Neural network perturbation theory and its application to the Born series

Bastian Kaspschak [1] and Ulf-G. Meißner [1,2,3]

[1] *Helmholtz-Institut für Strahlen- und Kernphysik and Bethe Center for Theoretical Physics, Universität Bonn, D-53115 Bonn, Germany*

[2] *Institute for Advanced Simulation, Institut für Kernphysik, and Jülich Center for Hadron Physics, Forschungszentrum Jülich, D-52425 Jülich, Germany*

[3] *Tbilisi State University, 0186 Tbilisi, Georgia*

(Received 9 September 2020; accepted 26 May 2021; published 21 June 2021)

Deep learning using the eponymous deep neural networks (DNNs) has become an attractive approach towards various data-based problems of theoretical physics in the past decade. There has been a clear trend to deeper architectures containing increasingly more powerful and involved layers. Contrarily, Taylor coefficients of DNNs still appear mainly in the light of interpretability studies, where they are computed at most to first order. However, especially in theoretical physics, numerous problems benefit from accessing higher orders, as well. This gap motivates a general formulation of neural network (NN) Taylor expansions. Restricting our analysis to multilayer perceptrons (MLPs) and introducing quantities we refer to as propagators and vertices, both depending on the MLP's weights and biases, we establish a graph-theoretical approach. Similarly to Feynman rules in quantum field theories, we can systematically assign diagrams containing propagators and vertices to the corresponding partial derivative. Examining this approach for S-wave scattering lengths of shallow potentials, we observe NNs to adapt their derivatives mainly to the leading order of the target function's Taylor expansion. To circumvent this problem, we propose an iterative NN perturbation theory. During each iteration we eliminate the leading order, such that the next-to-leading order can be faithfully learned during the subsequent iteration. After performing two iterations, we find that the first- and second-order Born terms are correctly adapted during the respective iterations. Finally, we combine both results to find a proxy that acts as a machine-learned second-order Born approximation.

DOI: 10.1103/PhysRevResearch.3.023223

I. INTRODUCTION

Machine learning (ML) is a highly active field of research that provides a wide range of tools to tackle various data-based problems. As such, it has also received growing attention in the theoretical physics literature, such as in Refs. [1–15]. Many data-based problems involve modeling an input-target distribution from a data set, which is referred to as supervised learning. After a successful training procedure, the ML algorithm is capable of correctly predicting targets, even when given previously unknown inputs, i.e., it generalizes what it has learned to new data. Nowadays, neural networks (NNs) are a popular choice in the context of supervised learning. There is an overwhelming variety of NN architectures that are as diverse as the problems they are specially suited for. The certainly most fundamental class of NNs is given by multilayer perceptrons (MLPs). Many obvious properties of state-of-the-art NNs such as the concept of a layered architecture or the use of nonlinear activation functions originate in much simpler MLP architectures.

The property that makes NNs perform so well in many different applications is that of being a universal approximator: As long as the architecture comprises an output layer and at least one hidden layer that is activated via a bounded, nonlinear activation function, the NN can approximate any continuous map between inputs and targets arbitrarily precise for a sufficiently large number of neurons in that hidden layer, as described by the universal approximation theorem (UAT); see Refs. [16,17]. However, increasing the number of neurons in one layer is a rather inefficient way to improve the NN's performance. It is more promising to introduce additional nonlinearly activated hidden layers, instead, which eventually opens up the field of deep learning. Here, the term "deep" refers to a large number of such nonlinearly activated layers. Its protagonists, the deep neural networks (DNNs), are known for their enormous predictive power and for demonstrating superhuman performances for specific tasks. Last but not least, this makes them a promising approach towards problems of theoretical physics, as shown, e.g., in Refs. [1–3].

While there has clearly been a trend towards deeper and more complex architectures in the past decade, capable of approximating increasingly involved target functions, the interest in the analytical properties of NNs remains limited to

interpretability studies. Notable methods used to gather *post hoc* interpretations of NN predictions are the deep Taylor decomposition and local interpretable model-agnostic explanations (LIME); see Refs. [18] and [19]. The former is used exclusively for image classifiers: Given a root point of the classifier, a heat map can be constructed using the NN's first-order derivatives, assigning to each pixel a certain relevance value. This highlights those pixels that are substantially involved in the resulting decision. Similarly, LIME performs a linear regression of adjacent synthetic samples and thus provides a linear approximation, or in the language of Ref. [20] a linear proxy, of the target function in the vicinity of the root point. Both methods, indeed, facilitate powerful local *post hoc* interpretations of the respective NN's behavior in feature space. However, they do not provide any information about second- or higher-order derivatives and are, therefore, blind to most of the NN's analytical structure. This contrasts greatly with the fact that a majority of problems in theoretical physics certainly benefit from having access not only to the NN's first-order derivatives but also ideally to their entire analytical structure. Obvious examples that come to mind are the *post hoc* verification of equations of motion or, which the present study focuses on, the extraction of dominating terms from an underlying perturbation theory.

Closing the mentioned gap requires a general method to compute NN derivatives of arbitrary order. On one hand, a numerical differentiation is certainly no difficult task, but it may be rendered inaccurate due to truncation and roundoff errors and does not reveal the contribution of the individual weights and biases to a particular order. On the other hand, a naive analytical differentiation of NN predictions does not share these weaknesses but will suffer from an unmanageable amount of different contributions, especially for high-order derivatives and a large number of hidden layers. Restricting the analysis for simplicity to MLPs, we propose a graph-theoretical formalism to analytically compute partial derivatives of any order for arbitrarily many hidden layers, while keeping track of the combinatorics. Similarly to backpropagation in gradient-descent techniques, where the first-order derivatives of the loss function with respect to an internal parameter can be represented as a matrix product, we want to bypass the naive and inefficient use of the chain and product rules and understand arbitrary derivatives of an MLP in terms of tensor products. We observe two distinct classes of quantities, which we refer to as *propagators* and *vertices*, that each depend on the weights, biases, and chosen activation functions and naturally appear in such a tensor formulation. Their naming is intentional, as we discover several similarities between the Taylor expansion of MLPs and perturbation theory in quantum field theories: Analogously to Feynman rules, we find underlying rules that specify which combinations of vertices and propagators, i.e., which diagrams, are allowed and contribute to a given Taylor coefficient. One major difference, however, is that loops are not allowed in contrast to quantum field theories. In a graph-theoretical context, we can show that these diagrams are oriented and rooted trees, i.e., arborescences. The concept of explaining derivatives in terms of graphs is already known and is a successful approach in the context of ordinary differential equations or, more precisely, the Butcher series; see Ref. [21]. In contrast to the Butcher series, however, we clearly want to set our focus on Taylor expansions and perturbation theory.

Due to its simplicity and ubiquity in quantum physics, two-body scattering appears to be an adequate field for studying the adaptation of MLPs to perturbation theories. In fact, the present study is strongly motivated by the weight inspections performed in Ref. [11]: Considering the first hidden layer of MLPs trained to predict S-wave scattering lengths for shallow, attractive potentials of finite range, it is shown that weights among all of its active neurons satisfy a quadratic pattern $w_{nm} \propto m^2$. This can be proven to reproduce the first-order Born approximation. As soon as MLPs are trained on successively deeper potentials, additional structures emerge within their weights, which are later identified with the second-order Born term. This qualitatively indicates that during training, MLPs adapt to the Born series and thus develop a quantum perturbation theory. Applying the proposed graph-theoretical formalism, we complement these findings by a quantitative investigation of the dominating analytical structure of MLP ensembles that predict S-wave scattering lengths. Since we observe these MLPs to mainly adapt their derivatives to the leading order, we develop an iterative scheme that can be understood as an NN perturbation theory to successively obtain remaining terms of the target function's Taylor expansion: At each iteration, the idea is to eliminate the leading order from the current targets in the training and test sets, which generates new data sets for the next iteration, which a new auxiliary ensemble of MLPs can be trained on. A downside of this approach is that each iteration requires that one run an additional training pipeline. However, the dominating contributions of the auxiliary ensembles are significantly more faithful to the corresponding terms of target functions' Taylor expansion than a differentiation of a single, naively trained ensemble could provide. The first- and second-order Taylor coefficients found this way can be identified one to one with the first- and second-order Born term, respectively. These two results are then combined to a machine-learned second-order Born approximation, which performs well for shallow potentials. Finally, these indicate that our NN perturbation theory naturally translates to a perturbation theory for scattering lengths.

This paper is organized as follows: At first, Sec. II introduces the S-wave scattering length as a functional, briefly presents the first two variational derivatives, and relates them to the Born series in quantum two-body scattering. When approximated by NNs, their sampled form gives rise to understanding the Born series as a Taylor series in the space of sampled potentials. In Sec. III, we then propose a graphically motivated approach to compute partial derivatives of arbitrarily deep MLPs in terms of propagators and vertices. Section IV builds on these findings, develops the NN perturbation theory we finally want to apply to MLPs trained on S-wave scattering lengths, and sheds light on the training pipeline as well as on architecture details. The first-order Born term is evaluated and discussed in Sec. V, followed by investigation of the second-order Born term after one iteration in Sec. VI. We end with a discussion and outlook in Sec. VII. Various technicalities are relegated to Appendix.

II. THE BORN SERIES AS A TAYLOR SERIES

Let $\boldsymbol{Y} : \mathbb{R}^{H_0} \to \mathbb{R}^{H_L}$ be an analytical function. The local behavior of $\boldsymbol{Y}$ in the vicinity of an arbitrary point $\boldsymbol{x}_0 \in \mathbb{R}^{H_0}$ can be described by neglecting higher-order terms in the Taylor expansion

$$Y_n(\boldsymbol{x}) - Y_n(\boldsymbol{x}_0) = \sum_{k=1}^{H_0} \frac{\partial Y_n}{\partial x_k}\bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} (\boldsymbol{x}-\boldsymbol{x}_0)_k + \frac{1}{2}\sum_{k_1=1}^{H_0}\sum_{k_2=1}^{H_0} \frac{\partial^2 Y_n}{\partial x_{k_1}\partial x_{k_2}}\bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} (\boldsymbol{x}-\boldsymbol{x}_0)_{k_1}(\boldsymbol{x}-\boldsymbol{x}_0)_{k_2} + \cdots . \tag{1}$$

In the following, we interpret the components $x_k = f(k/H_0)$ of any given vector $\boldsymbol{x} \in \mathbb{R}^{H_0}$ as samples of an analytical function $f \in C^\omega([0,1])$. In this context, higher dimensions H_0 correspond to higher sampling rates. Note that $\boldsymbol{x}$ and f become totally equivalent in the limit of an infinitely fine sampling rate; that is, $H_0 \to \infty$. In this case the given function $\boldsymbol{Y}$ can rather be understood as a functional $\boldsymbol{Y} : C^\omega([0,1]) \to \mathbb{R}^{H_L}$. Consequently, the above expression generalizes to an expansion around a given function $f_0 \in C^\omega([0,1])$,

$$\begin{aligned} Y_n[f] - Y_n[f_0] = &\int_0^1 dk\, \frac{\delta Y_n[f]}{\delta f(k)}\bigg|_{f=f_0} \{f(k) - f_0(k)\} \\ &+ \frac{1}{2}\int_0^1 dk_1 \int_0^1 dk_2\, \frac{\delta^2 Y_n[f]}{\delta f(k_1)\delta f(k_2)}\bigg|_{f=f_0} \{f(k_1) - f_0(k_1)\}\{f(k_2) - f_0(k_2)\} + \cdots . \end{aligned}$$

In this limit it is no longer the partial derivatives $\partial^N \boldsymbol{Y}/\partial x_{k_1}\cdots\partial x_{k_N}|_{\boldsymbol{x}=\boldsymbol{x}_0}$, but the variational derivatives $\delta^N \boldsymbol{Y}[f]/\delta f(k_1)\cdots\delta f(k_N)|_{f=f_0}$ that parametrize the local behavior of $\boldsymbol{Y}$. Sampling the latter yields partial derivatives and again reproduces the multidimensional Taylor expansion in Eq. (1). An example we study thoroughly in Secs. V and VI is the functional that maps an attractive, dimensionless potential $U = 2\mu\rho^2 V$ with finite range ρ to the corresponding dimensionless S-wave scattering length in units of ρ,

$$\begin{aligned} a_0[U] &= \frac{2\pi^2}{\rho^3}\langle 0|T|0\rangle = \frac{2\pi^2}{\rho^3}\langle 0|U|0\rangle + \frac{\pi^2}{\rho^6}\langle 0|UG_0U|0\rangle + \cdots \\ &= \int_0^1 dr\, r^2 U(r) - \frac{1}{2}\int_0^1 dr_1 \int_0^1 dr_2\, r_1 r_2 (r_1 + r_2 - |r_1 - r_2|)U(r_1)U(r_2) + \cdots . \end{aligned} \tag{2}$$

Here, the quantities μ, V, T, and G_0 denote the reduced mass of the two-body system, the dimensionful potential, and the dimensionless T matrix and resolvent, respectively. Equation (2) not only contains the expansion of a_0 around the force-free case $U = 0$ but also displays the classical representation of the S-wave scattering length as the Born series and therefore suggests that we treat a_0 perturbatively for shallow potentials. The two lowest-order variational derivatives,

$$\frac{\delta a_0[U]}{\delta U(r)}\bigg|_{U=0} = r^2, \qquad \frac{\delta^2 a_0[U]}{\delta U(r_1)\delta U(r_2)}\bigg|_{U=0} = -r_1 r_2 (r_1 + r_2 - |r_1 - r_2|),$$

can then be used to compute the first- and second-order Born approximation of a_0, respectively. Consequently, their sampled versions are given by

$$\frac{\partial a_0}{\partial U_k}\bigg|_{\boldsymbol{U}=\boldsymbol{0}} = \frac{k^2}{(H_0)^3}, \qquad \frac{\partial^2 a_0}{\partial U_{k_1}\partial U_{k_2}}\bigg|_{\boldsymbol{U}=\boldsymbol{0}} = -\frac{1}{(H_0)^5} k_1 k_2 (k_1 + k_2 - |k_1 - k_2|). \tag{3}$$

Equations (1) and (3) give rise to understanding the Born series in the space of sampled potentials as a Taylor series. Thereby, each sampled potential $\boldsymbol{U}$ corresponds to an H_0-dimensional vector with components $U_k = U(r = k/H_0)$. It is obvious that the discretization error becomes negligibly small for sufficiently high sampling rates H_0. Now $\boldsymbol{Y}$ could serve as a target function that we try to imitate by an NN. In the context of the above example this means that the NN can successfully predict S-wave scattering lengths for sampled potentials after completing a supervised training procedure. According to the UAT, these predictions can be arbitrarily precise, provided that the given architecture contains sufficiently many neurons or is sufficiently deep. Nonetheless, the UAT in no way guarantees that the NN also reproduces the analytical properties, i.e., the target function's partial derivatives at each order. A pathological, but obvious example is NNs with Heaviside activations: Here, the derivatives at any order can only take the values 0 or $\pm\infty$, which can be realized as a superposition of delta functions.

