A Neural Network Perturbation Theory Based on the Born Series

Bastian Kaschak and Ulf-G. Meißner

1Helmholtz-Institut für Strahlen- und Kernphysik and Bethe Center for Theoretical Physics, Universität Bonn, D-53115 Bonn, Germany
2Institute for Advanced Simulation, Institut für Kernphysik, and Jülich Center for Hadron Physics, Forschungszentrum Jülich, D-52425 Jülich, Germany
3Tbilisi State University, 0186 Tbilisi, Georgia

Deep Learning has become an attractive approach towards various data-based problems of theoretical physics in the past decade. Its protagonists, the deep neural networks (DNNs), are capable of making accurate predictions for data of arbitrarily high complexity. A well-known issue most DNNs share is their lack of interpretability. In order to explain their behavior and extract physical laws they have discovered during training, a suitable interpretation method has, therefore, to be applied post-hoc. Due to its simplicity and ubiquity in quantum physics, we decide to present a rather general interpretation method in the context of two-body scattering: We find a one-to-one correspondence between the $n$th-order Born approximation and the $n$th-order Taylor approximation of deep multilayer perceptrons (MLPs), that predict $S$-wave scattering lengths $a_0$ for discretized, attractive potentials of finite range. This defines a perturbation theory for MLPs similarly to Born approximations defining a perturbation theory for $a_0$. In the case of shallow potentials, lower-order approximations, that can be argued to be local interpretations of respective MLPs, reliably reproduce $a_0$. As deep MLPs are highly nested functions, the computation of higher-order partial derivatives, which is substantial for a Taylor approximation, is an effortful endeavour. By introducing quantities we refer to as propagators and vertices that depend on the MLP’s weights and biases, we establish a graph-theoretical approach towards partial derivatives and local interpretability. Similar to Feynman rules in quantum field theories, we find rules that systematically assign diagrams consisting of propagators and vertices to the corresponding order of the MLP perturbation theory.
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 also receives 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 like the concept of a layered architecture or the use of non-linear activation functions originate in much simpler MLP-architectures. Furthermore, an MLP representation can be found for most NNs with feedforward architectures, like convolutional neural networks (CNNs), radial-basis-function networks (RBNFs) or residual neural networks (ResNets).

The property that makes NNs perform so well in many different applications is that of being an universal approximator: As long as the architecture comprises an output layer and at least one hidden layer that is activated via a bounded, non-linear 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, 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 non-linearly activated hidden layers, instead, which eventually opens up the field of Deep Learning. Here, the term “deep” refers to a large number of such non-linearly activated layers. Its protagonists, the deep neural networks (DNNs), are known for their enormous predictive power and for demonstrating super-human performances for specific tasks like object recognition. Last but not least, this makes them a promising approach towards problems of theoretical physics, as shown e.g. in Refs. [1–3].

An issue many NNs, especially DNNs, share is that the way they act on the input space remains unclear to the human observer, despite the knowledge of all trained parameters. This is the reason why the ML-literature refers to them as black boxes. Given a black box, it is almost impossible to predict how its output changes for small perturbations around some reference input without an explicit evaluation. Furthermore, this forbids extracting the underlying formalism the NN has developed during training and now applies to new input data. However, the latter is an important earmark of interpretability. Ref. [18] defines the interpretability of an NN in terms of simulatability, the understanding of how that NN acts on the input space, and decomposability, the understanding of the smaller components like neurons and layers of an NN. Indeed, there are applications in which a lack of interpretability is not too dramatic. However, as soon as an NN is the only instance in an important decision or evaluation process, or if the understanding of a data set depends significantly on the predictions of an NN, interpretability is highly desirable.

Due to developments in deep learning and a trend towards deeper architectures, an exponentially increasing number of articles on interpretability has been published over the past years [18]. A beautiful example is given in Ref. [11]: An MLP is trained on a set of shallow, attractive potentials to predict corresponding S-wave scattering lengths. Subsequently, by inspecting the weight matrix of the first hidden layer, it is found that the weights in all active neurons appear to satisfy a quadratic pattern, that is $w_{nm} \propto m^2$, which finally can be proven to reproduce the first-order Born-approximation. When trained on a set containing deeper potentials, an additional pattern among some of the active neurons emerges, which can be related to the second-order Born-term. Therefore, it is argued that MLP’s develop a quantum perturbation theory by applying Born approximations of successively higher order, when trained on deeper potentials. Ref. [19] is another illustrative example for interpretability: Here, the deep Taylor decomposition of image classifiers is considered. Given a root point of the classifier, a heatmap can be constructed using first-order derivatives, assigning to each pixel a certain relevance value. This visualizes those pixels, that are substantially involved in the resulting decision, and, therefore, falls under the category of saliency methods according to Ref. [18].
Motivated by Refs. [11] and [19], we develop a rather general interpretation method in the context of two-body scattering. Here, two-body scattering appears to be a suitable field of application, due to its simplicity and its ubiquity in quantum physics. Given an ensemble of deep MLPs that predict S-wave scattering lengths for shallow, attractive potentials of finite range, the idea is to generate an interpretable proxy by considering the ensemble’s Taylor approximation. This approach itself is not new. However, while local interpretation methods like the deep Taylor decomposition in Ref. [19] or LIME in Ref. [20] merely use first-order derivatives or a linear regression of adjacent synthetic samples, respectively, higher-order derivatives have not yet been considered in the literature. This makes it even more compelling for us to show how the latter can be systematically determined in the case of MLPs.

Especially for deep MLPs, which, as such, are highly nested non-linear functions, the analytical computation of Taylor coefficients of arbitrarily high order is anything but trivial. Although using numerical instead of analytical derivatives seemingly simplifies this problem, they are inaccurate due to truncation and round-off errors and do not reveal the contribution of the individual weights and biases to a particular order. Similar 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-rule and understand arbitrary derivatives of an MLP in terms of tensor products. We observe two distinct classes of quantities, 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. The naming is intentional, as we discover several similarities between the Taylor expansion of MLPs and perturbation theory in quantum field theories. We find a one-to-one correspondence between the $n$th-order Born approximation and the $n$th-order Taylor approximation of the trained MLPs. In the same way as Born approximations define a perturbation theory for scattering lengths, we argue for the above Taylor approximations to define a perturbation theory for MLPs. Analogously to Feynman rules, there are underlying rules that specify which combinations of vertices and propagators, i.e. which diagrams are allowed and contribute to a certain Taylor coefficient. One major difference is, however, that loops are not allowed in contrast to quantum field theories. In a graph theoretical context, we can show these diagrams to be oriented and rooted trees, i.e. arborescences.

Having established such a perturbation theory for MLPs, based on the Born series, we use it to derive the first- and second-order Taylor coefficients of our ensemble. While the first one reproduces the first-order Born term, indeed, the latter deviates strongly from the actual second-order Born term and has a devastatingly large variance. This is neither right nor wrong, as the contribution of the second order is much lower compared to the first order, and just reveals how the ensemble operates on extremely shallow potentials, namely by relying primarily on the first order. Nonetheless, we also want to recover the second-order Born term using MLP perturbation theory. Therefore, we generate data sets with scattering lengths from which the first-order Born term has been removed. In these reduced scattering lengths the second order is, therefore, the leading order. When training an additional ensemble of MLPs to predict these, we observe a much better agreement between its second-order Taylor coefficients and the second-order Born term.

The manuscript is organized as follows: In Sect. [II] we develop the perturbation theory for MLPs. Sect. [III] discusses the Born series in quantum two-body scattering as a Taylor series in the space of discretized potentials. Sect. [IV] contains the details of the NN used and the its training. The first-order Born term is evaluated and discussed in Sect. [V] followed by the investigation of the second-order Born term in Sect. [VI]. We end with a discussion and outlook in Sect. [VII]. Various technicalities are relegated to the appendix.

### II. PERTURBATION THEORY FOR MULTILAYER PERCEPTRONS

Given an input $x_0 \in \mathbb{R}^{H_0}$ that the output $Y(x_0) \in \mathbb{R}^{H_L}$ is produced for by the NN we denote as $Y$. If and only if $Y$ is interpretable in vicinity of $x_0$, it is possible to predict how the output will change for a
small perturbation $\delta x$ of the input without actually processing the perturbed input $x = x_0 + \delta x$ by $Y$. This is equivalent to discovering an underlying formalism that the neural network has learned during training and applying exactly that set of rules to $x + \delta x$. Obviously, local interpretability can be established by a Taylor expansion of each component of $Y$ around a given expansion point $x_0$.

$$Y_n(x) = Y_n(x_0) + \sum_{k=1}^d \frac{\partial Y_n}{\partial x_k} |_{x=x_0} (x - x_0)_k$$

$$+ \frac{1}{2} \sum_{k_1=1}^d \sum_{k_2=1}^d \frac{\partial^2 Y_n}{\partial x_{k_1} \partial x_{k_2}} |_{x=x_0} (x - x_0)_{k_1} (x - x_0)_{k_2}$$

$$+ \ldots$$

The mentioned formalism can be parametrized by the Taylor coefficients $\partial^N Y_n / (\partial x_{k_1} \ldots \partial x_{k_N}) |_{x=x_0}$. Finally, since higher-order terms can be neglected for small perturbations, this naturally defines a perturbation theory for the neural network $Y$. As many feedforward-architectures like CNNs, RBFNs and ResNets have an MLP-representation, the following description of such an NN perturbation theory is accordingly given with respect to MLPs.

An MLP with $L$ layers is the prototype example of a layered architecture and can be understood as a non-linear function $Y : \mathbb{R}^{H_0} \rightarrow \mathbb{R}^{H_L}$ between real vector spaces. The term “layered” describes that $Y$ is a composition $Y = Y_L \circ \ldots Y_1$ of $L$ layers $Y_l : \mathbb{R}^{H_{l-1}} \rightarrow \mathbb{R}^{H_l}$ containing $H_l$ neurons $z_n^{(l)}$. In MLPs exclusively linear layers are used in combination with non-linear activation functions $a^{(l,n)} : \mathbb{R} \rightarrow \mathbb{R}$, where $a^{(l,n)}$ is applied to the $n$th neuron of the $l$th layer. This can be formulated recursively,

$$Y_n(x) = y_n^{(L)}, \quad (1)$$

with the recursive step

$$y_n^{(l)} = a^{(l,n)} (z_n^{(l)}), \quad z_n^{(l)} = \sum_{m=1}^{H_{l-1}} w_{nm}^{(l)} y_m^{(l-1)} + b_n^{(l)}, \quad (2)$$

together with the weights $w_{nm}^{(l)}$ and biases $b_n^{(l)}$. The recursion in Eq. (2) is terminated for $l = 1$ due to reaching its base $y_m^{(0)} = x_m$. For deep architectures, that is for $L \gg 1$, $Y$ 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, that is an error function of the network $Y$ 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 to naively apply chain- and product-rules.

