M. Krenn, V. Dunjko, M. Tiersch, A. Zeilinger, and H. J. Briegel, *Proceedings of the National Academy of Sciences* **115**, 1221 (2018).
- [50] H. J. Briegel and G. D. I. Cuevas, *Scientific Reports* **2**, 400 (2012).
- [51] G. Carleo and M. Troyer, *Science* **355**, 602 (2017).
- [52] A. Rocchetto, E. Grant, S. Strelchuk, G. Carleo, and S. Severini, *npj Quantum Information* **4**, 28 (2018).
- [53] I. Glasser, N. Pancotti, M. August, I. D. Rodriguez, and J. I. Cirac, *Physical Review X* **8**, 011006 (2018).
- [54] G. Carleo, Y. Nomura, and M. Imada, *Nature Communications* **9**, 5322 (2018).
- [55] Z. Cai and J. Liu, *Physical Review B* **97** (2018), 10.1103/PhysRevB.97.035116.
- [56] Y. Huang and J. E. Moore, (2017), arXiv:1701.06246.
- [57] D.-L. Deng, X. Li, and S. D. Sarma, *Physical Review B* **96** (2017), 10.1103/PhysRevB.96.195145.
- [58] M. Schmitt and M. Heyl, *SciPost Physics* **4** (2018), 10.21468/SciPostPhys.4.2.013.
- [59] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, *Nature Physics* **14**, 447 (2018).
- [60] Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, *Physical Review B* **96** (2017), 10.1103/PhysRevB.96.205152.
- [61] D.-L. Deng, X. Li, and S. D. Sarma, *Physical Review X* **7** (2017), 10.1103/PhysRevX.7.021021.
- [62] X. Gao and L.-M. Duan, *Nature Communications* **8** (2017), 10.1038/s41467-017-00705-2.
- [63] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, *Nature Machine Intelligence* **1**, 155 (2019).
- [64] M. J. S. Beach, I. De Vlucht, A. Golubeva, P. Huembeli, B. Kulchitsky, X. Luo, R. G. Melko, E. Merali, and G. Torlai, *SciPost Physics* **7**, 009 (2019).
- [65] G. Torlai and R. G. Melko, (2019), arXiv:1905.04312.
- [66] M. Koch-Janusz and Z. Ringel, *Nature Physics* **14**, 578 (2018).
- [67] J. Hamrick, P. Battaglia, and J. B. Tenenbaum, *In Cognitive Science Society*, 1545 (2011).
- [68] P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum, *Proceedings of the National Academy of Sciences* **110**, 18327 (2013).
- [69] T. D. Ullman, A. Stuhlmüller, N. D. Goodman, and J. B. Tenenbaum, *Cognitive Psychology* **104**, 57 (2018).
- [70] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, and K. Kavukcuoglu, *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 4509 (2016).
- [71] M. B. Chang, T. Ullman, A. Torralba, and J. B. Tenenbaum, (2016), arXiv:1612.00341.
- [72] D. Raposo, A. Santoro, D. Barrett, R. Pascanu, T. Lillicrap, and P. Battaglia, (2017), arXiv:1702.05068.
- [73] T. S. Cubitt, J. Eisert, and M. M. Wolf, *Physical Review Letters* **108**, 120503 (2012).
- [74] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther, in *Advances in Neural Information Processing Systems 30*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc., 2017) pp. 3601–3610.
- [75] Y. Bengio, (2013), 1305.0445.
- [76] A. Achille and S. Soatto, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **40**, 2897 (2018).
- [77] N. Tishby, F. C. Pereira, and W. Bialek, (2000), arXiv:physics/0004057.
- [78] N. Tishby and N. Zaslavsky, *2015 IEEE Information Theory Workshop (ITW)*, 1 (2015).
- [79] R. Shwartz-Ziv and N. Tishby, (2017), arXiv:1703.00810.
- [80] J. Lee, *Introduction to Smooth Manifolds*, 2nd ed., Graduate Texts in Mathematics (Springer-Verlag, New York, 2012).
- [81] Note that any element in $H^{-1}(\{o\})$ could be chosen.
- [82] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, (2015), arXiv:1511.07289.
- [83] G. Cybenko, *Mathematics of Control, Signals and Systems* **2**, 303 (1989).
- [84] K. Hornik, M. Stinchcombe, and H. White, *Neural Networks* **2**, 359 (1989).
- [85] The variables x and z correspond to the observation o and the representation r used in the main text.
- [86] The interpretation of $h(z)$ as a prior is clear only when deriving VAEs as generative networks. For details, see [35].
- [87] The KL divergence satisfies all axioms of a metric apart from symmetry.
- [88] This parameterization of a measurement ω assumes that we know the equivalence between binary projective measurements and states. However, this is not a fundamental assumption, since we could parameterize the set of possible measurements by any parameterization that is natural for the experimental setup, for example the settings of the dials and buttons on an experimental apparatus. Such a natural parameterization is assumed to fully specify the measurement, in the sense that the same settings on the experimental apparatus will always result in the same measurement being performed. Because \mathcal{M}_2 only represents our choice for parameterizing the mea-

surement setup, it is natural to assume that \mathcal{M}_2 is tomographically complete.

- [89] In the case of a single qubit, there is an additional small improvement in going from two to three latent neurons: this is a technical issue caused by the fact that any two-parameter representation of a single qubit, for example the Bloch sphere representation, includes a *cyclic parameter*, which cannot be exactly represented by a continuous encoder (see the supplementary materials F). The same likely applies in the case of two qubits, going from 6 to 7 latent neurons. This restriction also makes it difficult to

interpret the details of the learned representation.

- [90] N. Pitelis, C. Russell, and L. Agapito, IEEE Conference on Computer Vision and Pattern Recognition , 1642 (2013).
- [91] E. O. Korman, (2018), arXiv:1803.00156.
- [92] The function f is not a chart, since it is not injective and its domain is not open.
- [93] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, 2010).