At this point the following questions arise: What conditions must the given architecture satisfy such that loss minimization during training also causes the partial derivatives of the MLP for any given order to approximate the corresponding derivatives of the target function? Asked differently, If we are given a raw data set and do not know the analytical representation of the target function, how can we be sure that its analytical structure is reproduced by the trained NN? What is the benefit of having the NN approximate the analytical structure of the target function? How can MLP and target function derivatives be compared with each other analytically in the first place?

Of course, in order to comply with the assumed analyticity of the target function, the activation functions themselves need

to be analytical. As many pooling layers such as maximum (max) pooling or rectifiers such as the rectified linear unit (ReLU) are not everywhere differentiable, this already excludes a wide range of conventional architectures. Also, note that this analyticity criterion is just a necessary and not a sufficient condition for the NN in order to approximate the analytical structure of the target function. To give an example, the Gaussian error linear unit (GELU) is an analytical rectifier that serves as an activation function later in this paper. While lower-order derivatives are unproblematic, higher-order derivatives vanish almost everywhere and diverge for a small range of inputs, which renders them highly unstable. Also, there is no *ad hoc* guarantee that the NN approximates all lower-order derivatives of the target function equally well: If the training set only covers a narrow range of inputs around the expansion point, the NN may tend to approximate only the leading order. While we will observe this issue for the numerically stable first- and second-order derivatives of the Born series in Secs. V and VI, a further investigation for higher-order derivatives is beyond the scope of the present study and, therefore, left for future studies.

III. PARTIAL DERIVATIVES OF MULTILAYER PERCEPTRONS

An MLP with L layers is the prototype example of a layered architecture and can be understood as a nonlinear function $\mathcal{Y}: \mathbb{R}^{H_0} \to \mathbb{R}^{H_L}$ between real vector spaces. The term "layered" describes that $\mathcal{Y}$ is a composition $\mathcal{Y} = \mathcal{Y}_L \circ \cdots \circ \mathcal{Y}_1$ of L layers $\mathcal{Y}_l: \mathbb{R}^{H_{l-1}} \to \mathbb{R}^{H_l}$ containing H_l neurons $z_n^{(l)}$. In MLPs, exclusively linear layers are used in combination with nonlinear activation functions $a^{(l,n)}: \mathbb{R} \to \mathbb{R}$, where $a^{(l,n)}$ is applied to the nth neuron of the lth layer. This can be formulated recursively,

$$\mathcal{Y}_n(\boldsymbol{x}) = y_n^{(L)}, \tag{4}$$

with the recursive step

$$y_n^{(l)} = a^{(l,n)}\big(z_n^{(l)}\big), \quad z_n^{(l)} = \sum_{m=1}^{H_{l-1}} w_{nm}^{(l)} y_m^{(l-1)} + b_n^{(l)}, \tag{5}$$

together with the weights $w_{nm}^{(l)}$ and biases $b_n^{(l)}$. The recursion in Eq. (5) is terminated for $l = 1$ due to reaching its base $y_m^{(0)} = x_m$. For deep architectures, i.e., for $L \gg 1$, $\mathcal{Y}$ is a strongly nested function, such that computing derivatives becomes an extremely difficult task because of a hardly manageable amount of chain- and product-rule applications. In fact, there is another field of machine learning in which it is well known how to efficiently compute first-order partial derivatives of a strongly nested function: Within gradient-descent-based training algorithms it is necessary to compute the gradient of a loss function, which is an error function of the network $\mathcal{Y}$ and therefore nested to the same extent. Here, the first-order partial derivatives of the loss function with respect to any internal parameter can be expressed by a matrix product. This is the famous backpropagation which significantly speeds up training steps by avoiding naively applying chain and product rules; see Ref. [22].

In order to derive Taylor coefficients of $\mathcal{Y}$ of any order and for an arbitrary number L of layers in terms of the weights and biases, we desire a systematic description in the spirit of backpropagation. Let us therefore at first define

$$D_{nm}^{(l,p)} = w_{nm}^{(l+1)} \frac{d^p a^{(l,m)}}{\mathrm{d}x^p}\big(z_m^{(l)}\big), \tag{6}$$

which we refer to as the nmth matrix element of the lth-layer *propagator* of order p. Since the last layer is usually activated via the identity, $a^{(L,n)} = \mathrm{id}$, and has no bias, i.e., $b_n^{(L)} = 0$, we can write

$$\mathcal{Y}_n(\boldsymbol{x}) = \sum_{m=1}^{H_{L-1}} D_{nm}^{(L-1,0)}. \tag{7}$$

This redefinition entirely describes outputs in terms of propagators and reduces the search for Taylor coefficients to computing partial derivatives of propagator matrix elements,

$$\frac{\partial^N \mathcal{Y}_n}{\partial x_{k_1} \cdots \partial x_{k_N}} = \sum_{m=1}^{H_{L-1}} \frac{\partial^N D_{nm}^{(L-1,0)}}{\partial x_{k_1} \cdots \partial x_{k_N}}, \tag{8}$$

$$\frac{\partial D_{nm}^{(l,p)}}{\partial x_k} = D_{nm}^{(l,p+1)} \Delta_{mk}^{(l,1)},$$

$$\Delta_{mk}^{(l,1)} = \frac{\partial z_m^{(l)}}{\partial x_k} = \sum_{q_l=1}^{H_l} \cdots \sum_{q_1=1}^{H_1} \delta_{mq_l} w_{q_1 k}^{(1)} \prod_{i=1}^{l-1} D_{q_{i+1} q_i}^{(i,1)}. \tag{9}$$

In Eq. (9) we make two observations: First, a derivation increases the order of the propagator by 1. Second, an additional factor $\Delta_{mk}^{(l,1)}$ is introduced, which impacts higher-order derivatives of propagators. We introduce the tensor elements

$$\Delta_{mk_1 \cdots k_p}^{(l,p)} = \frac{\partial z_m^{(l)}}{\partial x_{k_1} \cdots \partial x_{k_p}}, \tag{10}$$

which are obviously invariant under permutations of the indices $k_1, \ldots, k_p$. Now we can express the Nth derivative of the propagator as the following superposition by successively applying the rule mentioned in Eq. (9) and by absorbing the remaining derivatives by Eq. (10) (see Appendix A 2),

$$\frac{\partial^N D_{nm}^{(l,p)}}{\partial x_{k_1} \cdots \partial x_{k_N}} = \sum_{c=1}^{N} D_{nm}^{(l,p+c)} \sum_{\sigma \in \mathcal{S}_N} \sum_{\boldsymbol{\pi} \in \Pi_N^c} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \prod_{i=1}^{c} \Delta_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)} \cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}^{(l,\pi_i)}. \tag{11}$$

Each summand in Eq. (11) includes a higher-order propagator. Here, the second sum runs over the set of all partitions of the number N with length c,

$$\Pi_N^c = \left\{ \boldsymbol{\pi} \in \mathbb{N}^c \,\middle|\, \sum_{i=1}^{c} \pi_i = N \wedge (\pi_1 \geqslant \cdots \geqslant \pi_c) \right\},$$

$$\Pi_N = \bigcup_{c=1}^{N} \Pi_N^c. \tag{12}$$

The set Π_N of all partitions is, thereby, simply given by the union of all Π_N^c. Lastly, the third sum runs over the per-

TABLE I. All partitions $\boldsymbol{\pi} \in \Pi_N$ and corresponding factors $\varepsilon_{\boldsymbol{\pi}}$ for $N = 1, 2, 3, 4$ required for computing the propagator derivatives $\partial^N D^{(l,p)}_{nm}/(\partial x_{k_1} \cdots \partial x_{k_N})$ using Eq. (11). Note that the index permutation symmetry of the $\Delta^{(l,p)}_{mk_1\cdots k_p}$ has been used to combine $\varepsilon_{\boldsymbol{\pi}}$ summands. Thus there are only $N!/\varepsilon_{\boldsymbol{\pi}}$ remaining summands for each partition $\boldsymbol{\pi}$.

N	$\boldsymbol{\pi} \in \Pi_N$	$\varepsilon_{\boldsymbol{\pi}}$	$\frac{\partial^N D^{(l,p)}_{nm}}{\partial x_{k_1} \cdots \partial x_{k_N}}$
1	(1)	1	$D^{(l,p+1)}_{nm}\Delta^{(l,1)}_{mk_1}$
2	(1,1)	2	$D^{(l,p+2)}_{nm}\Delta^{(l,1)}_{mk_1}\Delta^{(l,1)}_{mk_2}$
2	(2)	2	$+D^{(l,p+1)}_{nm}\Delta^{(l,2)}_{mk_1k_2}$
3	(1,1,1)	6	$D^{(l,p+3)}_{nm}\Delta^{(l,1)}_{mk_1}\Delta^{(l,1)}_{mk_2}\Delta^{(l,1)}_{mk_3}$
3	(2,1)	2	$+D^{(l,p+2)}_{nm}\left(\Delta^{(l,2)}_{mk_1k_2}\Delta^{(l,1)}_{mk_3}+\Delta^{(l,2)}_{mk_1k_3}\Delta^{(l,1)}_{mk_2}+\Delta^{(l,2)}_{mk_2k_3}\Delta^{(l,1)}_{mk_1}\right)$
3	(3)	6	$+D^{(l,p+1)}_{nm}\Delta^{(l,3)}_{mk_1k_2k_3}$
4	(1,1,1,1)	24	$D^{(l,p+4)}_{nm}\Delta^{(l,1)}_{mk_1}\Delta^{(l,1)}_{mk_2}\Delta^{(l,1)}_{mk_3}\Delta^{(l,1)}_{mk_4}$
4	(2,1,1)	4	$+D^{(l,p+3)}_{nm}\left(\Delta^{(l,2)}_{mk_1k_2}\Delta^{(l,1)}_{mk_3}\Delta^{(l,1)}_{mk_4}+\Delta^{(l,2)}_{mk_1k_3}\Delta^{(l,1)}_{mk_2}\Delta^{(l,1)}_{mk_4}+\Delta^{(l,2)}_{mk_1k_4}\Delta^{(l,1)}_{mk_2}\Delta^{(l,1)}_{mk_3}+\right.$ $\left.+\Delta^{(l,2)}_{mk_2k_3}\Delta^{(l,1)}_{mk_1}\Delta^{(l,1)}_{mk_4}+\Delta^{(l,2)}_{mk_2k_4}\Delta^{(l,1)}_{mk_1}\Delta^{(l,1)}_{mk_3}+\Delta^{(l,2)}_{mk_3k_4}\Delta^{(l,1)}_{mk_1}\Delta^{(l,1)}_{mk_2}\right)$
4	(2,2)	8	$+D^{(l,p+2)}_{nm}\left(\Delta^{(l,2)}_{mk_1k_2}\Delta^{(l,2)}_{mk_3k_4}+\Delta^{(l,2)}_{mk_1k_3}\Delta^{(l,2)}_{mk_2k_4}+\Delta^{(l,2)}_{mk_1k_4}\Delta^{(l,2)}_{mk_2k_3}\right)$
4	(3,1)	6	$+D^{(l,p+2)}_{nm}\left(\Delta^{(l,3)}_{mk_1k_2k_3}\Delta^{(l,1)}_{mk_4}+\Delta^{(l,3)}_{mk_1k_2k_4}\Delta^{(l,1)}_{mk_3}+\Delta^{(l,3)}_{mk_1k_3k_4}\Delta^{(l,1)}_{mk_2}+\Delta^{(l,3)}_{mk_2k_3k_4}\Delta^{(l,1)}_{mk_1}\right)$
4	(4)	24	$+D^{(l,p+1)}_{nm}\Delta^{(l,4)}_{mk_1k_2k_3k_4}$

mutation group $\mathcal{S}_N$. For a given partition $\boldsymbol{\pi} \in \Pi^c_n$, the factor $1/\varepsilon_{\boldsymbol{\pi}}$ takes the symmetry of the tensor elements $\Delta^{(l,p)}_{mk_{\sigma(1)}\cdots k_{\sigma(p)}} = \Delta^{(l,p)}_{mk_1\cdots k_p}$ under index permutations $\sigma \in \mathcal{S}_N$ into account and can be derived via

$$\varepsilon_{\boldsymbol{\pi}} = \prod_{i=1}^{c}(\pi_i)!\left[\left(\sum_{j=1}^{c}\delta_{\pi_i\pi_j}\right)!\right]^{1/\sum_{j=1}^{c}\delta_{\pi_i\pi_j}}. \tag{13}$$