In order to derive Taylor coefficients of $Y$ 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

$$D_{nm}^{(l,p)} = w_{nm}^{(l+1)} \frac{\partial^{p} a^{(l,m)}}{\partial x^{p}} (z_m^{(l)}), \quad (3)$$
which we refer to as the $nm^{\text{th}}$ matrix element of the $l^{\text{th}}$ layer propagator of order $p$. Since the last layer is usually activated via the identity, $a^{(L,n)} = \text{id}$, and has no bias, that is $b^{(L)}_n = 0$, we can write

$$Y_n(x) = \sum_{m=1}^{H_{L-1}} D^{(L-1,0)}_{nm}. \tag{4}$$

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

$$\frac{\partial^N Y_n}{\partial x_{k_1} \ldots \partial x_{k_N}} = \sum_{m=1}^{H_{L-1}} \frac{\partial^N D^{(L-1,0)}_{nm}}{\partial x_{k_1} \ldots \partial x_{k_N}}. \tag{5}$$

Applying the chain-rule throughout all layers yields for the first-order derivatives (see App. A on p. 23,)

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \Delta^{(l,1)}_{mk}, \quad \Delta^{(l,1)}_{mk} = \sum_{q_l=1}^{H_l} \sum_{q_1=1}^{H_1} \delta_{mq} w^{(1)}_{q_l k} \prod_{i=1}^{l-1} D^{(l,1)}_{q_i+1 q_i}. \tag{6}$$

In Eq. (6) we make two observations: First, a derivation increases the order of the propagator by one. Second, an additional factor $\Delta^{(l,1)}_{mk}$ is introduced, that impacts higher order derivatives of propagators. Defining the tensor elements

$$\Delta^{(l,p+1)}_{mk1\ldots pk_p+1} = \frac{\partial \Delta^{(l,p)}_{mk1\ldots pk_p}}{\partial x_{k_{p+1}}}, \tag{7}$$

we can express the $N^{\text{th}}$ derivative of the propagator as the following superposition by successively applying the rule mentioned in Eq. (6) and by absorbing the remaining derivatives by Eq. (7) (see App. A on p. 24),

$$\frac{\partial^N D^{(l,p)}_{nm}}{\partial x_{k_1} \ldots \partial x_{k_N}} = \sum_{c=1}^{N} D^{(l,p+c)}_{nm} \sum_{(\pi_i)_{i=1}^c} \prod_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk \sigma(1+\ldots+\pi_i)} \prod_{j=1}^{i} \sigma(1+\ldots+\pi_j) \ldots \sigma(1+\ldots+\pi_c). \tag{8}$$

Each summand in Eq. (8) depends on a higher order propagator. Here, the second sum runs over the set of all partitions of the number $N$ with length $c$,

$$\Pi^c_N = \left\{(\pi_i)_{i=1}^c \in \mathbb{N}^c \mid \sum_{i=1}^{c} \pi_i = N \land (\pi_1 \geq \ldots \geq \pi_c)\right\}. \tag{9}$$

The set of all partitions is, thereby, simply given by the union

$$\Pi_N = \bigcup_{c=1}^{N} \Pi^c_N.$$  

Lastly, the third sum runs over a subset

$$S[(\pi_i)_{i=1}^c] = \left\{\sigma \in S : \sum_{i=1}^{c} \pi_i = N \land (\pi_i)_{i=1}^c \right\} \left\{(\forall i \in \{1,\ldots,c\} : \sigma\left(1+\sum_{j=1}^{i-1} \pi_j\right) < \ldots < \sigma\left(\sum_{j=1}^{i} \pi_j\right)\right) \land \left(\forall i \in \{1,\ldots,c-1\} : (\pi_i = \pi_{i+1} \Rightarrow \sigma\left(1+\sum_{j=1}^{i-1} \pi_j\right) < \sigma\left(1+\sum_{j=1}^{i} \pi_j\right)\right)\right\}. \tag{10}$$

of the permutation group $S_N$, respecting the structure of the given partition. Tab. I contains all partitions, respective permutation subgroups and resulting propagator derivatives for $N = 1, 2, 3, 4$. 


| $N$ | $(\pi_i)_{i=1}^N \in \Pi_N$ | $S[(\pi_i)_{i=1}^N]$ | $\frac{\partial^N D_{nm}^{(l,p)}}{\partial x_{k_1} \ldots \partial x_{k_N}}$ |
|-----|----------------|----------------|---------------------------------|
| 1   | (1)             | {id}           | $D_{nm}^{(l,p+1)} \Delta_{mk_1}^{(l,1)}$ |
| 2   | (1, 1)          | {id}           | $+ D_{nm}^{(l,p+2)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_2}^{(l,1)}$ |
|     |                 |                | $+ D_{nm}^{(l,p+1)} \Delta_{mk_2}^{(l,2)}$ |
| 3   | (1, 1, 1)       | {id}           | $+ D_{nm}^{(l,p+2)} (\Delta_{mk_1}^{(l,2)} + \Delta_{mk_3}^{(l,2)} \Delta_{mk_2}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |
|     |                 | {id}           | $+ D_{nm}^{(l,p+1)} \Delta_{mk_2}^{(l,3)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |
|     | (2, 1)          | {id, (2, 3), (1, 2, 3)} | $+ D_{nm}^{(l,p+2)} (\Delta_{mk_1}^{(l,2)} \Delta_{mk_2}^{(l,1)} + \Delta_{mk_3}^{(l,2)} \Delta_{mk_4}^{(l,1)}$ |
|     |                  |                 | $+ D_{nm}^{(l,p+1)} \Delta_{mk_1}^{(l,4)} \Delta_{mk_2}^{(l,1)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |
|     | (2, 1)          | {id, (2, 3), (2, 4, 3)} | $+ D_{nm}^{(l,p+3)} \Delta_{mk_1}^{(l,2)} \Delta_{mk_2}^{(l,1)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |
|     |                  |                | $+ D_{nm}^{(l,p+2)} \Delta_{mk_1}^{(l,2)} \Delta_{mk_2}^{(l,1)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |
|     | (2, 2)          | {id, (2, 3), (2, 4, 3)} | $+ D_{nm}^{(l,p+2)} \Delta_{mk_1}^{(l,2)} \Delta_{mk_2}^{(l,1)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |
|     |                  |                | $+ D_{nm}^{(l,p+1)} \Delta_{mk_1}^{(l,4)} \Delta_{mk_2}^{(l,1)} \Delta_{mk_3}^{(l,1)} \Delta_{mk_4}^{(l,1)}$ |

TABLE I. All partitions $(\pi_i)_{i=1}^N \in \Pi_N$ and all corresponding permutation subsets $S[(\pi_i)_{i=1}^N]$ for $N = 1, 2, 3, 4$ required for computing the propagator derivatives $\frac{\partial^N D_{nm}^{(l,p)}}{\partial x_{k_1} \ldots \partial x_{k_N}}$ using Eq. (8).
What remains is to find an expression of the tensors $\Delta_{mk}^{(l,p)}$ in terms of propagators such that we can completely determine the partial derivatives of the propagators in Eq. (8). We therefore introduce the $mk$th matrix element of a vertex of order $p$ in the $l$th layer, acting as a weighted sum,

$$\Omega_{mk}^{(l,p)} q_{j_1} \cdots q_{j_p} = \sum_{q_1} H_1 \cdots \sum_{q_{l-1}} H_1 \sum_{j_1=1}^{l-1} \sum_{j_2=1}^{l-1} \cdots \sum_{j_{p-1}=1}^{l-1} \left( \prod_{i=1}^{l-1} D^{(i,1)}_{q_{i+1} q_i} \right) f_{q_{j_1} \cdots q_{j_p}}. \quad (11)$$

If $l - 1 \leq p$, 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. (6), vertices display the following behavior when exposed to a partial derivative,

$$\frac{\partial}{\partial x_{k_2}} \Omega_{mk}^{(l,p)} f_{q_{j_1} \cdots q_{j_p-1}} = \Theta(l - p) \Omega_{mk}^{(l,p-1)} f_{q_{j_1} \cdots q_{j_p-1}} D^{(j_p,2)}_{q_{j_p} q_{j_{p-1}}} \Omega_{mk}^{(l,p-1)} q_{j_p} q_{j_{p-1}}$$

$$+ \sum_{q_{j_1} q_{j_2}} \frac{\partial}{\partial x_{k_2}} f_{q_{j_1} \cdots q_{j_p-1}}. \quad (12)$$

Obviously, vertices of order $p$ in the $l$th 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 $\Delta_{mk}^{(l,1)}$ as a vertex of order zero in order to arrive at Eq. (12),

$$\Delta_{mk}^{(l,1)} = \sum_{(q_0)_{a=1}^l (j_0)_{b=1}^p} = (k_1 \bullet)_{m \cdot l}, \quad (13)$$

for which we choose the graphical representation of a single vertex. Applying Eq. (12) to Eq. (13) yields

$$\Delta_{mk}^{(l,2)} = \Theta(l - 1) \sum_{(q_0)_{a=1}^l (j_1)_{b=1}^p} D^{(j_1,2)}_{q_{j_1+1} q_{j_1}} \sum_{(q_0)_{a=1}^l (j_1)_{b=1}^p} q_{j_1} q_{j_2}$$

$$= \Theta(l - 1) \sum_{(q_0)_{a=1}^l (j_1)_{b=1}^p} D^{(j_1,2)}_{q_{j_1+1} q_{j_1}} (k_2 \bullet)_{q_{j_1} q_{j_2}}$$

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 two arrow heads, directed from the first to the second vertex,

$$\Delta_{mk}^{(l,2)} = \Theta(l - 1) (k_1 \longrightarrow \bullet k_2)_{m \cdot l}. \quad (14)$$
Applying Eq. \([12]\) once more to Eq. \([14]\) yields

\[
\Delta_{\begin{array}{c}mk_1k_2k_3 \\
\end{array}}^{(l,3)} = \Theta(l - 2) \sum_{q_{a_j}^{1}\left(j_1, j_2\right)} D_{q_{j_1}^{1}+1q_{j_2}^{1}}^{(j_1, 2)} D_{q_{j_2}^{1}+1q_{j_1}^{1}}^{(j_2, 2)} \sum_{q_{a_j}^{1}\left(j_1\right)} q_{j_1}^{1} k_2 \sum_{q_{a_j}^{1}\left(j_2\right)} q_{j_2}^{1} k_3 \\
+ \Theta(l - 1) \sum_{q_{a_j}^{1}\left(j_1\right)} D_{q_{j_1}^{1}+1q_{j_1}^{1}}^{(j_1, 3)} \sum_{q_{a_j}^{1}\left(j_1\right)} q_{j_1}^{1} k_2 \sum_{q_{a_j}^{1}\left(j_2\right)} q_{j_2}^{1} k_3 \\
+ \Theta(l - 1) \sum_{q_{a_j}^{1}\left(j_1\right)} D_{q_{j_1}^{1}+1q_{j_1}^{1}}^{(j_1, 2)} \sum_{q_{a_j}^{1}\left(j_1\right)} q_{j_1}^{1} k_2 \sum_{q_{a_j}^{1}\left(j_2\right)} q_{j_2}^{1} k_3 \\
= \Theta(l - 2) \begin{array}{c} k_1 \\
\end{array} \begin{array}{c} k_2 \\
\end{array} \begin{array}{c} k_3 \\
\end{array} \begin{array}{c} ml \\
\end{array} + \Theta(l - 1) \begin{array}{c} k_1 \\
\end{array} \begin{array}{c} k_2 \\
\end{array} \begin{array}{c} k_3 \\
\end{array} \begin{array}{c} ml \\
\end{array} \\
+ \Theta(l - 1) \begin{array}{c} k_1 \\
\end{array} \begin{array}{c} k_2 \\
\end{array} \begin{array}{c} k_3 \\
\end{array} \begin{array}{c} ml \\
\end{array} .
\]