Table I contains all partitions, respective $\varepsilon_{\boldsymbol{\pi}}$, and resulting propagator derivatives for $N = 1, 2, 3, 4$. What remains is to find an expression of the tensors $\Delta^{(l,p)}_{mk_1\cdots k_p}$ in terms of propagators such that we can completely determine the partial derivatives of the propagators in Eq. (11). We therefore introduce the mkth matrix element of a vertex of order p in the lth layer, acting as a weighted sum over elements of arbitrary tensors f,

$$\bigOmega_{\substack{mk\\(q_a)_{a=1}^{l}(j_b)_{b=1}^{p}}} f_{q_{j_1}\ldots q_{j_p}} = \sum_{q_l}^{H_l}\cdots\sum_{q_1}^{H_1}\delta_{mq_l}w^{(1)}_{q_1k}\sum_{j_1=1}^{l-1}\sum_{\substack{j_2=1\\j_2\neq j_1}}^{l-1}\cdots\sum_{\substack{j_p=1\\j_p\neq j_1\\\vdots\\j_p\neq j_{p-1}}}^{l-1}\left(\prod_{\substack{i=1\\i\neq j_1\\\vdots\\i\neq j_p}}^{l-1}D^{(i,1)}_{q_{i+1}q_i}\right)f_{q_{j_1}\ldots q_{j_p}}. \tag{14}$$

If $l-1 \leqslant p$, the vertex becomes saturated; that is, it becomes a constant and is equal to any higher-order vertex in the same layer. Because of this and due to Eq. (9), vertices display the following behavior when exposed to a partial derivative:

$$\begin{aligned}\frac{\partial}{\partial x_{k_2}}\bigOmega_{\substack{mk_1\\(q_a)_{a=1}^{l}(j_b)_{b=1}^{p-1}}} f_{q_{j_1}\ldots q_{j_{p-1}}} &= \Theta(l-p)\bigOmega_{\substack{mk_1\\(q_a)_{a=1}^{l}(j_b)_{b=1}^{p}}} f_{q_{j_1}\ldots q_{j_{p-1}}}D^{(j_p,2)}_{q_{j_{p+1}}q_{j_p}}\bigOmega_{\substack{q_{j_p}k_2\\(q'_a)_{a=1}^{j_p}}}\\ &+\bigOmega_{\substack{mk_1\\(q_a)_{a=1}^{l}(j_b)_{b=1}^{p-1}}}\frac{\partial}{\partial x_{k_2}}f_{q_{j_1}\ldots q_{j_{p-1}}}.\end{aligned} \tag{15}$$

Obviously, vertices of order p in the lth layer only commute with partial derivatives if they are saturated. This is embodied by the proportionality of the commutator to the step function with $\Theta(0) = 0$. Note that we have expressed $\Delta^{(l,1)}_{mk}$ as a vertex of order zero in order to arrive at Eq. (15),

$$\Delta^{(l,1)}_{mk_1} = \bigOmega_{\substack{mk_1\\(q_a)_{a=1}^{l}}} = \Theta(l-0)\left(\bullet\right)^{(l,1)}_{mk_1}, \tag{16}$$

for which we choose the graphical representation of a single vertex. Applying Eq. (15) to Eq. (16) yields

$$\begin{aligned}\Delta^{(l,2)}_{mk_1k_2} &= \Theta(l-1) \bigsum_{(q_a)_{a=1}^{l}(j_1)} D^{(j_1,2)}_{q_{j_1+1}q_{j_1}} \bigsum_{(q'_a)_{a=1}^{j_1}} \\ &= \Theta(l-1)\Theta(l-0) \bigsum_{(q_a)_{a=1}^{l}(j_1)} D^{(j_1,2)}_{q_{j_1+1}q_{j_1}} \left(\bullet\right)^{(j_1,1)}_{q_{j_1}k_2} \\ &= \Theta(l-1) \left(\bullet\!\!\rightarrow\!\!\bullet\right)^{(l,2)}_{mk_1k_2} .\end{aligned} \tag{17}$$

This term depends on a first-order vertex that sums over a second-order propagator and a zeroth-order vertex, which suggests the graphical representation of two vertices that are connected via a propagator with one arrowhead, directed from the first to the second vertex. Note that the index permutation symmetry of the $\Delta^{(l,p)}_{mk_1\cdots k_p}$ is inherited by the right-hand side of Eq. (17). Applying Eq. (15) once more to Eq. (17) yields

$$\begin{aligned}\Delta^{(l,3)}_{mk_1k_2k_3} &= \Theta(l-2) \bigsum_{(q_a)_{a=1}^{l}(j_1,j_2)} D^{(j_1,2)}_{q_{j_1+1}q_{j_1}} D^{(j_2,2)}_{q_{j_2+1}q_{j_2}} \bigsum_{(q'_a)_{a=1}^{j_1}} \bigsum_{(q''_a)_{a=1}^{j_2}} \\ &+ \Theta(l-1) \bigsum_{(q_a)_{a=1}^{l}(j_1)} D^{(j_1,3)}_{q_{j_1+1}q_{j_1}} \bigsum_{(q'_a)_{a=1}^{j_1}} \bigsum_{(q''_a)_{a=1}^{j_1}} \\ &+ \Theta(l-1) \bigsum_{(q_a)_{a=1}^{l}(j_1)} D^{(j_1,2)}_{q_{j_1+1}q_{j_1}} \bigsum_{(q'_a)_{a=1}^{j_1}(j'_1)} D^{(j'_1,2)}_{q'_{j'_1+1}q_{j'_1}} \bigsum_{(q''_a)_{a=1}^{j_1}} ,\end{aligned}$$

which can be graphically represented as

$$\Delta^{(l,3)}_{mk_1k_2k_3} = \Theta(l-2) \left(\;\right)^{(l,3)}_{mk_1k_2k_3} + \Theta(l-1) \left(\;\right)^{(l,3)}_{mk_1k_2k_3} + \Theta(l-1) \left(\;\right)^{(l,3)}_{mk_1k_2k_3} . \tag{18}$$

This is a superposition of three different terms, to each of which we can assign a different graph. Note that the second term only contains one propagator of third order instead of two second-order propagators as in both other terms. Given a vertex, we decide to enumerate outgoing propagators by the number of their arrowheads. If there are n outgoing edges with the same number of arrowheads, as is the case in the second term with $n = 2$, they represent the same propagator of order $n + 1$.

It would not be of much use to continue with successively deriving higher-order tensors as above. The general idea of how the $\Delta^{(l,N)}_{mk_1\cdots k_N}$ are structured and how the individual terms can be translated into graphs should be clear:

(i) $\Delta^{(l,N)}_{mk_1\cdots k_N}$ can be represented by a sum of directed and rooted trees, i.e., arborescences; see Ref. [23]. Each arborescence consists of N vertices and up to $N - 1$ propagators.

(ii) At each vertex, outgoing propagators are counted by the number of arrowheads on the respective edges. Therefore an edge with n arrowheads belongs to the nth propagator originating at that vertex.

(iii) If there are n edges with the same number of arrowheads originating in a given vertex, these represent a propagator of order $n + 1$ originating in that vertex. Consequently, this propagator establishes connections to n other vertices.

(iv) A term of the structure

$$\cdots \bigsum_{(q_a^{\{1\}})_{a=1}^{j_1^{\{0\}}}(j_a^{\{1\}})_{a=1}^{p_1}} D^{(j_b^{\{1\}},n)}_{q^{\{1\}}_{j_b^{\{1\}}+1}q_{j_b^{\{1\}}}} \cdots \bigsum_{(q_a^{\{2\}})_{a=1}^{j_a^{\{1\}}}(j_a^{\{2\}})_{a=1}^{p_2}} \cdots \bigsum_{(q_a^{\{n\}})_{a=1}^{j_a^{\{1\}}}(j_a^{\{n\}})_{a=1}^{p_n}} \cdots$$

indicates that it is the bth propagator originating in the k_{x_1}th vertex that establishes a connection to the k_{x_2}th, ..., $k_{x_{n-1}}$th and k_{x_n}th vertex. This implies that this propagator is of order n. If no propagator is originating in a vertex, that vertex is called a leaf of the given arborescence.

(v) Derivatives of saturated vertices vanish, as seen in Eq. (15). Thus whether a certain arborescence contributes or not depends on the layer l for which propagators and vertices are considered. This is embodied by multiplying a factor $\Theta(l - \alpha)$ to each arborescence. The given arborescence only

contributes if l overshoots its saturation threshold, which we denote by α. The appearance of internal vertices and propagators of order 3 or higher decreases α.

In order to pursue a more systematical approach, we want to understand an arborescence in terms of an adjacency matrix $A \in \mathbb{N}_0^{N\times N}$ containing information about which of the N vertices are connected by which propagators. At this point, it is useful to recall the propagator numbering based on arrowheads, which has been introduced above: If $A_{ij} = 0$, there is no connection from the ith to the jth vertex. Otherwise, it is the A_{ij}th propagator originating in the ith vertex that establishes this connection. Alternatively, A_{ij} can be understood as the number of arrowheads on the edge directed from the ith to the jth vertex. Due to orientation, each allowed adjacency matrix is an upper triangular matrix with a vanishing main diagonal. Since all vertices but the first one have exactly one incoming propagator, there is exactly one nonzero entry in each but the first column of A. The set of such triangular matrices over $\mathbb{K}$ is given by

$$\mathbb{T}_{\mathbb{K}}^{N} = \big\{ M \in \mathbb{K}^{N\times N} \big| \forall i \in \{1,\dots,N\} \forall j \in \{1,\dots,i\} : M_{ij} = 0 \times \wedge\big(N > 1 \Rightarrow \forall j \in \{2,\dots,N\} \exists^{=1} i \in \{1,\dots,N-1\} : M_{ij} \neq 0\big)\big\}.$$

Then the set of all allowed adjacency matrices for N vertices is the following subset of $\mathbb{T}_{\mathbb{N}_0}^{N}$:

$$\mathbb{A}^{N} = \big\{ A \in \mathbb{T}_{\mathbb{N}_0}^{N} \big| \forall i \in \{1,\dots,N-1\} \forall j \in \{2,\dots,N\} : (A_{ij} > 0 \Rightarrow \exists j' < j : A_{ij'} = A_{ij} - 1)\big\}. \tag{19}$$

Counting the appearances of $A_{ij} > 0$ in the ith line of a given adjacency matrix $A \in \mathbb{A}^N$ determines the order of the respective propagator: If A_{ij} appears $n-1$ times in the ith line, the corresponding propagator is of order n. Note that the appearance of a propagator of order n decreases the saturation threshold $\alpha(A)$ by $n-2$. Another source that leads to smaller $\alpha(A)$ is internal vertices: $\alpha(A)$ is decreased by 1 for each internal vertex, or, in terms of adjacency matrices, for each nonzero line but the first one. This behavior is entirely described by the simple expression

$$\alpha(A) = \max_{i,j} A_{ij}. \tag{20}$$

If l reaches or undershoots $\alpha(A)$, the corresponding arborescence is saturated and does not contribute to $\Delta^{(l,N)}_{mk_1\cdots k_N}$. Table II lists example arborescences and corresponding adjacency matrices A as well as saturation thresholds $\alpha(A)$. Note that there may be several allowed adjacency matrices for one arborescence due to index permutation symmetry. The only thing left for expressing $\Delta^{(l,N)}_{mk_1\cdots k_N}$ solely in terms of propagators and biases is an analytical representation $\delta^{(l,N)}_{mk_1\cdots k_N}(A)$ of an individual arborescence with N vertices for a given adjacency matrix A. For its formulation, we use the function

$$\beta_c(A) = \sum_{i=1}^{N} i \cdot \Theta(A_{ic}), \tag{21}$$

which determines the line in which the entry in the jth column is nonzero. That means it provides the unique vertex, from which a propagator leads to the cth vertex. Since there is no antecedent propagator to the root of an arborescence, Eq. (21) vanishes for $c = 1$. Using Eq. (21), the abbreviations

$$j_c(A) = j^{\{\beta_c(A)\}}_{A_{\beta_c(A)\ c}}, \quad q_c(A) = q^{\beta_c(A)}_{j_c(A)},$$

$$n_{bc}(A) = \sum_{j=1}^{N} \delta_{b\,A_{cj}}, \quad D^{(p)}_{bc}(A) = D^{(j_b^{\{c\}},p)}_{q^{\{c\}}_{j_b^{\{c\}}+1}\, q^{\{c\}}_{j_b^{\{c\}}}},$$

and the previous observations, we write

$$\delta^{(l,N)}_{mk_1\dots k_N}(A) = \sum_{q_l^{\{0\}}} \delta_{mq_l^{\{0\}}} \sum_{j^{\{0\}}_{A_{01}}} \delta_{lj^{\{0\}}_{A_{01}}} \prod_{c=1}^{N} \Omega_{q_c(A)\ k_c} \prod_{b=1}^{\max_r(A_{cr})} D_{bc}^{(1+n_{bc}(A))}(A). \tag{22}$$
$$(q_a^{\{c\}})_{a=1}^{j_c(A)}\ (j_b^{\{c\}})_{b=1}^{\max_r(A_{cr})}$$

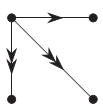
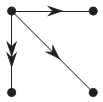
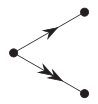
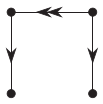
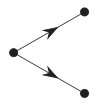
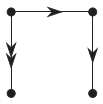
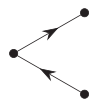
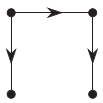
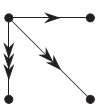
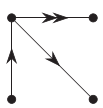
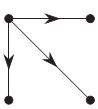
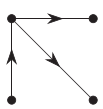
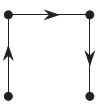
Equation (22) may appear very intimidating at first sight, but its individual terms are easy to interpret and to recognize in the summands of Eqs. (16)–(18):

(i) The expression $\max_r(A_{cr})$ corresponds to the number of all propagators originating in the cth vertex. Thus all these propagators are collected by the product

$$\prod_{b=1}^{\max_r(A_{cr})} D_{bc}^{(1+n_{bc}(A))}(A).$$

The order of the bth propagator originating in the cth vertex is equal to 1 plus the total number $n_{bc}(A)$ of appearances of the entry b in the cth line of A. If the cth vertex is a leaf of the arborescence, i.e., if it is external such that there are no outgoing propagators and $\max_r(A_{cr}) = 0$, the product can be neglected.

TABLE II. All arborescences and corresponding adjacency matrices $A \in \mathbb{A}^N$ as well as saturation thresholds $\alpha(A)$ for $N = 1, 2, 3, 4$ vertices. In the way arborescences are represented here, it is always the vertex k_1 that serves as the root. From here, we can deduce that the maximum saturation threshold among all arborescences with N vertices is $N - 1$.

MLP arborescence	A	$\alpha(A)$	MLP arborescence	A	$\alpha(A)$
	$\begin{pmatrix} 0 \end{pmatrix}$	0		$\begin{pmatrix} 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	2
	$\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$	1		$\begin{pmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	2
	$\begin{pmatrix} 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	2		$\begin{pmatrix} 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	2
	$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	1		$\begin{pmatrix} 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	2
	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$	1		$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	1
	$\begin{pmatrix} 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	3		$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	2
	$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	1		$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	1
	$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	1			

(ii) The cth vertex of the arborescence is denoted by the expression

$$\Omega_{\substack{q_c(A)\ k_c \\ (q_a^{\{c\}})_{a=1}^{j_c(A)}\ (j_b^{\{c\}})_{b=1}^{\max_r(A_{cr})}}}.$$

For each of the $\max_r(A_{cr})$ propagators that originate in the cth vertex, there is a summation index $j_b^{\{c\}}$. It is the $\beta_c(A)$th vertex, whose $A_{\beta_c(A)c}$th propagator leads to the cth vertex. This is the reason why we sum over the $(q_c(A), k_c)$th matrix element of the cth vertex in the $j_c(A)$th layer.

(iii) All N vertices of the entire arborescence are collected by

$$\sum_{q_l^{\{0\}}} \delta_{mq_l^{\{0\}}} \sum_{j_{A_{01}}^{\{0\}}} \delta_{l j_{A_{01}}^{\{0\}}} \prod_{c=1}^{N}.$$

Using Eq. (22) and taking the corresponding saturation thresholds given in Eq. (20) into account, we can represent $\Delta^{(l,N)}_{mk_1\cdots k_N}$ as the following sum over all adjacency matrices A in $\mathbb{A}^N$ (see Appendix A 3):

$$\Delta^{(l,N)}_{mk_1\cdots k_N} = \sum_{A\in\mathbb{A}^N} \Theta[l - \alpha(A)]\, \delta^{(l,N)}_{mk_1\cdots k_N}(A). \tag{23}$$

Finally, Eq. (23) can be inserted into Eq. (11), which is required for computing the Taylor coefficients of $\boldsymbol{\mathcal{Y}}$ as shown in Eq. (8). Considering Eq. (11), it is important to note that none of the indices $k_1, \ldots, k_N$ is shared among several factors $\Delta^{(l,\pi_i)}$. Therefore each summand of $\partial^N D^{(l,p)}_{nm}/(\partial x_{k_1} \cdots \partial x_{k_N})$ consists of arborescences that each contain different vertices. The effective saturation threshold of a product of several $\Delta^{(l,\pi_i)}$ is, thereby, simply the maximal saturation threshold among all given factors. The Taylor coefficients up to third order then turn out as

$$\left.\frac{\partial \mathcal{Y}_n}{\partial x_{k_1}}\right|_{\boldsymbol{x}=\boldsymbol{x}_0} = \sum_{m=1}^{H_{L-1}} D^{(L-1,1)}_{nm}\Theta(L-1)\left(\bullet\right)^{(L-1,1)}_{mk_1}\Bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} \tag{24}$$

$$\left.\frac{\partial^2 \mathcal{Y}_n}{\partial x_{k_1}\,\partial x_{k_2}}\right|_{\boldsymbol{x}=\boldsymbol{x}_0} = \sum_{m=1}^{H_{L-1}} \Bigg[D^{(L-1,1)}_{nm}\Theta(L-2)\left(\bullet\!\!\rightarrow\!\!\bullet\right)^{(L-1,2)}_{mk_1k_2} + D^{(L-1,2)}_{nm}\Theta(L-1)\left(\bullet\right)^{(L-1,1)}_{mk_1}\left(\bullet\right)^{(L-1,1)}_{mk_2}\Bigg]\Bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} \tag{25}$$

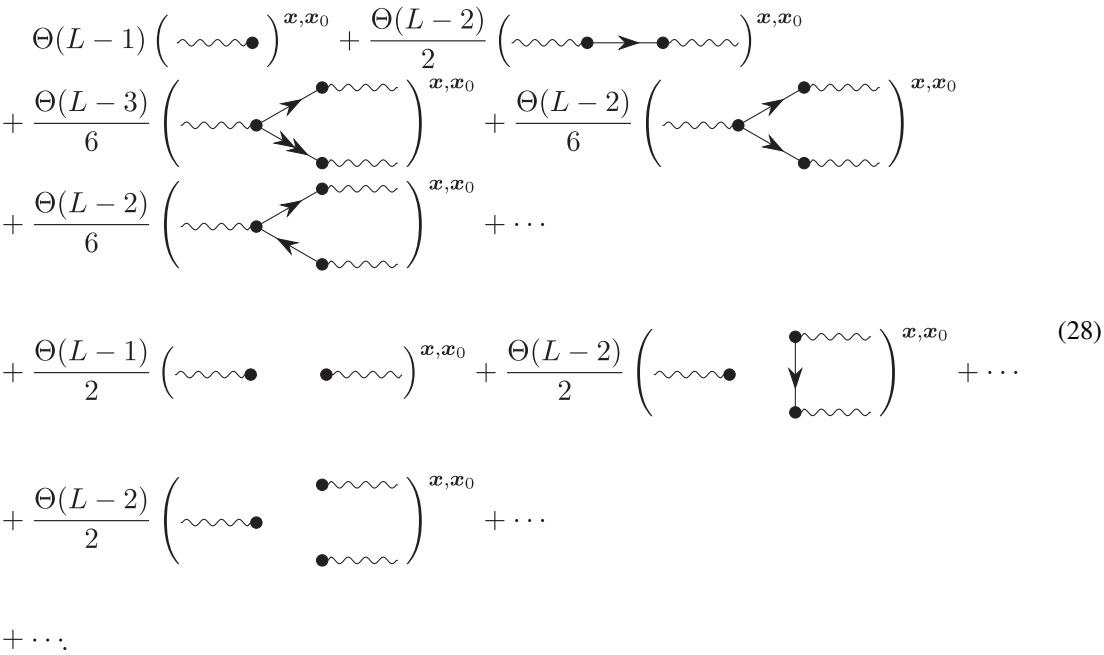
$$\left.\frac{\partial^3 \mathcal{Y}_n}{\partial x_{k_1}\,\partial x_{k_2}\,\partial x_{k_3}}\right|_{\boldsymbol{x}=\boldsymbol{x}_0} = \sum_{m=1}^{H_{L-1}} \Bigg\{ D^{(L-1,1)}_{nm}\Bigg[\Theta(L-3)\left(\begin{array}{c}\bullet\\ \bullet\!\!\rightarrow\!\!\bullet\end{array}\right)^{(L-1,3)}_{mk_1k_2k_3} + \Theta(L-2)\left(\begin{array}{c}\bullet\\ \bullet\!\!\rightarrow\!\!\bullet\end{array}\right)^{(L-1,3)}_{mk_1k_2k_3} + \Theta(L-2)\left(\begin{array}{c}\bullet\\ \bullet\!\!\rightarrow\!\!\bullet\end{array}\right)^{(L-1,3)}_{mk_1k_2k_3}\Bigg] + D^{(L-1,2)}_{nm}\Bigg[\Theta(L-2)\,(\bullet)^{(L-1,1)}_{mk_1}\left(\begin{array}{c}\bullet\\ \downarrow\\ \bullet\end{array}\right)^{(L-1,2)}_{mk_2k_3} + \Theta(L-2)\,(\bullet)^{(L-1,1)}_{mk_2}\left(\begin{array}{c}\bullet\\ \downarrow\\ \bullet\end{array}\right)^{(L-1,2)}_{mk_1k_3} + \Theta(L-2)\,(\bullet)^{(L-1,1)}_{mk_3}\left(\begin{array}{c}\bullet\\ \downarrow\\ \bullet\end{array}\right)^{(L-1,2)}_{mk_1k_2}\Bigg] + D^{(L-1,3)}_{nm}\;\Theta(L-1)\;\;(\bullet)^{(L-1,1)}_{mk_1}(\bullet)^{(L-1,1)}_{mk_2}(\bullet)^{(L-1,1)}_{mk_3}\Bigg\}\Bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} \tag{26}$$

Let $I_j = \{i^j_1, \ldots, i^j_{y_j}\} \subseteq \{1, \ldots, N\}$ with $j \in \{1, \ldots, p\}$ be pairwise disjoint index sets such that $I_1 \cup \cdots \cup I_p = \{1, \ldots, N\}$ and $I_{j_1} \cap I_{j_2} = \emptyset$ for $j_1 \neq j_2$. This implies $y_1 + \cdots + y_p = N$. Given these indices, the following short-hand notation proves useful for the Taylor series, since it helps to combine p arborescences covering N vertices to one disconnected graph with p connected components:

$$\left(\begin{array}{c}\vdots\\ \cdots\\ \vdots\end{array}\right)^{\boldsymbol{x},\boldsymbol{x}_0}_{n} \cdots \left(\begin{array}{c}\vdots\\ \cdots\\ \vdots\end{array}\right) = \sum_{m=1}^{H_{L-1}} D^{(L-1,p)}_{nm} \sum_{k_1}^{H_0} \cdots \sum_{k_N}^{H_0} \left(\begin{array}{c}\vdots\\ \cdots\\ \vdots\end{array}\right)^{(L-1,y_1)}_{mk_{i^1_1}\ldots k_{i^1_{y_1}}} \cdots \left(\begin{array}{c}\vdots\\ \cdots\\ \vdots\end{array}\right)^{(L-1,y_p)}_{mk_{i^p_1}\ldots k_{i^p_{y_1}}} \Bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} \prod_{i=1}^{N} (\boldsymbol{x}-\boldsymbol{x}_0)_{k_i}. \tag{27}$$

The resulting graph is only connected for $p = 1$. Using the Taylor coefficients from Eqs. (24)–(26) and the short-hand notation from Eq. (27) finally yields the interesting series representation, which is ordered by the number of connected components:

$$\mathcal{Y}(\boldsymbol{x}) - \mathcal{Y}(\boldsymbol{x}_0) =$$



(28)

Graphs that contain up to N vertices contribute to the Nth Taylor approximation of $\mathcal{Y}$. The deeper $\mathcal{Y}$, i.e., the larger L, the more graphs contribute to a given order of the Taylor expansion due to overshooting the corresponding saturation threshold.

IV. NN PERTURBATION THEORY APPLIED TO SCATTERING LENGTHS

Depending on the analytical structure of the target function, there may be a difference of several orders of magnitude between the Taylor coefficients of different orders. For example, if the first-order Taylor coefficients dominate and if all training samples are closely distributed around the expansion point, then a trained MLP basically applies a linear approximation to imitate the target function. As a supervisor one does not gain any insights into higher-order terms in such cases, since these presumably are not faithful to the derivatives of the target function. As can be seen in Eq. (3), the first- and second-order Taylor coefficients of the sampled Born series vary by a factor $1/(H_0^2) \approx 10^{-3}$. The sampling rate H_0 must be sufficiently large such that the discretization error becomes negligible, which we assume to be the case for $H_0 = 32$, as used in the following analysis. For the specific application, this implies that we cannot expect to recover both the first- and second-order Born terms from a naively trained MLP or ensemble. Therefore we propose an iterative scheme to gain information on Taylor coefficients of successively rising order. Given a training set $T_1^{(0)}$ and test set $T_2^{(0)}$ and assuming we do not train single MLPs, but ensembles of several MLPs, the ith iteration contains the following steps:

(1) Initialize a new ensemble $\mathcal{Y}^{(i)} = \{\mathcal{Y}_n^{(i)}\}_{n=1}^{N_i}$ of MLPs $\mathcal{Y}_n^{(i)}$. The output of the ensemble is simply the mean of the individual member outputs.

(2) Train the ensemble $\mathcal{Y}^{(i)}$ on the training set $T_1^{(i-1)}$ and validate it later using the test set $T_2^{(i-1)}$.

(3) Compute the ith-order term $1/(N_i i!) \sum_n \sum_{k_1} \cdots \sum_{k_i} (\partial^i \mathcal{Y}_n^{(i)})/(\partial x_{k_1} \cdots \partial x_{k_i})|_{\boldsymbol{x}=\boldsymbol{x}_0} \prod_j (\boldsymbol{x} - \boldsymbol{x}_0)_{k_j}$ of the Taylor expansion of $\sum_n \mathcal{Y}_n^{(i)}/N_i$ around the expansion point $\boldsymbol{x}_0$. As this term is of leading order, the corresponding ith-order derivatives and, therefore, Taylor coefficients can be assumed to be faithful to the analytical structure of the target function.

(4) Generate new training and test sets $T_1^{(i)}$ and $T_2^{(i)}$ by subtracting the leading-order term of the previous step from the targets of $T_1^{(i-1)}$ and $T_2^{(i-1)}$, respectively. If necessary, the targets must be normalized or standardized again.

At the cost of rerunning the training pipeline for each iteration anew, we especially expect this procedure to yield faithful first- and second-order Taylor coefficients in the case of S-wave scattering lengths. Note that such an iterative approach is anything but unnatural and is really just the central idea of perturbation theory.

At first we generate training and test sets of shallow potentials without any bound states. The scattering lengths for sampled potentials are derived using the transfer matrix method (see Ref. [24]) and are uniformly distributed between the boundaries $a_0 = -1$ and $a_0 = 0$. The training and test sets contain $|T_1^{(0)}| = 3 \times 10^4$ and $|T_2^{(0)}| = 3 \times 10^3$ samples, respectively.