(15)

This is a superposition of three different terms that we can each assign a different graph to. Note that the second term only contains one propagator of third order instead of two second-order propagators as in both other terms. In order to connect three vertices, we decide to graphically represent a propagator of third order via one incoming edge and two outgoing edges, each displayed with one arrowhead. It would not be of much use to continue with successively deriving higher order tensors as above. The general idea of how the \(\Delta_{\begin{array}{c}mk_1\ldots k_N \\
\end{array}}^{(l,N)}\) are structured and how the individual terms can be translated into graphs should be clear:

- \(\Delta_{\begin{array}{c}mk_1\ldots k_N \\
\end{array}}^{(l,N)}\) can be represented by a sum of weakly connected and directed tree-level graphs consisting of \(N\) vertices and between one and \(N - 1\) propagators. The weak connectivity expresses that there would be a path between each pair \(u, v\) of vertices, if each propagator was understood as an undirected edge. Each of these graphs if oriented away from the \(k_1^{st}\) vertex, which is designated as the root. In fact, the root is connected to each other vertex by exactly one path, meaning that each other vertex has exactly one incoming propagator, while the root has none. Graphs with these properties are also referred to as arborescences, see e.g. Ref. \([21]\).

- Propagators of order \(n\) are represented by one incoming edge and \(n - 1\) outgoing edges. The number of arrow heads is equal to the order of that propagator.

- A term of the structure

\[
\ldots \sum_{(q_{a_1}^{1})_{i=1}^{(1)}} q_{j_1}^{(1)} k_{x_1} \ldots \sum_{(q_{a_1}^{2})_{i=2}^{(1)}} q_{j_2}^{(1)} k_{x_2} \ldots \sum_{(q_{a_1}^{n})_{i=n}^{(1)}} q_{j_n}^{(1)} k_{x_n} \ldots
\]

indicates that it is the \(t^{th}\) propagator originating in the \(k_{x_1}^{st}\) vertex that establishes a connection to the \(k_{x_2}^{nd}, \ldots, k_{x_{n-1}}^{th}\) 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.
• Derivatives of saturated vertices vanish, as seen in Eq. (12). Thus, it depends on the layer \( l \) propagators and vertices are considered for, whether a certain arborescence contributes or not. 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 \( \alpha \). The appearance of internal vertices and propagators of order three or higher decreases \( \alpha \).

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 each vertex, we therefore enumerate outgoing propagators starting with 1. If \( A_{ij} = 0 \), there is no connection from the \( i \)th vertex to the \( j \)th vertex. Otherwise, it is the \( A_{ij} \)th propagator originating in the \( i \)th vertex that establishes this connection. 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 non-zero entry in each but the first column of \( A \). The set of such triangular matrices over \( \mathbb{K} \) is given by

\[
\mathbb{T}_N = \{ M \in \mathbb{K}^{N \times N} \mid \forall i \in \{1, \ldots, N\} \forall j \in \{1, \ldots, i\} : M_{ij} = 0 \\
\land (N > 1 \Rightarrow \forall j \in \{2, \ldots, N\} \exists i = 1, \ldots, N-1 : M_{ij} \neq 0) \}.
\]

Then the set of all allowed adjacency matrices for \( N \) vertices is the following subset of \( \mathbb{T}_N \):

\[
\mathbb{A}_N = \{ A \in \mathbb{T}_N \mid \forall i \in \{1, \ldots, N-1\} \forall j \in \{2, \ldots, N\} : (A_{ij} > 0 \Rightarrow \exists j' < j : A_{ij'} = A_{ij} - 1) \} \tag{16}
\]

Counting the appearances of \( A_{ij} > 0 \) in the \( i \)th 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 \( i \)th 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 \( \alpha(A) \) are internal vertices: \( \alpha(A) \) is decreased by 1 for each internal vertex, or, in terms of adjacency matrices, for each non-zero line but the first one. This behavior is entirely described by the expression

\[
\alpha(A) = \Theta \left( \sum_{i=1}^{N} \sum_{j=1}^{N} A_{ij} \right) + \sum_{i=1}^{N} \left( \sum_{j=1}^{N} \Theta(A_{ij}) - \Theta \left( \sum_{j=1}^{N} A_{ij} \right) \right) \tag{17}
\]

If \( l \) reaches or undershoots \( \alpha(A) \), the corresponding arborescence is saturated and does not contribute to \( \Delta_{mk_1\ldots k_N}^{(l,N)} \). Tab. 1 lists example arborescences and corresponding adjacency matrices \( A \) as well as saturation thresholds \( \alpha(A) \). The only thing left for expressing \( \Delta_{mk_1\ldots k_N}^{(l,N)} \) solely in terms of propagators and biases is an analytical representation \( \delta_{mk_1\ldots k_N}^{(l,N)}(A) \) of an individual arborescence with \( N \) vertices for a given adjacency matrix \( A \). For its formulation, we use the function

\[
\beta_c(A) = \sum_{i=1}^{N} i \cdot \Theta(A_{ic}) \tag{18}
\]

that determines the line in which the entry in the \( j \)th column is non-zero. Since there is no antecedent propagator to the root of an arborescence, Eq. (18) vanishes for \( j = 1 \).
| MLP-Arborescence | $A$ | $\alpha(A)$ |
|------------------|-----|-------------|
| $k_1 \bullet$    | (0) | 0           |
| $k_1 \rightarrow k_2$ | $\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ | 1 |
| $k_1 \rightarrow k_2 \rightarrow k_3$ | $\begin{pmatrix} 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ | 2 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 1 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 2 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 2 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 2 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 1 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 1 |
| $k_1 \rightarrow k_2 \rightarrow k_3 \rightarrow k_4$ | $\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$ | 1 |

**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 root. From here, we can deduce that the maximum saturation threshold among all arborescences with $N$ vertices is $N - 1$. 
Using Eq. (18) and the previous observations, we write

\[
\delta_{mk_1...k_N}^{(l,N)} (A) = \sum_{q_i^{(0)}} \sum_{j_{A_{\beta_1}}^{(0)}} \prod_{c=1}^{N} \delta_{m_q^{(0)} j_{A_{\beta_1}}^{(0)}}^{(l)} \times \bigg( \prod_{b=1}^{\max_r(A_{cr})} D \left( j_b^{(c)} + 1, \sum_{j_{b}^{(c)}} \delta_{b_{A_{\beta_1}}}^{(c)} \right) \bigg).
\]

Eq. (19) may appear very intimidating at first sight, but its individual terms are easy to interpret and to recognize in the summands of Eqs. (13), (14) and (15):

- The expression \( \max_r(A_{cr}) \) corresponds to the number of all propagators originating in the \( c \)th vertex. Thus, all these propagators are collected by the product

\[
\prod_{b=1}^{\max_r(A_{cr})} D \left( j_b^{(c)} + 1, \sum_{j_{b}^{(c)}} \delta_{b_{A_{\beta_1}}}^{(c)} \right).
\]

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

- The \( c \)th vertex of the arborescence is denoted by the expression

\[
\bigg( \prod_{b=1}^{\max_r(A_{cr})} D \left( j_b^{(c)} + 1, \sum_{j_{b}^{(c)}} \delta_{b_{A_{\beta_1}}}^{(c)} \right) \bigg)
\]

For each of the \( \max_r(A_{cr}) \) propagators that originate in the \( c \)th vertex, there is a summation index \( j_b^{(c)} \). It is the \( \beta_1(A) \)-th vertex, whose \( A_{\beta_1(A)} \) propagator leads to the \( c \)th vertex. This is the reason, why we sum over the \( q_j^{(\beta_1(A))} k_c \)-th matrix element of the \( c \)th vertex in the \( J_{A_{\beta_1(A)}}^{(\beta_1(A))} \)-th layer.

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

\[
\sum_{q_i^{(0)}} \sum_{j_{A_{\beta_1}}^{(0)}} \prod_{c=1}^{N} \delta_{m_q^{(0)} j_{A_{\beta_1}}^{(0)}}^{(l)}.
\]

Using Eq. (19) and taking the corresponding saturation thresholds given in Eq. (17) into account, we can represent \( \Delta_{mk_1...k_N}^{(l,N)} \) as the following sum over all adjacency matrices \( A \in \mathbb{A}^N \) (see App. A on p. 28):

\[
\Delta_{mk_1...k_N}^{(l,N)} = \sum_{A \in \mathbb{A}^N} \Theta(l - \alpha(A)) \delta_{mk_1...k_N}^{(l,N)} (A).
\]

Finally, Eq. (20) can be inserted into Eq. (8), which is required for computing the Taylor coefficients of \( Y \) as shown in Eq. (5). As we approach the end, we want to motivate two more notations. At first, considering
Eq. (8), 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 $\partial^N D_n^m / (\partial x_{k_1} \ldots \partial x_{k_N})$ consists of arborescences that each contain different vertices. For simplicity, let us assume we are given a product of two arborescences covering the vertices $k_{i_1}, \ldots, k_{i_x}$ and $k_{j_1}, \ldots, k_{j_y}$, respectively, with $I \cap J = \emptyset$ for $I = \{i_1, \ldots, i_x\}$ and $J = \{j_1, \ldots, j_y\}$. Then we write the product as one disconnected graph with each former arborescence being a connected component and, as such, a subgraph,

\[
\begin{pmatrix}
k_{i_1} \\
\vdots \\
k_{i_x}
\end{pmatrix}
\begin{pmatrix}
k_{j_1} \\
\vdots \\
k_{j_y}
\end{pmatrix}
= \begin{pmatrix}
k_{i_1} \\
\vdots \\
k_{i_x}
\end{pmatrix}
\begin{pmatrix}
k_{j_1} \\
\vdots \\
k_{j_y}
\end{pmatrix}
. \tag{21}
\]