Training and validation of the ensembles at each iteration are performed in PYTORCH [25]. At first, we initialize an ensemble $\mathcal{Y}^{(1)}$ of $N_1 = 10^2$ MLPs, in which each but the output layer is activated via the GELU activation function [26]. GELU is smooth in the origin in contrast to other rectifiers such as ReLU. Being a rectifier, it bypasses the vanishing-gradients problem, which makes it particularly interesting for deeper architectures; see Ref. [27]. The weights and biases of the ensemble are initialized using He initialization [28]. Apart

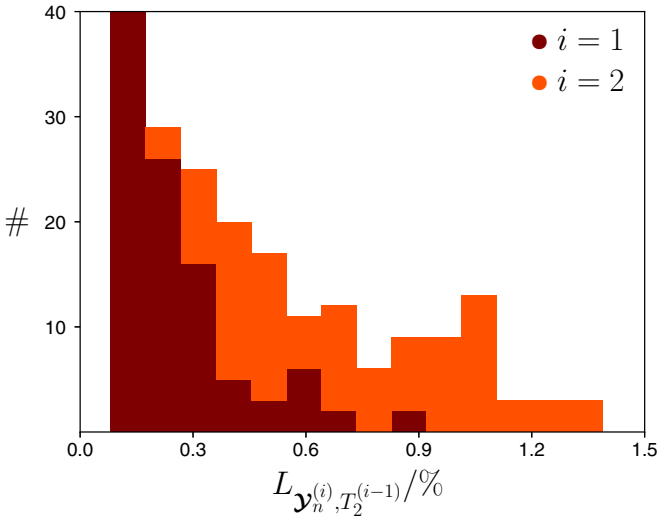


FIG. 1. Histogram of individual MAPEs $L_{\mathcal{Y}_n^{(1)},T_2^{(0)}}$ and $L_{\mathcal{Y}_n^{(2)},T_2^{(1)}}$ among all members of the ensembles $\mathcal{Y}^{(1)}$ and $\mathcal{Y}^{(2)}$ with respect to the corresponding test sets $T_2^{(0)}$ and $T_2^{(1)}$. We see that members of the first ensemble perform slightly better, which also leads to a better MAPE $L_{\mathcal{Y}^{(1)},T_2^{(0)}}$ for the entire ensemble $\mathcal{Y}^{(1)}$. Since all hyperparameters are drawn from the same distributions for both ensembles, it appears that it is an easier task to learn a linear relation than to learn a quadratic relation.

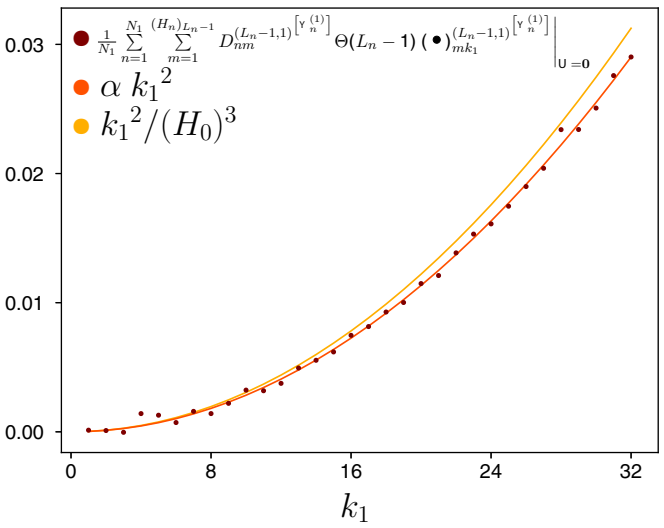


FIG. 2. First-order Taylor coefficients of the ensemble $\mathcal{Y}^{(1)}$, values of the model $\alpha {k_1}^2$ fitted over these coefficients, and first-order Taylor coefficients ${k_1}^2/(H_0)^3$ of the sampled Born series over the index k_1. As α deviates just slightly more than 1σ from $1/(H_0)^3$, this ensemble, indeed, applies the first-order Born approximation to shallow potentials in order to predict S-wave scattering lengths.

from the requirement of being a GELU-activated MLP, we allow various numbers of layers $L_n^{(i)}$, numbers $(H_l)_n^{(i)}$ of units per hidden layer, learning rates $\eta_n^{(i)}$ and weight decays $\lambda_n^{(i)}$: The former two are uniformly distributed random integers in the intervals $[3, 10]$ and $[16, 256]$, respectively. For the random floats $\overline{\eta}_n^{(1)}$ and $\overline{\lambda}_n^{(1)}$ drawn from the uniform distributions over the intervals $[2, 3]$ and $[3, 5]$, we work with an exponentially decaying learning rate schedule

$$(\eta_n^{(i)})_\epsilon = \exp\left(-\frac{\epsilon - 1}{\overline{\eta}_n^{(i)}}\right) \times 10^{-\overline{\eta}_n^{(i)}} \tag{29}$$

and with the weight decay

$$\lambda_n^{(i)} = 10^{-\overline{\lambda}_n^{(i)}}. \tag{30}$$

The index ϵ labels the current training epoch and ranges from $\epsilon = 1$ to $\epsilon = 20$. We decide to use the mean average percentage error (MAPE) as loss function,

$$L_{\mathcal{Y}_n^{(i)},t} = \frac{1}{|t|} \sum_{(\boldsymbol{U},a_0)\,\in\, t\,\subseteq\, T_{1/2}^{(i-1)}} \left| \frac{\mathcal{Y}_n^{(i)}(\boldsymbol{U}) - a_0}{a_0} \right|,$$

$$L_{\mathcal{Y}^{(i)},t} = \frac{1}{|t|} \sum_{(\boldsymbol{U},a_0)\,\in\, t\,\subseteq\, T_{1/2}^{(i-1)}} \frac{\left| \frac{1}{N_i} \sum_{i=1}^{N_i} \mathcal{Y}_n^{(i)}(\boldsymbol{U}) - a_0 \right|}{|a_0|}.$$

The upper expression is used to evaluate the loss of a single member $\mathcal{Y}_n^{(i)}$ during training, while the lower expression corresponds to the MAPE of the entire ensemble. Processing these losses and computing corresponding weight updates by the Adam optimizer (see Refs. [29] and [30]), we perform minibatch learning with batch size $B = 128$.

When it comes to the Taylor decomposition of the ensembles $\mathcal{Y}^{(i)}$, it is convenient to choose the same expansion point, i.e., $\boldsymbol{U} = \boldsymbol{0}$, as we have already seen for a_0 in Sec. II. We note that the scattering length $a_0(\boldsymbol{U}) = 0$ vanishes in the case with no interaction. Therefore we expect the dominating term of the Taylor series of $\mathcal{Y}^{(1)}$ not to be the first, constant term, but the second summand, containing the first-order derivatives $\partial \mathcal{Y}_n^{(1)}/\partial U_k|_{\boldsymbol{U}=\boldsymbol{0}}$. With $(\boldsymbol{U}, a_0(\boldsymbol{U})) \in T_{1/2}^{(0)}$ this motivates us to skip one iteration and to directly perform the substraction

$$a_0'(\boldsymbol{U}) = a_0(\boldsymbol{U}) - \frac{1}{N_1} \sum_{n=1}^{N_1} \mathcal{Y}_n^{(1)}(\boldsymbol{0}) - \frac{1}{N_1} \sum_{n=1}^{N_1} \sum_{k=1}^{H_0} \left. \frac{\mathcal{Y}_n^{(1)}}{\partial U_k} \right|_{\boldsymbol{U}=\boldsymbol{0}} U_k \tag{31}$$

in order to compute samples $(\boldsymbol{U}, a_0'(\boldsymbol{U})) \in T_{1/2}^{(1)}$ of the successive data sets. We expect these new targets to have a vanishing constant and linear contribution and, therefore, to behave mainly as $1/2 \cdot \boldsymbol{U}^\top K \boldsymbol{U}$ with the Hessian $K \in \mathbb{R}^{H_0 \times H_0}$. Since we are already satisfied with these first two orders, we stop after training the auxiliary ensemble $\mathcal{Y}^{(2)}$ on $T_1^{(1)}$, using the same training pipeline as before, and do not perform further iterations beyond that.

Both ensembles perform sufficiently well on their respective test sets, which follows from their low MAPEs of $L_{\mathcal{Y}^{(1)},T_2^{(0)}} = 0.152\%$ and $L_{\mathcal{Y}^{(2)},T_2^{(1)}} = 0.438\%$. Note that $\mathcal{Y}^{(2)}$ as well as its individual members exhibits a slightly worse performance than the members of $\mathcal{Y}^{(1)}$; cf. Fig. 1. Since both ensembles draw their hyperparameters from the same probability distributions, the task of adapting to a dominating, quadratic relation between inputs and targets appears to be more challenging than learning a constant or linear relation.

V. FIRST-ORDER BORN TERM

Given the first ensemble $\mathcal{Y}^{(1)}$, we first verify that its members have, indeed, adapted to a vanishing axis intercept. This is an important performance requirement, as the scattering length vanishes in the force-free case $\boldsymbol{U} = 0$, which we choose as an expansion point for our proxy model. Deriving errors of ensemble-related quantities by computing the standard deviation of that quantity among all members, we find

$$\mathcal{Y}^{(1)}(\mathbf{0}) = \frac{1}{N_1}\sum_{n=1}^{N_1}\mathcal{Y}_n^{(1)}(\mathbf{0}) = -(1.51 \pm 2.38)\times 10^{-4}. \quad (32)$$

As $\mathcal{Y}^{(1)}(\mathbf{0})$ takes a small value compared with the range of all targets in $T_{1/2}^{(0)}$ and since the corresponding error even has a slightly larger magnitude, we can confirm a vanishing axis intercept for the first ensemble.

The next step is crucial, not only to this iteration but also to the success of the following one: We compute the first-order Taylor coefficients of the ensemble $\mathcal{Y}^{(1)}$ using Eq. (24) and the formalism provided in Sec. III. This reveals its dominating, linear contribution in the space of sampled potentials. Ideally, the ensemble would reproduce the linear contribution in Eq. (3) of the scattering length one to one, which then would imply for the Taylor coefficients

$$\left.\frac{\partial\mathcal{Y}^{(1)}}{\partial U_{k_1}}\right|_{\boldsymbol{U}=\mathbf{0}} = \frac{1}{N_1}\sum_{n=1}^{N_1}\sum_{m=1}^{(H_n)_{L_n-1}} D_{nm}^{(L_n-1,1)\left[\mathcal{Y}_n^{(1)}\right]}\,\Theta(L_n-1)\left(\bullet\right)_{mk_1}^{(L_n-1,1)\left[\mathcal{Y}_n^{(1)}\right]}\Bigg|_{\boldsymbol{U}=\mathbf{0}} = \frac{{k_1}^2}{(H_0)^3} \quad (33)$$

The superscript $[\mathcal{Y}_n^{(1)}]$ points out that the respective quantity is computed for the weights and biases of the ensemble member $\mathcal{Y}_n^{(1)}$. In order to evaluate how well the left-hand and right-hand sides actually match, we fit the model $\alpha {k_1}^2$ to the $H_0 = 32$ ensemble Taylor coefficients on the left-hand side. Considering the mean and standard deviation of the distribution of all fitting parameters among the members of $\mathcal{Y}^{(1)}$ yields

$$\alpha = (2.834 \pm 0.169)\times 10^{-5}. \quad (34)$$

The ensemble's Taylor coefficients are displayed together with the fitted curve and the values ${k_1}^2/(H_0)^3$ in Fig. 2. We note that the deviation of the fitting parameter α from the value $1/(H_0)^3 = 3.052\times 10^{-5}$ is just slightly larger than 1σ. This shows that the ensemble $\mathcal{Y}^{(1)}$ reproduces the first-order Born approximation sufficiently well and, thereby, predicts S-wave scattering lengths for shallow potentials.

VI. SECOND-ORDER BORN TERM

Having identified and analyzed the linear contribution of the first ensemble $\mathcal{Y}^{(1)}$, it is time to move over to the successive data sets $T_{1/2}^{(1)}$ with their targets derived according to Eq. (31) and to train the auxiliary ensemble $\mathcal{Y}^{(2)}$. As argued in Sec. IV, it is the quadratic contribution, based on the Hessian, which dominates these new targets. Similar to the investigation of the first-order Taylor coefficients in Sec. V, we can specify an ideal case in which $\mathcal{Y}^{(2)}$ would reproduce the second-order Born term one to one, namely, if their second-order Taylor coefficients satisfy

$$\begin{aligned}\left.\frac{\partial^2\mathcal{Y}^{(2)}}{\partial U_{k_1}\,\partial U_{k_2}}\right|_{\boldsymbol{U}=\mathbf{0}} &= \frac{1}{N_2}\sum_{n=1}^{N_2}\sum_{m=1}^{(H_n)_{L_n-1}}\Bigg[D_{nm}^{(L_n-1,1)[\mathcal{Y}_n^{(2)}]}\,\Theta(L_n-2)\left(\bullet\!\!\rightarrow\!\!\bullet\right)_{mk_1k_2}^{(L-1,2)[\mathcal{Y}_n^{(2)}]} \\ &\quad + D_{nm}^{(L_n-1,2)[\mathcal{Y}_n^{(2)}]}\,\Theta(L_n-1)\left(\bullet\right)_{mk_1}^{(L-1,1)[\mathcal{Y}_n^{(2)}]}\left(\bullet\right)_{mk_2}^{(L-1,1)[\mathcal{Y}_n^{(2)}]}\Bigg]\Bigg|_{\boldsymbol{U}=\mathbf{0}} \\ &= -\frac{1}{(H_0)^5}k_1k_2\left(k_1+k_2-|k_1-k_2|\right).\end{aligned} \quad (35)$$

In order to investigate how closely the auxiliary ensemble actually approximates this ideal case, we fit the model $\beta k_1k_2(k_1+k_2-|k_1-k_2|)$. If we observe the fitting parameter β to closely approach the value $-1/(H_0)^5 = -2.98\times 10^{-8}$, we can be certain that $\mathcal{Y}^{(2)}$ applies the second-order Born term to predict the dominating, quadratic contribution.

To justify our perturbative ansatz, we not only compute the Hessian of the auxiliary ensemble $\mathcal{Y}^{(2)}$ but also consider the Hessian of the ensemble $\mathcal{Y}^{(1)}$ from the previous iteration. The latter can be expected to be significantly less faithful to the second-order Born term, since the quadratic contribution to scattering lengths of shallow potentials is lower than that of the linear term from Sec. V and thus has not been prioritized during training. Both Hessians and the Hessian of the Born series [cf. Eq. (3)] are shown in Fig. 3. For $\mathcal{Y}^{(2)}$ we find the fitting parameter

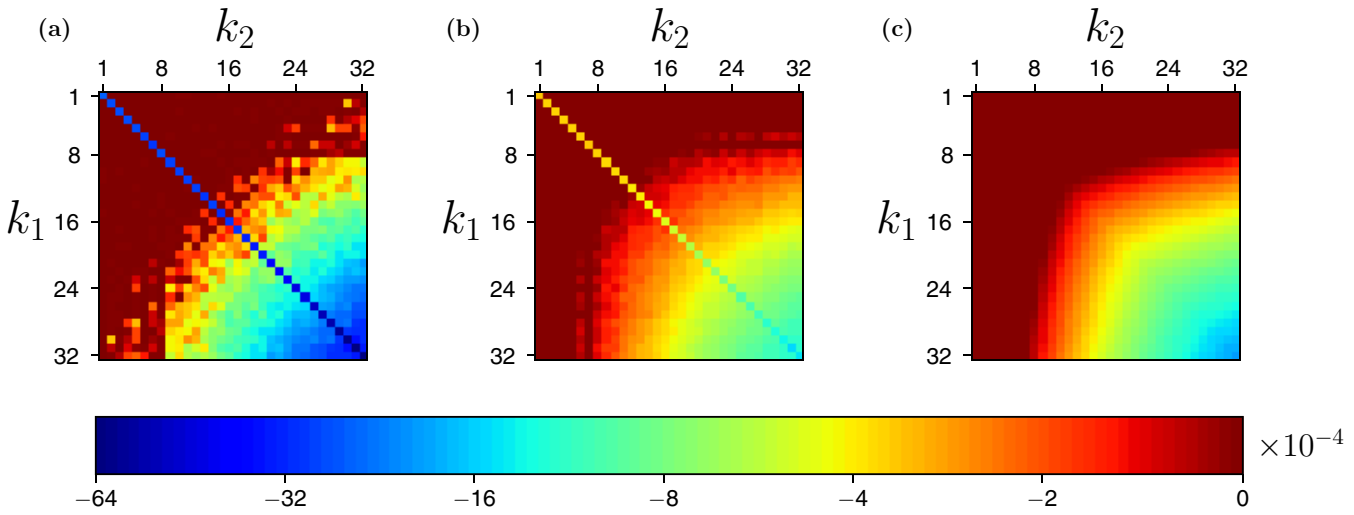


FIG. 3. Second-order Taylor coefficients of the ensembles $\mathcal{Y}^{(1)}$ (a) and $\mathcal{Y}^{(2)}$ (b) and the Hessian of the Born series (c). Both ensembles reproduce the basic behavior displayed by the Hessian in (c). Since adapting to the second-order Born term has not been prioritized during the training of $\mathcal{Y}^{(1)}$, the resulting elements in (a) are noisy, and very large values appear in the bottom right corner. In contrast to $\mathcal{Y}^{(1)}$, the auxiliary ensemble $\mathcal{Y}^{(2)}$ has adapted to the second-order Born term much better. The diagonals appearing on both ensemble Hessians are presumably an artifact that could be eliminated using other and more capable architectures than MLPs.

$\beta = (-2.25 \pm 1.63) \times 10^{-8}$. Indeed, the deviation from $-1/(H_0)^5$ is less than 1σ. However, at this point, we also notice the unfortunately large error, which may be explained by the slightly weaker performance of the auxiliary ensemble. Moreover, interestingly, we can observe a very distinct diagonal in both ensemble Hessians, which does not appear in the second-order Born term. Since weight decay and ensembling have a regulatory influence on the resulting predictions, we can exclude overfitting as cause. We therefore conjecture that these diagonals are artifacts that might disappear when using other, more capable architectures than MLPs. Apart from that, Fig. 3(b) shows that the auxiliary ensemble $\mathcal{Y}^{(2)}$ mostly reproduces the desired behavior. A glimpse at Fig. 3(a) lets us surmise that even the ensemble $\mathcal{Y}^{(1)}$ very roughly behaves as $-1/(H_0)^5 k_1 k_2 (k_1 + k_2 - |k_1 - k_2|)$. Nonetheless, by moving to the auxiliary ensemble, much of the noise attached to the coefficients in Fig. 3(a) is heavily reduced, and the desired shape of the Hessian displayed in Fig. 3(c) is much better approximated. In conclusion, we consider the Hessian of the auxiliary ensemble to be suitable for constructing an S-wave scattering length proxy.

In completing the second iteration, we can now use the gradient and the vanishing axis intercept of the first ensemble $\mathcal{Y}^{(1)}$ and the Hessian of the auxiliary ensemble $\mathcal{Y}^{(2)}$ to construct a much simpler proxy

$$p_0(\boldsymbol{U}) = \mathcal{Y}^{(1)}(\boldsymbol{0}) + \sum_{k_1=1}^{H_0} \frac{\partial \mathcal{Y}^{(1)}}{\partial U_{k_1}}\bigg|_{\boldsymbol{U}=\boldsymbol{0}} U_{k_1} + \frac{1}{2}\sum_{k_1=1}^{H_0}\sum_{k_2=1}^{H_0} \frac{\partial^2 \mathcal{Y}^{(2)}}{\partial U_{k_1}\partial U_{k_2}}\bigg|_{\boldsymbol{U}=\boldsymbol{0}} U_{k_1} U_{k_2}. \tag{36}$$

The proxy p_0 can be understood as a machine-learned second-order Born approximation. In order to examine its range of validity, we proceed as follows: At first we randomly generate two different potential shapes $\boldsymbol{n}_i \in \mathbb{R}^{H_0}$. For both of these shapes we generate a set of 100 equidistant potentials $\boldsymbol{U}_i = \|\boldsymbol{U}_i\| \boldsymbol{n}_i$ with magnitudes $\|\boldsymbol{U}_i\| \in [0, \ldots, 5]$. For each of these potentials $\boldsymbol{U}$ the above, machine-learned Born approximation $p_0(\boldsymbol{U})$ and true scattering lengths $a_0(\boldsymbol{U})$ are evaluated and plotted in Fig. 4. We observe that the relative error between $p_0(\boldsymbol{U})$ and $a_0(\boldsymbol{U})$ is less than 3% for shallow potentials with $\|\boldsymbol{U}\| \lesssim 1$. Beyond that regime, additional higher-order terms must be introduced to the proxy to make better scattering length predictions.

VII. DISCUSSION AND OUTLOOK

In this paper we propose a neural network perturbation theory for MLPs. This allows us to construct much simpler proxies of the original target function. The key idea of that perturbation theory is the successive identification and elimination of the leading-order contribution to the ensemble's Taylor decomposition. This establishes a sequence of ensembles that each specialize in approximating consecutive orders of the target function's Taylor decomposition. Combining these accordingly can, thus, be viewed as the first step of a proxy method; see Ref. [20].

Especially when dealing with deep MLPs, the computation of higher-order Taylor coefficients can be a challenging task. Nonetheless, we manage to obtain an analytical expression for partial derivatives of any order for arbitrarily deep, analytically activated MLPs in terms of propagators and vertices. The underlying formalism is motivated by Feynman diagrams in quantum field theories, and the entailed graph-theoretical approach makes the underlying combinatorics significantly more systematical and manageable. Note that the graphical representation of its derivatives does not depend on

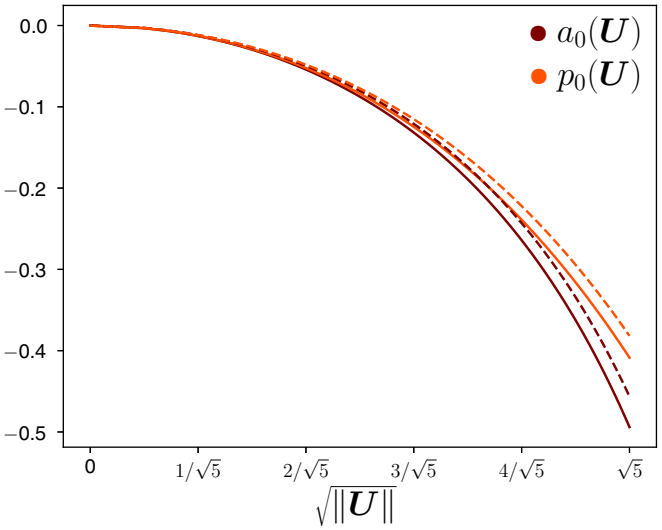


FIG. 4. Scattering lengths $a_0(\boldsymbol{U})$ and corresponding scattering length predictions $p_0(\boldsymbol{U})$ by the proxy from Eq. (36) for 200 different, sampled potentials that take one of two randomly generated shapes. Solid (dashed) lines are related to potentials of the first (second) randomly generated shape. The relative error between $p_0(\boldsymbol{U})$ and $a_0(\boldsymbol{U})$ is less than 3% for shallow potentials with $\|\boldsymbol{U}\| \lesssim 1$. Introducing successively higher-order terms to the proxy would provide better predictions and would, consequently, increase the range of validity.

the particular choice of an MLP: Indeed, the calculation of propagators, vertices, and saturation thresholds themselves may be altered due to varying weights, biases, activation functions, number of neurons, and hidden layers. However, there is no way to uniquely infer more information about its architecture from the mere structure of the contributing arborescences.

We apply this graphical formalism and neural network perturbation theory to S-wave scattering lengths of shallow potentials. For this, we train two ensembles within the same number of iterations. Using the axis intercept and gradient of the first ensemble and the Hessian from the second, auxiliary ensemble yields a proxy of the Born series, which reproduces the second-order Born approximation for shallow potentials. At this point, one could, of course, argue that it would have been much more convenient to simply train an NN $\mathcal{Y} : \mathbb{R}^{32} \to \mathbb{R}$ that is just the sum of one linear layer $\boldsymbol{l}$ and one bilinear layer B, i.e.,

$$\mathcal{Y}(\boldsymbol{U}) = \boldsymbol{l} \cdot \boldsymbol{U} + \tfrac{1}{2}\boldsymbol{U}^\top B \boldsymbol{U}.$$

Using this architecture instead of deep MLPs not only would reduce the computational effort significantly but also would have imposed some desired properties such as $\mathcal{Y}(\boldsymbol{0}) = 0$ and simultaneously learning the first- and second-order Born terms. Note that $\mathcal{Y}$ in this case is a second-order Taylor approximation by itself, which allows one to directly read off Taylor coefficients instead of deriving them first, as performed in our analysis. However, such an NN is not a universal approximator, as it violates the UAT, and therefore will fail in reproducing scattering lengths for deeper potentials than in this analysis. This is because models of this architecture are unable to adapt to higher-order terms of the Born series. Therefore using such an architecture may indeed simplify the analysis but must be well justified for the particular case.

Note that the obvious next step of this analysis would be the interpretation of the constructed proxy in the space of all sampled potentials. In doing so, we would just gather a *post hoc* interpretation, based on approximations and thus deviations from actual scattering lengths. In this case, prediction and interpretation therefore have to be understood as two independent instances. In recent years there have been many efforts to close the gap between prediction and interpretation by *ad hoc* interpretation methods. These exemplarily involve training NNs whose architectures either are intrinsically interpretable or can be brought into an interpretable representation; see Ref. [20]. At the cost of a prediction-interpretation trade-off, the advantage of *ad hoc* methods is that resulting interpretations are completely faithful to the NN's prediction, in contrast to the mentioned *post hoc* methods.

ACKNOWLEDGMENTS

We are grateful to Jürgen Gall as well as the reviewers for useful comments. We acknowledge partial financial support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) and the NSFC through the funds provided to the Sino-German Collaborative Research Center TRR110 "Symmetries and the Emergence of Structure in QCD" (DFG Project ID 196253076 - TRR 110, NSFC Grant No. 12070131001). Further support was provided by the Chinese Academy of Sciences (CAS) President's International Fellowship Initiative (PIFI) (Grant No. 2018DM0034), by VolkswagenStiftung (Grant No. 93562), and by the EU Horizon 2020 (Grant No. 824093).