Eq. (21) allows expressing Eq. (8) and therefore $\partial^N Y / (\partial x_{k_1} \ldots \partial x_{k_N})$ as a sum of graphs, whose connected components are arborescences. If and only if that graph has exactly one connected component, it is an arborescence itself as the ones in Tab. IV. The saturation threshold of a disconnected graph is simply the maximum saturation threshold among all its connected components. The Taylor coefficients up to third order then turn out as

\[
\begin{align*}
\frac{\partial Y_n}{\partial x_{k_1}} \bigg|_{x = x_0} &= \sum_{m=1}^{H_{L-1}} D_{nm}^{(L-1,1)} \Theta(L - 1) \left( k_{1} \bullet m, L-1 \right) \bigg|_{x = x_0} \\
\frac{\partial^2 Y_n}{\partial x_{k_1} \partial x_{k_2}} \bigg|_{x = x_0} &= \sum_{m=1}^{H_{L-1}} \left[ D_{nm}^{(L-1,1)} \Theta(L - 2) \left( k_{1} \bullet k_{2} m, L-1 \right) + D_{nm}^{(L-1,2)} \Theta(L - 1) \left( k_{1} \bullet k_{2} m, L-1 \right) \right] \bigg|_{x = x_0} \\
\frac{\partial^3 Y_n}{\partial x_{k_1} \partial x_{k_2} \partial x_{k_3}} \bigg|_{x = x_0} &= \sum_{m=1}^{H_{L-1}} \left\{ D_{nm}^{(L-1,1)} \left[ \Theta(L - 3) \left( k_{1} \bullet k_{2} k_{3} m, L-1 \right) + \Theta(L - 2) \left( k_{1} \bullet k_{2} k_{3} m, L-1 \right) \right] + D_{nm}^{(L-1,2)} \left[ \Theta(L - 2) \left( k_{1} \bullet k_{2} k_{3} m, L-1 \right) + \Theta(L - 2) \left( k_{1} \bullet k_{2} k_{3} m, L-1 \right) \right] \\
&\quad + \Theta(L - 2) \left( k_{1} \bullet k_{2} k_{3} m, L-1 \right) \right\} \bigg|_{x = x_0}
\end{align*}
\tag{22-24}
\]

Although graphs, whose connected components are arborescences, are in general not invariant under vertex permutations, invariance is recovered by summing over all vertices, as it is the case in the Taylor series. This collapses graphs, that are related via vertex permutations (e.g. graphs 4, 5 and 6 in Eq. (24)), onto one graph and correspondingly introduces symmetry factors to the naive Taylor series.
The following short-hand notation naturally agrees with vertex permutation symmetry and proves useful for the Taylor series:

$$
\begin{align*}
\begin{bmatrix}
\vdots \\
\vdots \\
\vdots \\
\end{bmatrix}^{ml, x_0}
&= d \sum_{k_1} d \sum_{k_i} d \sum_{k_N} \begin{bmatrix}
\vdots \\
\vdots \\
\vdots \\
\end{bmatrix}^{ml, x_0} \\
&= \prod_{i=1}^{N} (x - x_0)^{k_i}.
\end{align*}
$$

(25)

Here, each external leg stands for the contraction of the graph with some component of the displacement $(x - x_0)$ to the expansion point $x_0$. Since $L > 0$ by construction, assigning to the empty graph, that is $A \in \emptyset$, the value

$$
\Delta^{(L-1)}(A) = \Theta(L - 0) ( )_{m,L-1} = 1
$$

and using the short-hand notation from Eq. (25) finally yields the interesting series representation:

$$
Y_n(x) = \sum_{m=1}^{H_{L-1}} D^{(L-1,0)}_{nm} \Theta(L - 0) ( )_{m,L-1,x_0}
\begin{align*}
+ \sum_{m=1}^{H_{L-1}} &D^{(L-1,1)}_{nm} \left[ \Theta(L - 1) ( )_{m,L-1,x_0} + \frac{\Theta(L - 2)}{2} ( )_{m,L-1,x_0} \\
&+ \frac{\Theta(L - 2)}{6} ( )_{m,L-1,x_0} + \frac{\Theta(L - 2)}{6} ( )_{m,L-1,x_0} + \cdots \\
+ \frac{\Theta(L - 3)}{6} ( )_{m,L-1,x_0} + \cdots \right]
\end{align*}
\begin{align*}
+ \sum_{m=1}^{H_{L-1}} &D^{(L-1,2)}_{nm} \left[ \frac{\Theta(L - 1)}{2} ( )_{m,L-1,x_0} + \frac{\Theta(L - 2)}{2} ( )_{m,L-1,x_0} \\
&+ \cdots \right]
\end{align*}
\begin{align*}
+ \sum_{m=1}^{H_{L-1}} &D^{(L-1,3)}_{nm} \left[ \frac{\Theta(L - 1)}{6} ( )_{m,L-1,x_0} + \cdots \right]
\end{align*}
\begin{align*}
+ \cdots
\end{align*}
$$

(26)
This reveals the following structure of the component \( Y_n \): Graphs that contain up to \( N \) vertices contribute to the \( N^{th} \) Taylor approximation of \( Y_n \). Thus, a graph with \( c \) connected components is weighted with an additional factor \( D_n^{(L-1,c)} \). The deeper \( Y \), that is the larger \( L \), the more graphs contribute to a given order of the Taylor expansion due to overshooting the corresponding saturation threshold.

### III. THE BORN SERIES AS A TAYLOR SERIES

In the following, we exemplarily decide to apply the above established framework of a Neural Network Perturbation Theory to low-energy two-body scattering. All relevant observables can be parameterized by the S-wave scattering length \( a_0 \), which conversely provides \( a_0 \) as a meaningful and distinguished target and the corresponding potential \( V \) as a suitable input in a supervised-learning-scenario. By analyzing patterns in their first weight matrix, it could be shown in Ref. [11], that under similar training conditions, MLPs develop a quantum perturbation theory during training and approximate scattering lengths in terms of Born approximations. Unfortunately, the arguments given there strongly rely on the fact, that L2-regularization and ReLU activations are used. Considering the Taylor expansion of \( a_0 \), instead, provides an analytical interpretation that is independent of the particular MLP-architecture.

The representability as a Born series is carried over from the T-matrix to the S-wave scattering length. Assuming that potentials \( V \) vanish after a finite range \( \rho \), the Born series for \( a_0 \) can then be written as

\[
a_0 = 4\pi^2 \mu \langle 0 | T | 0 \rangle \\
= 4\pi^2 \mu \langle 0 | V | 0 \rangle + 4\pi^2 \mu \langle 0 | VG_0V | 0 \rangle + 4\pi^2 \mu \langle 0 | VG_0VG_0V | 0 \rangle + \ldots \\
= 2\mu \int_0^\rho dr \, r^2 V(r) - 2\mu^2 \int_0^\rho dr \int_0^\rho dr' \, V(r)V(r')rr'(r + r' - |r - r'|) + \ldots
\]

(27)

with the reduced mass \( \mu \) of the two-body system. For sufficiently shallow potentials, Eq. (27) can be treated perturbatively by neglecting higher order summands, which finally results in Born approximations of \( a_0 \). Correspondingly, the first two Born approximations are given by \( a_0^{(1)} \) and \( a_0^{(1)} + a_0^{(2)} \), respectively, with the Born terms

\[
a_0^{(1)} = 2\mu \int_0^\rho dr \, r^2 V(r)
\]

(28)

and

\[
a_0^{(2)} = -2\mu^2 \int_0^\rho dr \int_0^\rho dr' \, V(r)V(r')rr'(r + r' - |r - r'|).
\]

(29)

An MLP can only process data in a finite dimensional vector space. When training an MLP \( A_i \) to predict dimensionless scattering lengths \( a_0/\rho \) based on dimensionless input potentials, we therefore need to discretize potentials at first. A dimensionless, discretized potential can be understood as a vector \( U \in \mathbb{R}^d \) with \( d \) the degree of discretization and the component \( U_n = -2\mu \rho^2 V(n/d) \) corresponding to the \( n^{th} \) potential step. It is obvious that \( d \) must be chosen sufficiently large such that the discretization error becomes negligible. For simplicity, we only consider potentials with \( U_n \geq 0 \), that is attractive potentials. In terms of discretized potentials, the Born terms in Eqs. (28) and (29) reduce to the sums

\[
a_0^{(1)} \longrightarrow -\frac{\rho}{\pi^2} \sum_{n=0}^{d-1} n^2 U_n
\]

(30)
and

\[ a_0^{(2)} \longrightarrow -\frac{\rho}{2d^5} \sum_{n=0}^{d-1} \sum_{m=0}^{d-1} U_n U_m n m (n + m - |n - m|). \]  

(31)

Eqs. (30) and (31) strongly suggest to understand the Born series in Eq. (27) for discretized potentials as a Taylor series with respect to the expansion point \( U_0 = 0 \). Vice versa, the more precise the predictions of \( A_i \) become, the closer we expect its leading-order Taylor coefficients to approach the leading-order Taylor coefficients of \( a_0/\rho \). For the Taylor coefficients we could then ideally, that is if the loss would vanish, observe the equalities

\[ \sum_{m=1}^{H_{L-1}} D_{1m}^{(L-1,0)} \Theta(L - 0) \bigg|_{U=0} = 0, \]

(32)

\[ \sum_{m=1}^{H_{L-1}} D_{1m}^{(L-1,1)} \Theta(L - 1) (k_1 \bullet) m_{,L-1} \bigg|_{U=0} = -\frac{k_1^2}{d^3}, \]

(33)

and

\[ \sum_{m=1}^{H_{L-1}} \left[ D_{1m}^{(L-1,2)} \Theta(L - 2) (k_1 \bullet \bullet k_2) m_{,L-1} + D_{1m}^{(L-1,1)} \Theta(L - 1) (k_1 \bullet \bullet k_2) m_{,L-1} \bigg|_{U=0} = -\frac{1}{d^3} k_1 k_2 (k_1 + k_2 - |k_1 - k_2|). \]  

(34)

IV. NETWORK AND TRAINING DETAILS

In order to enhance precision even further and to reduce statistical noise, we consider an ensemble \( A \) of \( N = 20 \) MLPs \( A_i \), instead of working with a single MLP. Each member \( A_i \) consists of one output layer and nine linear \( 64 \times 64 \) layers, that each are activated via the GELU activation function [22]. GELU is smooth in the origin and bypasses the vanishing-gradients-problem, which makes it particularly interesting for such deeper architectures. Finally, the output of the ensemble \( A \) is simply given as the mean of all individual member outputs,

\[ A(U) = \frac{1}{N} \sum_{i=1}^{N} A_i. \]

(35)

Due to linearity, we observe the same relation between the respective derivatives of all \( A_i \) and \( A \),

\[ \frac{\partial^n A(U)}{\partial U_{k_1} \cdots \partial U_{k_n}} \bigg|_{U=0} = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial^n A_i(U)}{\partial U_{k_1} \cdots \partial U_{k_n}} \bigg|_{U=0}. \]

(36)

We decide to discretize potentials into \( d = 64 \) steps. Thereby, discrete potentials are generated as Gaussian random walks. In fact, we generate two training sets \( T_1, T_2 \) with lengths \( |T_{1,2}| = 4 \times 10^4 \) and corresponding test sets \( t_1, t_2 \) with lengths \( |t_{1,2}| = 4 \times 10^3 \). Through downsampling, each of the four target distributions is uniform in and limited to the narrow interval \([-0.01, 0]\) containing the origin. While the targets in \( T_1 \) and \( t_1 \) are simply the S-wave scattering lengths \( a_0 \) that belong to the given potentials, the targets in \( T_2 \) and \( t_2 \) are given by \( a_0 + (1/d^3) \sum_{n=0}^{d-1} n^2 U_n \), which are scattering lengths of which the first...
Born term has been substracted. Here, we derive scattering lengths for discretized potentials using the transfer matrix method, see Ref. [23]. While the first-order Born term dominates the targets in \( T_1 \) and \( t_1 \), it is the second-order Born term that is of leading-order within the targets of \( T_2 \) and \( t_2 \).