APPENDIX: PROOFS

1. Proof of Equation (9)

Theorem 1. The first-order partial derivatives of the nmth matrix element of the lth layer propagator $D^{(l,p)}_{nm}$ of order p are given by

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \Delta^{(l,1)}_{mk},$$

where we have introduced the matrix elements

$$\Delta^{(l,1)}_{mk} = \sum_{q_l=1}^{H_l} \cdots \sum_{q_1=1}^{H_1} \delta_{mq_l} w^{(1)}_{q_1 k} \prod_{i=1}^{l-1} D^{(i,1)}_{q_{i+1}q_i}.$$

Proof. First of all, it is easy to see that the derivative with respect to the kth component x_k of the input is proportional to a propagator of higher order $p+1$. Due to the chain rule, the term $\partial z^{(l)}_m / \partial x_k$ appears,

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = \underbrace{w^{(l+1)}_{nm} \frac{d^{p+1} a^{(l,m)}}{dx^{p+1}}\left(z^{(l)}_m\right)}_{=D^{(l,p+1)}_{nm}} \sum_{q_l=1}^{H_l} \delta_{mq_l} \frac{\partial z^{(l)}_{q_l}}{\partial x_k}.$$

By inserting the recursive step from Eq. (5), this dependency can be shifted to the previous layer,

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \sum_{q_{l-1}=1}^{H_{l-1}} w^{(l)}_{q_l q_{l-1}} \frac{\partial y^{(l-1)}_{q_{l-1}}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \sum_{q_{l-1}=1}^{H_{l-1}} \underbrace{w^{(l)}_{q_l q_{l-1}} \frac{da^{(l-1,q_{l-1})}}{dx}\left(z^{(l-1)}_{q_{l-1}}\right)}_{=D^{(l-1,1)}_{q_l q_{l-1}}} \frac{\partial z^{(l-1)}_{q_{l-1}}}{\partial x_k}.$$

In the same manner, we can apply the chain rule successively to all antecedent layers until the base $y^{(0)}_{q_0} = x_{q_0}$ is reached. Thereby, each layer provides a matrix multiplication with a first-order propagator:

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \sum_{q_{l-1}=1}^{H_{l-1}} D^{(l-1,1)}_{q_l q_{l-1}} \sum_{q_{l-2}=1}^{H_{l-2}} D^{(l-2,1)}_{q_{l-1}q_{l-2}} \cdots \sum_{q_1=1}^{H_1} D^{(1,1)}_{q_2 q_1} \sum_{q_0=1}^{d} w^{(1)}_{q_1 q_0} \underbrace{\frac{\partial x_{q_0}}{\partial x_k}}_{=\delta_{q_0 k}}.$$

Rearranging those propagators and sums finally yields

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \cdots \sum_{q_1=1}^{H_1} \delta_{mq_l} w^{(1)}_{q_1 k} \prod_{i=1}^{l-1} D^{(i,1)}_{q_{i+1}q_i}.$$

2. Proof of Equation (11)

Theorem 2 [Equation (11)]. The Nth derivative of the propagator $D^{(l,p)}_{nm}$ is given by

$$\frac{\partial^N D^{(l,p)}_{nm}}{\partial x_{k_1} \cdots \partial x_{k_N}} = \sum_{c=1}^{N} D^{(l,p+c)}_{nm} \sum_{\sigma \in \mathcal{S}_N} \sum_{\boldsymbol{\pi} \in \Pi^c_N} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \prod_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)} \cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}.$$

Proof. We prove Eq. (11) by a complete induction. Therefore we quickly convince ourselves of its validity in the base case $N = 1$ with $\varepsilon_{(1)} = 1$,

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_{k_1}} = D^{(l,p+1)}_{nm} \sum_{(\pi_1) \,\in\, \{(1)\}} \frac{1}{\varepsilon_{(1)}} \sum_{\sigma \in \{\mathrm{id}\}} \Delta^{(l,\pi_1)}_{mk_{\sigma(\pi_1)}} = D^{(l,p+1)}_{nm} \Delta^{(l,1)}_{mk_1}.$$

This, indeed, corresponds to Eq. (9), which is also the defining equation for $\Delta^{(l,1)}_{mk_1}$. Subsequently, the inductive step involves evaluating the derivative

$$\frac{\partial^{N+1} D^{(l,p)}_{nm}}{\partial x_{k_1} \cdots \partial x_{k_{N+1}}} = \frac{\partial}{\partial x_{k_{N+1}}} \sum_{c=1}^{N} D^{(l,p+c)}_{nm} \sum_{\sigma \in \mathcal{S}_N} \sum_{\boldsymbol{\pi} \in \Pi^c_N} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \prod_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)} \cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}.$$

By applying the product rule, we either encounter propagator derivatives or derivatives of tensor elements, which we split into two distinct sums,

$$\begin{aligned}
\frac{\partial^{N+1} D^{(l,p)}_{nm}}{\partial x_{k_1} \cdots \partial x_{k_{N+1}}} \overset{(9),\,(10)}{=} & \sum_{c=1}^{N} D^{(l,p+c+1)}_{nm} \Delta^{(l,1)}_{mk_{N+1}} \sum_{\sigma \in \mathcal{S}_N} \sum_{\boldsymbol{\pi} \in \Pi^c_N} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \prod_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)} \cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}} \\
& + \sum_{c=1}^{N} D^{(l,p+c)}_{nm} \sum_{\sigma \in \mathcal{S}_N} \sum_{\boldsymbol{\pi} \in \Pi^c_N} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \sum_{i=1}^{c} \Delta^{(l,\pi_i+1)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)} \cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)} k_{N+1}} \\
& \times \prod_{\substack{i'=1 \\ i' \neq i}}^{c} \Delta^{(l,\pi_{i'})}_{mk_{\sigma\left(1+\sum_{j=1}^{i'-1}\pi_j\right)} \cdots k_{\sigma\left(\sum_{j=1}^{i'}\pi_j\right)}}.
\end{aligned}$$

Up to the Nth summand in the first sum and the first summand in the second sum, all other summands can be combined into one sum from $c = 2$ to $c = N$,

$$\begin{aligned}\frac{\partial^{N+1} D^{(l,p)}_{nm}}{\partial x_{k_1}\cdots\partial x_{k_{N+1}}} &= D^{(l,p+N+1)}_{nm}\Delta^{(l,1)}_{mk_{N+1}}\sum_{\sigma\in\mathcal{S}_N}\sum_{\boldsymbol{\pi}\in\Pi^N_N}\frac{1}{\varepsilon_{\boldsymbol{\pi}}}\prod_{i=1}^{N}\Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}\\
&+\sum_{c=2}^{N} D^{(l,p+c)}_{nm}\sum_{\sigma\in\mathcal{S}_N}\Bigg[\Delta^{(l,1)}_{mk_{N+1}}\sum_{\boldsymbol{\alpha}\in\Pi^{c-1}_N}\frac{1}{\varepsilon_{\boldsymbol{\alpha}}}\prod_{i=1}^{c-1}\Delta^{(l,\alpha_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\alpha_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\alpha_j\right)}}\\
&+\sum_{\boldsymbol{\beta}\in\Pi^c_N}\frac{1}{\varepsilon_{\boldsymbol{\beta}}}\sum_{i=1}^{c}\Delta^{(l,\beta_i+1)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\beta_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\beta_j\right)}k_{N+1}}\\
&\times\prod_{\substack{i'=1\\ i'\neq i}}^{c}\Delta^{(l,\beta_{i'})}_{mk_{\sigma\left(1+\sum_{j=1}^{i'-1}\beta_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i'}\beta_j\right)}}\Bigg]\\
&+D^{(l,p+1)}_{nm}\sum_{\sigma\in\mathcal{S}_N}\sum_{(\pi_1)\in\Pi^1_N}\frac{1}{\varepsilon_{(\pi_1)}}\Delta^{(l,\pi_1+1)}_{mk_{\sigma(1)}\cdots k_{\sigma(\pi_1)}k_{N+1}}.\end{aligned}$$

The two external summands can be explicitly derived using $\Pi^N_N = \{(1)^N_{i=1}\}$ and $\Pi^1_N = \{(N)\}$. In both cases, the symmetry factor is given by

$$\varepsilon_{(1,\ldots,1)} = \varepsilon_{(N)} = N!$$

and

$$\varepsilon_{(1,\ldots,1,1)} = \varepsilon_{(N+1)} = (N+1)!,$$

respectively, such that each summand finally can be written as a sum over $\mathcal{S}_{N+1}$:

$$\begin{aligned}\Delta^{(l,1)}_{mk_1}\cdots\Delta^{(l,1)}_{mk_{N+1}} &= \Delta^{(l,1)}_{mk_{N+1}}\sum_{\sigma\in\mathcal{S}_N}\sum_{\boldsymbol{\pi}\in\Pi^N_N}\frac{1}{\varepsilon_{\boldsymbol{\pi}}}\prod_{i=1}^{N}\Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}\\
&=\sum_{\substack{\sigma\in\mathcal{S}_{N+1}\\ \sigma(N+1)=N+1}}\sum_{\boldsymbol{\pi}\in\Pi^{N+1}_{N+1}}\frac{N+1}{\varepsilon_{\boldsymbol{\pi}}}\prod_{i=1}^{N}\Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}\\
&=\sum_{\sigma\in\mathcal{S}_{N+1}}\sum_{\boldsymbol{\pi}\in\Pi^{N+1}_{N+1}}\frac{1}{\varepsilon_{\boldsymbol{\pi}}}\prod_{i=1}^{N}\Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}\end{aligned}$$

and

$$\begin{aligned}\Delta^{(l,N+1)}_{ml_1\cdots k_{N+1}} &= \sum_{\sigma\in\mathcal{S}_N}\sum_{(\pi_1)\in\,\Pi^1_N}\frac{1}{\varepsilon_{(\pi_1)}}\Delta^{(l,\pi_1+1)}_{mk_{\sigma(1)}\cdots k_{\sigma(\pi_1)}k_{N+1}}\\
&=\sum_{\substack{\sigma\in\mathcal{S}_{N+1}\\ \sigma(N+1)=N+1}}\sum_{(\pi_1)\in\,\Pi^1_{N+1}}\frac{N+1}{\varepsilon_{(\pi_1)}}\Delta^{(l,\pi_1+1)}_{mk_{\sigma(1)}\cdots k_{\sigma(\pi_1)}}\\
&=\sum_{\sigma\in\mathcal{S}_{N+1}}\sum_{(\pi_1)\in\,\Pi^1_{N+1}}\frac{1}{\varepsilon_{(\pi_1)}}\Delta^{(l,\pi_1+1)}_{mk_{\sigma(1)}\cdots k_{\sigma(\pi_1)}}.\end{aligned}\tag{A1}$$

Finally, the remaining sum can be expressed as a sum over Π^c_{N+1}. Then, combining all terms finally proves the inductive step,

$$\begin{aligned}\frac{\partial^{N+1} D^{(l,p)}_{nm}}{\partial x_{k_1}\cdots\partial x_{k_{N+1}}} &= D^{(l,p+(N+1))}_{nm}\sum_{\sigma\in\mathcal{S}_{N+1}}\sum_{\boldsymbol{\pi}\in\Pi^{N+1}_{N+1}}\frac{1}{\varepsilon_{\boldsymbol{\pi}}}\prod_{i=1}^{N}\Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}\\
&+\sum_{c=2}^{N} D^{(l,p+c)}_{nm}\sum_{\sigma\in\mathcal{S}_{N+1}}\sum_{\boldsymbol{\pi}\in\Pi^{c}_{N+1}}\frac{1}{\varepsilon_{\boldsymbol{\pi}}}\prod_{i=1}^{c}\Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}\end{aligned}$$

$$+ D^{(l,p+1)}_{nm} \sum_{\sigma\in\mathcal{S}_{N+1}} \sum_{\boldsymbol{\pi}\in\Pi^{1}_{N+1}} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \Delta^{(l,\pi_1+1)}_{mk_{\sigma(1)}\cdots\, k_{\sigma(\pi_1)}}$$

$$= \sum_{c=1}^{N+1} D^{(l,p+c)}_{nm} \sum_{\sigma\in\mathcal{S}_{N+1}} \sum_{\boldsymbol{\pi}\in\Pi^{c}_{N+1}} \frac{1}{\varepsilon_{\boldsymbol{\pi}}} \prod_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk_{\sigma\left(1+\sum_{j=1}^{i-1}\pi_j\right)}\cdots\, k_{\sigma\left(\sum_{j=1}^{i}\pi_j\right)}}.$$

3. Proof of Equation (23)

Theorem 3 [Equation (23)]. The tensor elements $\Delta^{(l,p)}_{mk_1\cdots k_N}$ can be expressed as the following weighted sum of all N-vertex arborescences, as defined in Eq. (22), with adjacency matrices $A \in \mathbb{A}^N$,

$$\Delta^{(l,N)}_{mk_1\cdots k_N} = \sum_{A\in\mathbb{A}^N} \Theta[l-\alpha(A)]\, \delta^{(l,N)}_{mk_1\cdots k_N}(A).$$

Weighting with factors $\Theta[l-\alpha(A)]$ causes an arborescence only to contribute as long as the layer l for which the tensor element is considered overshoots the saturation threshold $\alpha(A)$, given in Eq. (20).