Using PyTorch [24], two ensembles \( A \) and \( B \) as described in Eq. (35) are trained by training their members one after another, one using the data sets \( (T_1, t_1) \) and the other one with respect to \( (T_2, t_2) \). The target range is scaled to the interval \([0, 1]\) by multiplying a factor \(-10^2\) to the targets. Therefore, predictions and derivatives must be rescaled by multiplying \(-10^{-2}\) afterwards. Weights in both ensembles are initialized using He-initialization [25]. We use mini-batch learning with batch-size 10 and train over 20 epochs with an exponentially decaying learning-rate schedule

\[
\eta_t = 10^{-3}e^{-t/2}
\]

starting with \( t = 0 \), while using the RMSprop-optimizer, see Ref. [26], and abandoning L2-regularization. During training we minimize the mean-squared-error loss (MSELoss). Another useful performance measure is the mean-average-percentage-error (MAPE),

\[
\begin{align*}
L_{A,t} &= \frac{1}{|\mathcal{E}|} \sum_{(U,a) \in \mathcal{E}} \left| \frac{A_i(U) - a}{a} \right|, \\
L_{A,t} &= \frac{1}{|\mathcal{E}|} \sum_{(U,a) \in \mathcal{E}} \left| \frac{1}{N} \sum_{i=1}^{N} A_i(U) - a \right|
\end{align*}
\]

where \((U, a)\) is a pair of a discrete potential and the corresponding target \( a \) from the test set \( t \). Finally, when finishing an epoch, the state of the MLP \( A_i \) is indeed kept as the starting point for the subsequent epoch, but only saved to a file, if its loss is less than all previous losses for that \( A_i \).

After the two training procedures, we observe the two ensembles to have the satisfactorily low losses \( L_{A,t_1} = 3.034 \times 10^{-2}\% \) and \( L_{B,t_2} = 5.009 \times 10^{-1}\% \), which indicates that \( A \) and \( B \) have learned to accurately predict \( a_0 \) and \( a_0 + (1/d^3) \sum_{n=0}^{d-1} n^2 U_n \), respectively. This was to be expected, as the used MLP-architecture is very deep in relation to the simplicity of the problem. However, it is not their performance on the data sets itself, but the quality of their Taylor coefficients, that is linked to Born approximations and that is, therefore, as important to their interpretability as it is the actual focus of the following analysis.

V. FIRST-ORDER BORN TERM

From the ensemble \( A \), we expect its analytical first-order Taylor coefficients to agree with the numerical derivatives and the theoretically expected coefficients, that is

\[
\frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{H_{L-1}} D_{lm}^{(L-1,1)} \theta(L - 1) \frac{[A_i]}{[A_i]} (k_1 \bullet [A_i])_{m,L-1} \Big|_{U=0} \approx \frac{A(s e_{k_1}) - A(0)}{s} \approx -\frac{k_1^2}{d^3}
\]

The superscript \([A_i]\) in the analytical coefficients points out that the respective propagators and vertices are computed for the weights and biases of the member \( A_i \). For the numerical derivatives we use the step size \( s = 10^{-2} \). As we are given an ensemble, we estimate errors of ensemble quantities via the standard deviation of the corresponding member quantity distribution. To give an example, we observe the mean \( \mu = -2.503 \times 10^{-7} \) and standard deviation \( \sigma = 5.602 \times 10^{-7} \) for the distribution \( \{A_1(0), \ldots, A_N(0)\} \), such that we may estimate

\[
A(0) = -2.503(5602) \times 10^{-7}.
\]

Since there is no constant term in the Born series, such an axis intercept, that is by several orders of magnitude smaller than average scattering lengths in \( T_1 \) and \( t_1 \), was to be expected. The estimated error to the theoretically expected intercept \( A(0) = 0 \) is less than 1\( \sigma \).

FIG. 1. The analytically derived Taylor coefficients (blue) and the expected coefficients $-n^2/d^3$ (red) are shown in Fig. a). For the analytical Taylor coefficients we have used Eq. (33) and averaged according to Eq. (36). As can be seen, the two curves are in very good agreement with each other. Thus, the ensemble has learned to apply the first-order Born approximation to data. In Fig. b) the difference (black) between the analytical and numerical Taylor coefficients is displayed. With the difference being three orders of magnitude smaller than the actual coefficients, we can finally verify that Eq. (20) is a valid tool for computing derivatives of MLPs.

Now we can finally move to the first Born term. In Fig. a) the analytically and numerically computed Taylor coefficients are compared to the theoretical Taylor coefficients $-k_1^3/d^3$. As can be seen, the three curves agree very well with each other, which for one implies, that the ensemble applies a very good approximation of the first Born approximation to input data, and for another shows, due to the similarity to the numerical derivatives, that Eq. (20) is a valid tool for computing derivatives of MLPs.

Quantitatively, the quality of the analytical Taylor coefficients of $\mathcal{A}$ can be estimated by fitting a model $\alpha_{\mathcal{A}} k_1^2$ to the first-order Taylor coefficients, that are analytically computed using Eq. (34) for each member $\mathcal{A}_i$ and later averaged over the whole ensemble. Considering the distribution of all $\alpha_{\mathcal{A}_i}$, finally yields

$$\alpha_{\mathcal{A}} = -3.869(30) \times 10^{-6},$$

which deviates less than $2\sigma$ from the theoretical value $\alpha = -3.815 \times 10^{-6}$. As can be seen, the error $\Delta \alpha_{\mathcal{A}} = 3.0 \times 10^{-8}$ is by two orders of magnitude lower, which indicates that the Taylor coefficients have expectedly a clear quadratic behavior.

VI. SECOND-ORDER BORN TERM

For the same ensemble $\mathcal{A}$, we could naively try to reproduce the second-order Born term in the same fashion as presented above. This would involve fitting a model $\beta k_1 k_2 (k_1 + k_2 - |k_1 - k_2|)$ to the second-order Taylor coefficients, that are analytically computed using Eq. (34) for each member $\mathcal{A}_i$ and later averaged over the whole ensemble. However, the expected fit parameter $\beta = -1/d^5 = -9.313 \times 10^{-10}$ is by four orders of magnitude smaller than $\alpha$. As the contribution of the second-order Born term is following much smaller than the contribution of the first order, it is not guaranteed, that the $\mathcal{A}_i$ accurately reproduce the second-order. This can also be understood by the loss reduction benefiting considerably more from adapting to the first-order Born term during training, while the overall performance does not significantly suffer from a badly approximated second-order Born term. Indeed, we find the fit parameter $\beta_{\mathcal{A}} = -1.76(1134) \times 10^{-9}$, which has an incredibly large variance and thus miserably fails at repro-
ducing the second Born term. Therefore, it is much more reliable to consider the second ensemble \( B \) that has been trained under the same conditions with respect to the data sets \( T_2 \) and \( t_2 \). The leading term in their targets \( a_0 + (1/d^3) \sum_{n=0}^{d-1} n^2 U_n \) is now the second-order Born term. At first, we observe

\[
B(0) = 6.082(4825) \times 10^{-6} \quad \text{and} \quad \alpha_B = -9.173(692) \times 10^{-8}. \tag{39}
\]

Again, the axis intercept is negligibly small. In contrast to the previous ensemble \( A \), the fit parameter of the first-order Taylor coefficients is by two orders of magnitude smaller than \( \alpha \) from construction, which is just another evidence, that the second-order Born term is indeed of leading order in \( B \). The second-order Taylor coefficients are analytically computed as

\[
\frac{\partial^2 B}{\partial x_{k_1} \partial x_{k_2}} \bigg|_{\text{ana}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{H_{L-1}} \left[ D^{(L-1,1)}_{1m}[B_i] \Theta(L - 2)(k_1 \rightarrow \rightarrow \rightarrow k_2)_{m,L-1}^{[B_i]} \right. \\
+ \left. D^{(L-1,2)}_{1m}[B_i] \Theta(L - 1)(k_1 \rightarrow \rightarrow \rightarrow k_2)_{m,L-1}^{[B_i]} \right] \bigg|_{U = 0}.
\tag{40}
\]

Due to \( L = 10 \), both arborescences overshoot their saturation threshold and therefore contribute to the coefficients. These analytical coefficients can be compared to the numerical derivatives

\[
\frac{\partial^2 B}{\partial x_{k_1} \partial x_{k_2}} \bigg|_{\text{num}} = \frac{B(s e_{k_1} + s e_{k_2}) - B(s e_{k_1}) - B(s e_{k_2}) + B(0)}{s^2} \tag{41}
\]

and to the theoretically expected coefficients

\[
\frac{\partial^2 B}{\partial x_{k_1} \partial x_{k_2}} \bigg|_{\text{theo}} = -\frac{1}{d^3} k_1 k_2 (k_1 + k_2 - |k_1 - k_2|). \tag{42}
\]

An overview of these is given in Fig. 2. The shapes of the graphs in Figs. 2b) and 2b) slightly deviate from that in Fig. 2a), especially due to the distinctive diagonal that does not appear in theory. Its origin presumably lies in the absence of a proper weight regularization: Due to overfitting there could be residual oscillations near the origin that are not completely averaged out by ensembling, which implies a slightly larger curvature and, thus, diagonal elements in the Hessian. Apart from that, all three plots are sufficiently similar such that we may confidently claim \( B \) to have approximated the second-order Born term well, which is also reflected in the low loss of the \( B \). We measure the quality of the analytical Taylor coefficients by fitting to them the model

\[
\beta_B, k_1 k_2 (k_1 + k_2 - |k_1 - k_2|),
\]

which has already been mentioned above. Considering the distribution \{\( \beta_B \)\}, we obtain the fit parameter

\[
\beta_B = -7.176(1267) \times 10^{-10}. \tag{43}
\]

Unfortunately, its error is of the same order of magnitude, for which the noisy artifacts and the fact that the behavior of the ensemble’s Taylor coefficients is not purely \( k_1 k_2 (k_1 + k_2 - |k_1 - k_2|) \), as both can be seen in Fig. 2b), are responsible. These again originate in training- and architecture-details and probably could be alleviated by a suitable hyperparameter optimization. Nonetheless, \( \beta_B \) deviates less than \( 2 \sigma \) from the theoretical parameter \( \beta = -9.313 \times 10^{-10} \), which is additional evidence that \( B \) approximates the second-order Born term and applies it to new input potentials.
FIG. 2. Analytically (a) and numerically (b) derived second-order Taylor coefficients of the ensemble $B$, that have been computed using Eqs. (40) and (41), are compared to the theoretically expected Taylor coefficients given in Eq. (42) (c). Especially for the numerical coefficients we observe noisy artifacts that might originate in choosing a relatively large step size of $s = 0.01$ or in a missing regularization. The shapes of the graphs in a) and b) also slightly deviate from that in c), especially due to the distinctive diagonal. Nonetheless, all three graphs display the same behavior for the most part. Thus, the ensemble $B$ has learned to predict targets by computing the second-order Born term of the input data. In addition, this provides further validation of Eq. (20).
FIG. 3. Machine-learned, effective second-order Born approximation (blue, dashed) based on the first-order Taylor coefficients of $A$ and the second-order Taylor coefficients of $B$, compared to the predictions $A(U_i)$ (green, dotted) and the corresponding true scattering lengths $a_0$ (red, solid). These are evaluated as follows: At first we randomly generate three different potential shapes $n_i \in \mathbb{R}^{64}$. For each of these shapes we generate a set of 100 equidistant potentials $U_i = \|U_i\| n_i$ with magnitudes $\|U_i\| \in [0, \ldots, 144]$. For each of these potentials the Born approximation and true scattering lengths are evaluated and plotted above. Based on the pairwise deviations, we observe that the range of validity of the Born approximation and $A$ is by one order of magnitude larger than the actual training range. Both reliably predict scattering lengths up to $a_0 \approx -0.2$.

Finally, we can combine the first-order Taylor coefficients of $A$ and the second-order Taylor coefficients of $B$ to an effective, machine-learned second-order Born approximation. The comparison of that effective Born approximation with the predictions of $A$ and the true scattering lengths, displayed in Fig. 3, shows that its range of validity we estimate to be approximately $[-0.2, 0]$ is by one order of magnitude larger than the original training range. It appears that the second-order Born approximation does not suffice beyond that regime, such that the third-order Born term has to be included in order to predict scattering lengths for deeper potentials.

VII. DISCUSSION AND OUTLOOK

As soon as an NN can be represented as an arbitrarily deep MLP, we have proven that partial derivatives of any order can be analytically derived by Eqs. (8) and (20). Instead of expressing these in terms of the MLP’s weights and biases directly, we use the notion of propagators and vertices, instead, which is motivated by Feynman diagrams in a quantum field theory and makes underlying combinatorics more manageable. Given an expansion point $x_0$ in input space, their partial derivatives serve as coefficients for a Taylor series. Considering small perturbations around $x_0$, higher order terms can be neglected, which defines a perturbation theory and leaves us with a Taylor approximation. Thereby, changes in the MLP’s output can be easily predicted without explicit evaluations of that MLP, which is an important earmark of local interpretability.

We train two ensembles $A$ and $B$, each consisting of $N = 20$ MLPs with $L = 10$, to predict S-wave scattering lengths and scattering lengths from which the first Born approximation has been substracted, respectively, for given discretized potentials. We easily derive the individual first-order Taylor coefficients of $A$ applying the above framework and find them to behave similar to $-(k_1)^2/d^3$. This indicates that $A$ approximates scattering lengths even beyond the training regime by the first-order Born approximation. However, the second-order Taylor coefficients of $A$ do not reliably reproduce the second-order Born term, since the contribution of the second order is by several orders of magnitude lower than that of the first order.
Errors occurring here would, therefore, hardly affect the MLP’s loss. Instead we consider the ensemble \( B \), as the leading contribution in its predictions can be shown to accurately approximate the second-order Born term. For both ensembles, the similarity of the analytical, numerical and theoretical Taylor coefficients not only implies that the respective ensembles apply Born approximations, but also serves as a practical verification of the established MLP perturbation theory.

At this point one could of course argue that it would have been much more convenient to simply train an NN \( Y : \mathbb{R}^{64} \to \mathbb{R} \) that is just the sum of one linear layer \( l \) and one bilinear layer \( B \), that is

\[
Y(U) = l \cdot U + \frac{1}{2} U^\top B U.
\]

Using this architecture instead of deep MLPs would not only reduce the computational effort significantly, but also would have imposed some desired properties like \( Y(0) = 0 \) and simultaneously learning the first- and second-order Born terms. Note that \( Y \) in this case is a second-order Taylor approximation by itself, which allows to directly read off Taylor coefficients instead of deriving them first, as performed in our analysis. However, such an NN is not an universal approximator, as it violates the universal approximation theorem, and, therefore, will fail in reproducing more negative scattering lengths \( a_{0} \lesssim -0.2 \) for deeper potentials. This is because the third-order Born-term could be shown to be no longer negligible in this regime, but cannot be approximated by \( Y \) due to behaving like \( O(U^3) \). Therefore, using such an intrinsically interpretable architecture may indeed simplify the analysis, but must be well justified for the particular case.

The presented approach of establishing local interpretability by considering Taylor approximations on NNs in vicinity of given expansion points is a typical example of a proxy method according to Ref. [18]. As such, it just provides a post-hoc interpretation of the networks predictions, based on approximations and thus deviations from the actual predictions. 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 are either intrinsically interpretable or can be brought in an interpretable representation, see Ref. [18]. At the cost of a prediction-interpretation-tradeoff, 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. Nevertheless, the local post-hoc interpretations of \( \mathcal{A} \) and \( \mathcal{B} \) presented here are still eminently insightful, especially since they explain the observations made in [11] regarding the development of a perturbation theory for S-wave scattering lengths in an analytical and, up to the requirement of being an MLP, architecture-independent manner.

ACKNOWLEDGEMENTS

We acknowledge partial financial support from the Deutsche Forschungsgemeinschaft (TRR 110, “Symmetries and the Emergence of Structure in QCD”), Further support was provided by the Chinese Academy of Sciences (CAS) President’s International Fellowship Initiative (PIFI) (grant no. 2018DM0034) and by VolkswagenStiftung (grant no. 93562).

[1] P. Mehta and D. J. Schwab, arXiv:1410.3831 [stat.ML] (2014).
[2] P. Baldi, P. Sadowski and D. Whiteson, Nature communications 5.1, pp. 1-9 (2014).
[3] K. Mills, M. Spanner and I. Tamblyn, Phys. Rev. A 96, 042113 (2017).
[4] J. W. Richards et al., Astrophys. J. 733, 10 (2011).
[5] A. Buckley, A. Shilton and M. J. White, Comput. Phys. Commun. 183, 960 (2012).
[6] P. Graff, F. Feroz, M. P. Hobson and A. N. Lasenby, Mon. Not. Roy. Astron. Soc. 441, 1741 (2014).
[7] G. Carleo and M. Troyer, Science 355, 602 (2017).
[8] S. J. Wetzel and M. Scherzer, Phys. Rev. B 96, 184410 (2017).
[9] Y. H. He, Phys. Lett. B 774, 564 (2017).
[10] Y. Fujimoto, K. Fukushima and K. Murase, Phys. Rev. D 98, 023019 (2018).
[11] Y. Wu, P. Zhang, H. Shen and H. Zhai, Phys. Rev. A 98, 010701 (2018).
[12] Z. M. Niu, H. Z. Liang, B. H. Sun, W. H. Long and Y. F. Niu, Phys. Rev. C 99, 064307 (2019).
[13] J. Brehmer, K. Cranmer, G. Louppe and J. Pavez, Phys. Rev. Lett. 121, 111801 (2018).
[14] J. Steinheimer, L. Pang, K. Zhou, V. Koch, J. Randrup and H. Stoecker, JHEP 1912, 122 (2019).
[15] A. J. Larkoski, I. Moult and B. Nachman, Phys. Rept. 841, 1 (2020).
[16] G. Cybenko, Mathematics of control, signals and systems 2.4, pp. 303-314 (1989).
[17] K. Hornik, Neural networks 4.2, pp. 251-257 (1991).
[18] F. Fan, J. Xiong and G. Wang, arXiv:2001.02522 [cs.LG] (2020).
[19] G. Montavon, S. Lapuschkin, A. Binder, W. Samek and K.-R. Müller, Pattern Recognition 65, pp. 211-222 (2017).
[20] M. T. Ribeiro, S. Singh, C. Guestrin, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 11351144 (2016).
[21] N. Kamiyama, Interdisciplinary information sciences 20.1, pp. 51-70 (2014).
[22] D. Hendrycks, K. Gimpel, arXiv:1606.08415v4 [cs.LG] (2020).
[23] B. Jonsson, S.T. Eng, IEEE journal of quantum electronics 26.11, pp. 2025-2035 (1990).
[24] 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. Akyhan, S. Chilamkurthy, B. Steiner, F. Lu, J. Bai and S. Chintala, in: Advances in Neural Information Processing Systems 32, pp. 8024-8035 (eds. H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox and R. Garnett) (2019).
[25] K. He, X. Zhang, S. Ren and J. Sun, IEEE Proceedings, pp. 1026-1034 (2015).
[26] T. Tieleman and G. Hinton, COURSERA: Neural networks for machine learning 4.2, pp. 26-31 (2012).
APPENDIX A − PROOFS

A. Proof of Eq. (6)

Theorem 1. The first-order partial derivatives of the $nm^{th}$ matrix element of the $l^{th}$ layer propagator $D^{(l,p)}_{nm}$ of order $p$ is 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

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

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

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = w_{nm}^{(l+1)} \frac{H^{p+1} d^{(l,m)} z^{(l)}_{m}}{dx^{p+1}} \sum_{q_l=1}^{H_l} \delta_{mq_l} \frac{\partial z^{(l)}_{q_l}}{\partial x_k}.$$

By inserting the recursive step from Eq. (2), this dependency can be shifted to the previous layer,

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \sum_{q_{l-1}=1}^{H_{l-1}} \frac{w_{q_lq_{l-1}}^{(l)} \partial z^{(l-1)}_{q_{l-1}}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \frac{d a^{(l-1,q_{l-1})}}{dx} \frac{H^{p+1} d^{(l-1,m)} w^{(l-1)}_{q_{l-1}k}}{dx} \sum_{q_l=1}^{H_l} \delta_{mq_l} \frac{\partial z^{(l-1)}_{q_{l-1}}}{\partial x_k}.$$

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

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \sum_{q_{l-1}=1}^{H_{l-1}} \sum_{q_{l-2}=1}^{H_{l-2}} \cdots \sum_{q_1=1}^{H_1} \sum_{q_0=1}^{d} w_{q_0q_l}^{(1)} \frac{\partial z^{(l)}_{q_{l-1}}}{\partial x_k}.$$