Proof. Since the base case ($N = 1$) has already been shown in Eq. (16), we directly start with the inductive step. The commutator formula

$$\left[O, \prod_{c=1}^{N} B_c\right] = \sum_{i=0}^{N-1} \left(\prod_{j=1}^{i} B_j\right) [O, B_{i+1}] \left(\prod_{j=i+2}^{N} B_j\right) \tag{A2}$$

will later prove to be useful. For $O = \partial/\partial x_{k_{N+1}}$ and for

$$B_c(A) = \bigOmega_{\substack{q_c(A)\ k_c \\ (q_a^{\{c\}})_{a=1}^{j_c(A)}\ (j_b^{\{c\}})_{b=1}^{\max_r(A_{cr})}}} \prod_{b=1}^{\max_r(A_{cr})} D^{(1+n_{bc}(A))}_{bc}(A)$$

from the cth vertex of the arborescence $\delta^{(l,N)}_{mk_1\cdots k_N}(A)$, we derive the individual commutators

$$\left[\frac{\partial}{\partial x_{k_{N+1}}}, B_c(A)\right] \overset{(15)}{=} \Theta\left(l - \max_r(A_{cr}) - 1\right)$$
$$\times \bigOmega_{\substack{q_c(A)\ k_c \\ (q_a^{\{c\}})_{a=1}^{j_c(A)}\ (j_b^{\{c\}})_{b=1}^{\max_r(A_{cr})}}} \prod_{b=1}^{\max_r(A_{cr})} D^{(1+n_{bc}(A))}_{bc}(A)$$
$$\times D^{(\max_r(A_{cr})+1,2)}_{q^{\{c\}}_{j^{\{c\}}_{\max_r(A_{cr})+1}+1}\, q^{\{c\}}_{j^{\{c\}}_{\max_r(A_{cr})+1}}} \bigOmega_{\substack{q^{\{c\}}_{j^{\{c\}}_{\max_r(A_{cr})+1}}\ k_{N+1} \\ (q'^{\{c\}}_a)_{a=1}^{\max_r(A_{cr})+1}}}$$
$$+ \bigOmega_{\substack{q_c(A)\ k_c \\ (q_a^{\{c\}})_{a=1}^{j_c(A)}\ (j_b^{\{c\}})_{b=1}^{\max_r(A_{cr})}}} \sum_{b=1}^{\max_r(A_{cr})} D^{(2+n_{bc}(A))}_{bc}(A)$$
$$\times \prod_{\substack{b'=1 \\ b'\neq b}}^{\max_r(A_{cr})} D^{(1+n_{b'c}(A))}_{b'c} \bigOmega_{\substack{q^{\{c\}}_{j^{\{c\}}_b}\ k_{N+1} \\ (q''^{\{c\}}_a)_{a=1}^{j^{\{c\}}_b}}}$$

Due to the derivation, a new vertex is introduced to each summand. However, note that the way this new vertex is connected to the given vertices differs in both terms: In the first summand there now appears to be an additional propagator of second order establishing a connection to the $(N+1)$th vertex, while in the remaining $\max_r(A_{cr})$ summands, the order of the bth propagator is raised by 1, which also allows an additional connection to the new vertex. Let us define the set

$$\nu_c(A) = \bigcup_{b=1}^{\max_r(A_{cr})+1} \left\{ \begin{pmatrix} A_{11} & \cdots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \cdots & A_{cN} & b \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \cdots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right\}, \tag{A4}$$

which is a subset of $\mathbb{A}^{N+1}$. Its $|\nu_c(A)| = \max_r(A_{cr}) + 1$ elements correspond to adjacency matrices of N-vertex arborescences, which have been extended by an $(N+1)$th vertex, which is connected to the cth vertex. For the element with $b = \max_r(A_{cr}) + 1$, this corresponds to establishing a connection via an additional propagator, which is consequently of second order. Otherwise, we have $b \in \{1, \ldots, \max_r(A_{cr})\}$, which corresponds to raising the order of the bth propagator in the cth vertex and thereby allows being connected with the new vertex.

The observations made above can be formulated in the language of adjacency matrices: In terms of $(N+1) \times (N+1)$ adjacency matrices of the set $\nu_c(A)$, the commutator in Eq. (A3) is given by

$$\begin{aligned} \left[\frac{\partial}{\partial x_{k_{N+1}}}, B_c(A) \right] = \Theta[l - \max_r(A_{cr}) - 1] B_c \left[\begin{pmatrix} A_{11} & \cdots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \cdots & A_{cN} & \max_r(A_{cr}) + 1 \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \cdots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right] \\ \times B_{N+1} \left[\begin{pmatrix} A_{11} & \cdots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \cdots & A_{cN} & \max_r(A_{cr}) + 1 \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \cdots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right] \\ + \sum_{b=1}^{\max_r(A_{cr})} B_c \left[\begin{pmatrix} A_{11} & \cdots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \cdots & A_{cN} & b \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \cdots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right] B_{N+1} \left[\begin{pmatrix} A_{11} & \cdots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \cdots & A_{cN} & b \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \cdots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right]. \end{aligned} \tag{A5}$$

Here, we could express the $(N+1)$th vertex as a term $B_{N+1}(A')$ with $A' \in \nu_c(A)$, due to the $(N+1)$th line containing only zeros, thus $\max_r(A'_{N+1,r}) = 0$, and due to $\beta_{N+1}(A') = c$ as well as $A'_{\beta_{N+1}(A'),N+1} = b$,

$$\Omega_{\substack{(q_a^{\{N+1\}})_{a=1}^{j_b^{\{c\}}} \\ q_{j_b^{\{c\}}}^{\{c\}} k_{N+1}}} = \Omega_{\substack{(q_a^{\{N+1\}})_{a=1}^{j_{N+1}(A')} (j_b^{\{N+1\}})_{b=1}^{\max_r(A'_{N+1,r})} \\ q_{N+1}(A') k_{N+1}}} = B_{N+1} \left[\begin{pmatrix} A_{11} & \ldots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \ldots & A_{cN} & b \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \ldots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right].$$

Each element of $\nu_c(A)$ is represented in this sum, which implies that each possible connection from the cth vertex to the $(N+1)$th vertex is established. Note that the first summand, which introduces a new propagator, is the only term that may alter

the saturation threshold of the arborescence, namely, in the case that $\max_r(A_{cr}) \geqslant \alpha(A)$. Therefore we write

$$\Theta[l - \max_r(A_{cr}) - 1]\Theta[l - \alpha(A)] = \Theta\left(l - \alpha\left[\left[\begin{pmatrix} A_{11} & \cdots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \cdots & A_{cN} & \max_r(A_{cr}) + 1 \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \cdots & A_{NN} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}\right]\right]\right). \tag{A6}$$

The derivative of the arborescence $\delta^{(l,N)}_{mk_1\cdots k_N}(A)$ can be written as

$$\frac{\partial}{\partial x_{k_{N+1}}}\delta^{(l,N)}_{mk_1\cdots k_N}(A) = \left[\frac{\partial}{\partial x_{k_{N+1}}}, \delta^{(l,N)}_{mk_1\cdots k_N}(A)\right] = \sum_{q_l^{\{0\}}} \delta_{mq_l^{\{0\}}} \sum_{j^{\{0\}}_{A_{01}}} \delta_{l j^{\{0\}}_{A_{01}}} \left[\frac{\partial}{\partial x_{k_{N+1}}}, \prod_{c=1}^{N} B_c(A)\right].$$

Using the commutator relation in Eq. (A2), we can express $\delta^{(l,N)}_{mk_1\cdots k_N}(A)$ in terms of the individual commutators from Eq. (A5),

$$\frac{\partial}{\partial x_{k_{N+1}}}\delta^{(l,N)}_{mk_1\cdots k_N}(A) = \sum_{q_l^{\{0\}}} \delta_{mq_l^{\{0\}}} \sum_{j^{\{0\}}_{A_{01}}} \delta_{l j^{\{0\}}_{A_{01}}} \sum_{i=0}^{N-1} \left(\prod_{j=1}^{i} B_j\right)\left[\frac{\partial}{\partial x_{k_{N+1}}}, B_{i+1}\right]\left(\prod_{j=i+2}^{N} B_j\right). \tag{A7}$$

It is very insightful to analyze Eq. (A7): We already know that the ith commutator $[\partial/\partial x_{k_{N+1}}, B_i(A)]$ is a sum of $\max_r(A) + 1$ terms and corresponds to establishing a connection from the ith vertex to the $(N + 1)$th vertex, either by introducing a new propagator or by raising the order of an already existing propagator by 1. However, Eq. (A7) is a sum of N terms with the ith summand containing the ith commutator. This means that all allowed connections from all of the given N vertices to the $(N + 1)$th vertex are covered here. As the ith commutator leaves other vertices unaltered,

$$i \neq i' \Rightarrow \forall A' \in \nu_{i'}(A) : B_i(A') = B_i(A),$$

we can write for a single arborescence, using Eq. (A6),

$$\Theta[l - \alpha(A)]\frac{\partial}{\partial x_{k_{N+1}}}\delta^{(l,N)}_{mk_1\cdots k_N}(A) = \sum_{A' \in \nu(A)} \Theta[l - \alpha(A')]\delta^{(l,N+1)}_{mk_1\cdots k_{N+1}}(A'), \tag{A8}$$

where we sum over the union

$$\nu(A) = \bigcup_{c=1}^{N} \nu_c(A).$$

As the introduction of a new vertex to a given arborescence only influences the corresponding adjacency matrix by appending a new line and column but leaves the original adjacency matrix unaltered, the disjuncture

$$A_1 \neq A_2 \Rightarrow \nu(A_1) \cap \nu(A_2) = \emptyset$$

is obvious. Nonetheless, it can be easily argued that $\mathbb{A}^{N+1}$ is the union of all $\nu(A)$ for $A \in \mathbb{A}^N$. Therefore it follows that both of the following sums must be identical:

$$\sum_{A \in \mathbb{A}_N} \sum_{A' \in \nu(A)} \cdots = \sum_{A \in \mathbb{A}_{N+1}} \cdots .$$

Using Eq. (A8), we finally complete the inductive step,

$$\Delta^{(l,N+1)}_{mk_1\cdots k_{N+1}} = \frac{\partial}{\partial x_{k_{N+1}}}\Delta^{(l,N)}_{mk_1\cdots k_N} = \sum_{A \in \mathbb{A}^N} \Theta[l - \alpha(A)]\frac{\partial}{\partial x_{k_{N+1}}}\delta^{(l,N)}_{mk_1\cdots k_N}(A) = \sum_{A \in \mathbb{A}^{N+1}} \Theta[l - \alpha(A)]\delta^{(l,N+1)}_{mk_1\cdots k_{N+1}}(A).$$

[1] P. Mehta and D. J. Schwab, An exact mapping between the variational renormalization group and deep learning, arXiv:1410.3831.

[2] P. Baldi, P. Sadowski, and D. Whiteson, Searching for exotic particles in high-energy physics with deep learning, Nat. Commun. **5**, 4308 (2014).

[3] K. Mills, M. Spanner, and I. Tamblyn, Deep learning and the Schrödinger equation, Phys. Rev. A **96**, 042113 (2017).

[4] J. W. Richards, D. L. Starr, N. R. Butler, J. S. Bloom, J. M. Brewer, A. Crellin-Quick, J. Higgins, R. Kennedy, and M. Rischard, On machine-learned classification of variable stars with sparse and noisy time-series data, Astrophys. J. **733**, 10 (2011).

[5] A. Buckley, A. Shilton, and M. J. White, Fast supersymmetry phenomenology at the Large Hadron Collider using machine learning techniques, Comput. Phys. Commun. **183**, 960 (2012).

[6] P. Graff, F. Feroz, M. P. Hobson, and A. N. Lasenby, SKYNET: an efficient and robust neural network training tool for machine learning in astronomy, Mon. Not. R. Astron. Soc. **441**, 1741 (2014).

[7] G. Carleo and M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science **355**, 602 (2017).

[8] S. J. Wetzel and M. Scherzer, Machine learning of explicit order parameters: From the Ising model to SU(2) lattice gauge theory, Phys. Rev. B **96**, 184410 (2017).

[9] Y. H. He, Machine-learning the string landscape, Phys. Lett. B **774**, 564 (2017).

[10] Y. Fujimoto, K. Fukushima, and K. Murase, Methodology study of machine learning for the neutron star equation of state, Phys. Rev. D **98**, 023019 (2018).

[11] Y. Wu, P. Zhang, H. Shen, and H. Zhai, Visualizing neural network developing perturbation theory, Phys. Rev. A **98**, 010701(R) (2018).

[12] Z. M. Niu, H. Z. Liang, B. H. Sun, W. H. Long, and Y. F. Niu, Predictions of nuclear β-decay half-lives with machine learning and their impact on r-process nucleosynthesis, Phys. Rev. C **99**, 064307 (2019).

[13] J. Brehmer, K. Cranmer, G. Louppe, and J. Pavez, Constraining Effective Field Theories with Machine Learning, Phys. Rev. Lett. **121**, 111801 (2018).

[14] J. Steinheimer, L. Pang, K. Zhou, V. Koch, J. Randrup, and H. Stoecker, A machine learning study to identify spinodal clumping in high energy nuclear collisions, J. High Energy Phys. 12 (2019) 122.

[15] A. J. Larkoski, I. Moult, and B. Nachman, Jet substructure at the Large Hadron Collider: A review of recent advances in theory and machine learning, Phys. Rep. **841**, 1 (2020).

[16] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control, Signals Systems **2**, 303 (1989).

[17] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks **4**, 251 (1991).

[18] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, Explaining nonlinear classification decisions with deep Taylor decomposition, Pattern Recognit. **65**, 211 (2017).

[19] M. T. Ribeiro, S. Singh, and C. Guestrin, Why should I trust you?: Explaining the predictions of any classifier, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Association for Computing Machinery, New York, 2016), pp. 1135–1144.

[20] F. Fan, J. Xiong, and G. Wang, On interpretability of artificial neural networks, arXiv:2001.02522.

[21] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I. Nonstiff Problems* (Springer, Berlin, 1993).

[22] M. Nielsen, Neural Networks and Deep Learning, 2015, http://neuralnetworksanddeeplearning.com/.

[23] N. Kamiyama, Arborescence problems in directed graphs: Theorems and algorithms, Interdiscip. Inf. Sci. **20**, 51 (2014).

[24] B. Jonsson, and S. T. Eng, Solving the Schrodinger equation in arbitrary quantum-well potential profiles using the transfer matrix method, IEEE J. Quantum Electron. **26**, 2025 (1990).

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelstein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, T. Alykhan, S. Chilamkurthy, B. Steiner, F. Lu, J. Bai *et al.*, PyTorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., Vancouver, Canada, 2019), Vol. 32, pp. 8024–8035.

[26] D. Hendrycks and K. Gimpel, Gaussian error linear units (GELUs), arXiv:1606.08415v4.

[27] Y. Liu, J. Zhang, C. Gao, J. Qu, and L. Ji, Natural-logarithm-rectified activation function in convolutional neural networks, in *5th International Conference on Computer and Communications (ICCC)* (IEEE Computer Society, Los Alamitos, CA, 2019), pp. 2000–2008.

[28] K. He, X. Zhang, S. Ren, and J. Sun, Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification, in *2015 IEEE International Conference on Computer Vision (ICCV)* (IEEE, Piscataway, NJ, 2015), pp. 1026–1034.

[29] D. P. Kingma and J. L. Ba, Adam: A method for stochastic optimization, in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA (2015), arXiv:1412.6980v9.

[30] I. Loshchilov and F. Hutter, Decoupled weight decay regularization, in *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA (2019), arXiv:1711.05101v3.