Rearranging those propagators and sums finally yields

$$\frac{\partial D^{(l,p)}_{nm}}{\partial x_k} = D^{(l,p+1)}_{nm} \sum_{q_l=1}^{H_l} \delta_{mq_l} \sum_{q_{l-1}=1}^{H_{l-1}} \cdots \sum_{q_1=1}^{H_1} \delta_{mq_l} w_{q_{l}k}^{(1)} \prod_{i=1}^{l-1} D^{(i,1)}_{q_i+1q_i}.$$
Lemma 1. With $\Delta_{mk_1...k_N}^{(l,p)}$, $\Pi_N^c$, and $S\left[\left(\pi_i\right)^c_{i=1}\right]$ as defined in Eqs. (7), (9) and (10), the following relation holds between sums over the partition subsets $\Pi_{N+1}^c$, $\Pi_N^{c-1}$, and $\Pi_N^c$:

$$
\sum_{(\pi_i)^c_{i=1}\in\Pi_{N+1}^c} \sum_{\sigma\in S[\left(\pi_i\right)^c_{i=1}]} \prod_{i=1}^c \Delta_{mk_1...k_N}^{(l,\pi_i)} \frac{\sigma(\sum_{j=1}^{i-1} \pi_j)}{\sigma(\sum_{j=1}^{i} \pi_j)} \cdots k(\sum_{j=1}^{i} \pi_j)
$$

$$
= \Delta_{mk_{N+1}}^{(l,1)} \sum_{(\pi_i)^c_{i=1}\in\Pi_{N+1}^c} \sum_{\sigma\in S[\left(\pi_i\right)^c_{i=1}]} \prod_{i=1}^{c-1} \Delta_{mk_1...k_N}^{(l,\pi_i)} \frac{\sigma(\sum_{j=1}^{i-1} \pi_j)}{\sigma(\sum_{j=1}^{i} \pi_j)}
$$

$$
+ \sum_{(\pi_i)^c_{i=1}\in\Pi_{N+1}^c} \sum_{\sigma\in S[\left(\pi_i\right)^c_{i=1}]} \prod_{i=1}^c \Delta_{mk_1...k_N}^{(l,\beta_i+1)} \frac{\sigma(\sum_{j=1}^{i-1} \beta_j)}{\sigma(\sum_{j=1}^{i} \beta_j)} k_{N+1} \prod_{i' = 1 \atop i' \neq i}^c \Delta_{mk_{i+1}}^{(l,\beta_i')} \frac{\sigma(\sum_{j=1}^{i'-1} \beta_j)}{\sigma(\sum_{j=1}^{i'} \beta_j)}
$$

(44)

Proof. Due to the bijectivity of permutations, each summand on the left-hand side of Eq. (44) is explicitly either linear in or independent of $\Delta_{mk_{N+1}}^{(l,1)}$, which allows us to make the ansatz

$$
\sum_{(\pi_i)^c_{i=1}\in\Pi_{N+1}^c} \sum_{\sigma\in S[\left(\pi_i\right)^c_{i=1}]} \prod_{i=1}^c \Delta_{mk_1...k_N}^{(l,\pi_i)} \frac{\sigma(\sum_{j=1}^{i-1} \pi_j)}{\sigma(\sum_{j=1}^{i} \pi_j)} = f_{mk_1...k_N}^{(l,c)} \Delta_{mk_{N+1}}^{(l,1)} + g_{mk_1...k_{N+1}}^{(l,c)}.
$$

(45)

Understand this as a linear function in $\Delta_{mk_{N+1}}^{(l,1)}$, the slope $f_{mk_1...k_N}^{(l,c)}$ can be derived as the derivative

$$
f_{mk_1...k_N}^{(l,c)} = \frac{\partial}{\partial \Delta_{mk_{N+1}}^{(l,1)}} \sum_{(\pi_i)^c_{i=1}\in\Pi_{N+1}^c} \sum_{\sigma\in S[\left(\pi_i\right)^c_{i=1}]} \prod_{i=1}^c \Delta_{mk_1...k_N}^{(l,\pi_i)} \frac{\sigma(\sum_{j=1}^{i-1} \pi_j)}{\sigma(\sum_{j=1}^{i} \pi_j)}
$$

(46)

Due to the derivation, only summands with $\pi_i = 1$ and $\sigma \left(\sum_{j=1}^{i} \pi_j\right) = N + 1$ contribute to $f_{mk_1...k_N}^{(l,c)}$. This requires the considered partition $(\pi_i)^c_{i=1} \in \Pi_{N+1}^c$ to contain a summand 1. As we have defined partitions to be ordered towards lower summands, this condition can be easily formulated as $\pi_c = 1$. Therefore, $(\pi_i)^c_{i=1}$ can be expressed by a permutation $(\pi_i)^c_{i=1} \in \Pi_{N}^{c-1}$,

$$
(\pi_i)^c_{i=1} = (\omega_i)^c_{i=1} \oplus (1),
$$

(47)

which allows us to reduce the sum over $\Pi_{N+1}^c$ to a sum over $\Pi_N^{c-1}$. Note that for $i' \neq i$ each contributing $\sigma \in S[\left(\omega_i\right)^c_{i=1} \oplus (1)]$ satisfies

$$
\sigma \left(1 + \sum_{j=1}^{i'-1} \pi_j\right) < \ldots < \sigma \left(\sum_{j=1}^{i'} \pi_j\right) < \sigma \left(\sum_{j=1}^{i} \pi_j\right) = N + 1.
$$

This implies that the index $k_{N+1}$ does not appear in any of the remaining factors. Therefore, we can eliminate $\pi_i$ from $(\omega_i)^c_{i=1} \oplus (1)$ together with the sum over $i$ and write the second sum as a sum over $S \left[\left(\omega_i\right)^c_{i=1} \oplus (1)\right]$, which finally yields
\[ f_{\ell mk_1 \ldots k_N}^{(l,c)} = \sum_{(\alpha_i)_{i=1}^{c-1} \in \Pi_{N-1}^c} \sum_{\sigma \in S[\{(\alpha_i)_{i=1}^{c-1}, \emptyset\}(1) \sigma = 1]}^c \delta_{\pi_i,1} \delta_{N+1, \sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \prod_{i' = 1}^{c} \Delta_{mk}^{(l, \pi_{i'} \sigma)} \left( \frac{1}{1 + \sum_{j=1}^{\pi_j} \pi_j} \right) \ldots \frac{1}{\sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \]

\[ = \sum_{(\alpha_i)_{i=1}^{c-1} \in \Pi_{N-1}^c} \sum_{\sigma \in S[\{(\alpha_i)_{i=1}^{c-1}, \emptyset\}(1) \sigma = 1]}^c \prod_{i = 1}^{\ell} \Delta_{mk}^{(l, \pi_i)} \left( \sum_{i = 1}^{\pi_i} \pi_i \right) \ldots \frac{1}{\sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \]

It remains to derive the axis intercept,

\[ g_{\ell, m, k_1 \ldots k_N+1}^{(l,c)} = \sum_{(\pi_i)_{i=1}^{c+1} \in \Pi_{N+1}^c} \sum_{\sigma \in S[\{(\pi_i)_{i=1}^{c+1}, \emptyset\}(1) \sigma = 1]}^c \prod_{i' = 1}^{\ell} \Delta_{mk}^{(l, \pi_{i'} \sigma)} \left( \sum_{i = 1}^{\pi_i} \pi_i \right) \Theta \left( \pi_{i'} - 1 \right) \Delta_{mk}^{(l, \pi_{i'}) \sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \ldots \frac{1}{\sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \]

Here, summands for which \( \pi_{i'} = 1 \) and \( \sigma \left( \sum_{j=1}^{\pi_j} \pi_j \right) = N + 1 \) are both fulfilled, are the only ones to not contribute. In contrast to the previous case, \( \pi_c = 1 \) is therefore not required. For each summand there is exactly one \( i' = i_{N+1}[\sigma] \) with \( \sigma \left( \sum_{j=1}^{\pi_j} \pi_j \right) = N + 1 \), which can be written as

\[ i_{N+1}[\sigma] = \sum_{j=1}^c j \delta_{\sigma-1(N+1), \sum_{j=1}^c \pi_j} \]

In order for such a term to contribute, it is necessary that \( \pi_{i_{N+1}[\sigma]} > 1 \), which implies \( \pi_{i_{N+1}[\sigma]} \neq 1 \).

Separating the corresponding factor from the product yields

\[ g_{\ell, m, k_1 \ldots k_N+1}^{(l,c)} = \sum_{(\pi_i)_{i=1}^{c+1} \in \Pi_{N+1}^c} \sum_{\sigma \in S[\{(\pi_i)_{i=1}^{c+1}, \emptyset\}(1) \sigma = 1]}^c \delta_{N+1, \sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \Theta \left( \pi_{i_{N+1}[\sigma]} - 1 \right) \]

\[ \times \Delta_{mk}^{(l, \pi_{i_{N+1}[\sigma]} \sigma)} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \ldots \frac{1}{\sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \]

\[ \times \prod_{i' = 1}^{\ell} \Delta_{mk}^{(l, \pi_{i'} \sigma)} \left( \sum_{i = 1}^{\pi_i} \pi_i \right) \ldots \frac{1}{\sigma} \left( \sum_{j=1}^{\pi_j} \pi_j \right) \]

In analogy to the previous analysis of the slope \( f_{\ell mk_1 \ldots k_N}^{(l,c)} \), we again want to express the sum over \( \Pi_{N+1}^c \) in terms of simpler partitions. As \( \pi_c = 1 \) is no longer required, we are able to approach this by a sum over \( \Pi_N^c \). instead. This is possible, since for all \( (\pi_i)_{i=1}^{c+1} \in \Pi_{N+1}^c \) and \( i = i_{N+1}[\sigma] \in \{1, \ldots, c\} \) there is exactly one \( (\beta_{i'})_{i'=1}^{c} \in \Pi_N^c \), such that

\[ (\pi_{i'})_{i'=1}^{c} = (\beta_{i'} + \delta_{i_{N+1}[\sigma]} i)_{i=1}^{c} \]
Note that this naturally covers all contributing partitions due to \( \pi_i = \beta_i + 1 > 1 \), which allows us to eliminate the factor \( \Theta(\pi_i^{N+1}[\sigma] - 1) \). Therefore, we can write

\[
\begin{align*}
(N,c)
g_{mk_1\ldots k_{N+1}}^{(l,c)} &= \sum_{(\beta_i)_i^c \in \Pi_{N_i}^c} \sum_{i=1}^c \sum_{\sigma \in \mathcal{S}[(\beta_i + \delta_{i,l})_{i=1}^c]} \delta_{N+1,\sigma}(\Sigma_{j=1}^i \beta_j + 1) \\
&\quad \times \Delta_{mk}^{(l,\beta_{i,l})} \sigma(\Sigma_{j=1}^{i-1} \beta_j) \cdots k_{\sigma}(\Sigma_{j=1}^i \beta_j) \\
&\quad \times \prod_{i' = 1}^c \Delta_{mk}^{(l,\beta_{i,l})} \sigma(\Sigma_{j=1}^{i'-1} \beta_j) \cdots k_{\sigma}(\Sigma_{j=1}^i \beta_j).
\end{align*}
\]

All contributing permutations must not map other values than \( \sum_{j=1}^i \beta_j + 1 \) onto \( N + 1 \), which requires

\[
\sigma : \left\{ 1, \ldots, i_j = 1, \sum_{j=1}^i \pi_j - 1, \sum_{j=1}^i \pi_j + 1, \ldots, N + 1 \right\} \rightarrow \{ 1, \ldots, N \},
\]

such that we can reduce the sum over \( \mathcal{S}[(\beta_i')_{i'=1}^c] \) to a sum over \( \mathcal{S}[\beta_i']_{i'=1}^{c-1} \), which also no longer depends on the summation index \( i \). Finally, eliminating the factor \( \delta_{N+1,\sigma}(\Sigma_{j=1}^i \beta_j + 1) \) yields the desired expression

\[
\begin{align*}
(N,c)
g_{mk_1\ldots k_{N+1}}^{(l,c)} &= \sum_{(\beta_i)_i^c \in \Pi_{N_i}^c} \sum_{i=1}^c \sum_{\sigma \in \mathcal{S}[(\beta_i')_{i'=1}^{c-1}]} \Delta_{mk}^{(l,\beta_{i,l})} \sigma(\Sigma_{j=1}^{i-1} \beta_j) \cdots k_{\sigma}(\Sigma_{j=1}^i \beta_j) \\
&\quad \times \prod_{i' = 1}^c \Delta_{mk}^{(l,\beta_{i,l})} \sigma(\Sigma_{j=1}^{i'-1} \beta_j) \cdots k_{\sigma}(\Sigma_{j=1}^i \beta_j).
\end{align*}
\]

**Theorem 2 (Eq. (8)).** The \( N^{th} \) derivative of the propagator \( D_{nm}^{(l,p)} \) is given by

\[
\frac{\partial^N D_{nm}^{(l,p)}}{\partial x_{k_1} \ldots \partial x_{k_N}} = \sum_{c=1}^N (\pi_i')_{i=1}^c \in \Pi_{N_i}^c \sum_{\sigma \in \mathcal{S}[(\pi_i')_{i=1}^c]} \prod_{i=1}^c \Delta_{mk}^{(l,\pi_i)} \sigma(\Sigma_{j=1}^{i-1} \pi_j) \cdots k_{\sigma}(\Sigma_{j=1}^i \pi_j).
\]

**Proof.** We prove Eq. (8) by a complete induction. Therefore, we quickly convince ourselves of its validity in the base case \( N = 1 \),

\[
\frac{\partial D_{nm}^{(l,p)}}{\partial x_{k_1}} = D_{nm}^{(l,p+1)} \sum_{(\pi_1)_1^c \in \Pi_{N_1}^c} \sum_{\sigma \in \{[id]\}} \Delta_{mk}^{(l,\pi_1)} \sigma(\Sigma_{j=1}^{1-1} \pi_j) \cdots k_{\sigma}(\Sigma_{j=1}^1 \pi_j) = D_{nm}^{(l,p+1)} \Delta_{mk_1}^{(l,1)}.
\]

This, indeed, corresponds to Eq. (6), which is also the defining equation for \( \Delta_{mk_1}^{(l,1)} \). Subsequently, the inductive step involves evaluating the derivative

\[
\frac{\partial^{N+1} D_{nm}^{(l,p)}}{\partial x_{k_1} \ldots \partial x_{k_{N+1}}} = \frac{\partial}{\partial x_{k_{N+1}}} \sum_{c=1}^N D_{nm}^{(l,p+c)} \sum_{(\pi_i)_i^{c-1} \in \Pi_{N_i}^{c-1}} \sum_{\sigma \in \mathcal{S}[(\pi_i)_i^{c-1}]} \prod_{i=1}^c \Delta_{mk}^{(l,\pi_i)} \sigma(\Sigma_{j=1}^{i-1} \pi_j) \cdots k_{\sigma}(\Sigma_{j=1}^i \pi_j).
\]
By applying the product rule, we either encounter propagator derivatives or derivatives of tensor elements, which we split into two distinct sums,

\[
\frac{\partial^{N+1} D^{l,p}_{nm}}{\partial x_{k1} \ldots \partial x_{kN+1}} = \sum_{c=1}^{N} D^{l,p+c+1}_{nm} \Delta^{(l,1)}_{mk_{N+1}} \sum_{(\pi_i)_{i=1}^{c}} \Pi_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk_{c}} \sigma(1 + \Sigma_{j=1}^{i-1} \pi_j) \ldots \sigma(\Sigma_{j=1}^{i} \pi_j)
\]

\[
+ \sum_{c=1}^{N} D^{l,c}_{nm} \sum_{(\pi_i)_{i=1}^{c}} \Pi_{i=1}^{c} \Delta^{(l,\pi_i+1)}_{mk_{c}} \sigma(1 + \Sigma_{j=1}^{i-1} \pi_j) \ldots \sigma(\Sigma_{j=1}^{i} \pi_j) k_{N+1}
\]

\[
\times \prod_{i' = 1, \quad i' \neq i}^{c} \Delta^{(l,\pi_{i'})}_{mk} \sigma(1 + \Sigma_{j=1}^{i' - 1} \pi_j) \ldots \sigma(\Sigma_{j=1}^{i'} \pi_j),
\]

Up to the \(N\)th summand in the first sum and the first summand in the second sum, all other summands can be combined to one sum from \(c = 2\) to \(c = N\),

\[
\frac{\partial^{N+1} D^{l,p}_{nm}}{\partial x_{k1} \ldots \partial x_{kN+1}} = \sum_{c=1}^{N} D^{l,p+c+1}_{nm} \Delta^{(l,1)}_{mk_{N+1}} \sum_{(\pi_i)_{i=1}^{c}} \Pi_{i=1}^{c} \Delta^{(l,\pi_i)}_{mk_{c}} \sigma(1 + \Sigma_{j=1}^{i-1} \pi_j) \ldots \sigma(\Sigma_{j=1}^{i} \pi_j)
\]

\[
+ \sum_{c=2}^{N} D^{l,p+c}_{nm} \left[ \sum_{(\alpha_i)_{i=1}^{c-1}} \Pi_{i=1}^{c-1} \Delta^{(l,\alpha_i)}_{mk_{c-1}} \sigma(1 + \Sigma_{j=1}^{i-1} \alpha_j) \ldots \sigma(\Sigma_{j=1}^{i} \alpha_j)
\]

\[
+ \sum_{(\beta_i)_{i=1}^{c-1}} \Pi_{i=1}^{c-1} \Delta^{(l,\beta_i+1)}_{mk_{c-1}} \sigma(1 + \Sigma_{j=1}^{i-1} \beta_j) \ldots \sigma(\Sigma_{j=1}^{i} \beta_j)
\]

\[
\times \prod_{i' = 1, \quad i' \neq i}^{c-1} \Delta^{(l,\beta_{i'})}_{mk} \sigma(1 + \Sigma_{j=1}^{i' - 1} \beta_j) \ldots \sigma(\Sigma_{j=1}^{i'} \beta_j)
\]
Using Lemma 1, the remaining sum can be expressed as a sum over $\Pi_{N+1}^c$. Then, combining all terms finally proves the inductive step,

$$\frac{\partial^{N+1} D_{nm}^{(l,p)}}{\partial x_{k_1} \cdots \partial x_{k_{N+1}}} \Delta_{mk_1}^{(l,1)} \cdots \Delta_{mk_{N+1}}^{(l,1)}$$

$$= \sum_{c=2}^{N} D_{nm}^{(l,p+c)} \sum_{(\pi_i)_{i=1}^c \in \Pi_{N+1}^c} \sum_{\sigma \in S_{\{\pi_i\}_{i=1}^c}} \prod_{i=1}^c \Delta_{mk_{\sigma(i+1)+1}}^{(l,\pi_i)} \Delta_{mk_{\sigma(i)+1}}^{(l,\pi_i)_{i=1}} \cdots \Delta_{mk_{\sigma(i)+1}}^{(l,\pi_i)_{i=1}} \sigma(\sum_{j=1}^{c} \pi_j)$$

$$+ D_{nm}^{(l,p+1)} \Delta_{m l_{N+1}}^{(l,N+1)}$$

$$= \sum_{c=1}^{N+1} D_{nm}^{(l,p+c)} \sum_{(\pi_i)_{i=1}^c \in \Pi_{N+1}^c} \sum_{\sigma \in S_{\{\pi_i\}_{i=1}^c}} \prod_{i=1}^c \Delta_{mk_{\sigma(i+1)+1}}^{(l,\pi_i)} \Delta_{mk_{\sigma(i)+1}}^{(l,\pi_i)_{i=1}} \cdots \Delta_{mk_{\sigma(i)+1}}^{(l,\pi_i)_{i=1}} \sigma(\sum_{j=1}^{c} \pi_j) \cdots k_{\sum_{j=1}^{c} \pi_j}$$

C. Proof of Eq. (20)

**Theorem 3 (Eq. (20)).** The tensor elements $\Delta_{mk_1 \cdots k_N}^{(l,p)}$ can be expressed as the following weighted sum of all $N$-vertex arborescences, as defined in Eq. (19), with adjacency matrices $A \in A^N$,

$$\Delta_{mk_1 \cdots k_N}^{(l,N)} = \sum_{A \in A^N} \Theta(1 - \alpha(A)) \delta_{mk_1 \cdots k_N}^{(l,N)}(A).$$

Weighting with factors $\Theta(1 - \alpha(A))$ causes an arborescence only to contribute, as long as the layer $l$, the tensor element is considered for, overshoots the saturation threshold $\alpha(A)$, given in Eq. (17).

**Proof.** Since the base case ($N = 1$) has already been shown in Eq. (13), we directly start with the inductive step. The commutator formula

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

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

$$B_c(A) = \bigotimes_{q, b} q^{(j_b(A))_{i=1}^{\max_r(A_{cr})}}_{j_b(A)_{i=1}^{\max_r(A_{cr})}} D_{b=1}^{(j_b(A)_{i=1}^{\max_r(A_{cr})})} \left( j_b^{(c)}_{b+1} \right)$$

from the $c^{th}$ vertex of the arborescence $\delta_{mk_1 \cdots k_N}^{(l,N)}(A)$, we derive the individual commutators
\[
\left[ \frac{\partial}{\partial x_k^{N+1}}, B_c(A) \right] \Theta (l - \max_r(A_{cr}) - 1)
\]

\[
\times \Omega \left( \prod_{b=1}^{\max_r(A_{cr})} D \left( j_b^{(c)}, 1 + \sum_{j'=1}^n \delta_{b'A_{cj}} \right) \right)
\]

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 \(b^{th}\) propagator is raised by one, which also allows an additional connection to the new vertex. Let us define the set

\[
\nu_c(A) = \bigcup_{b=1}^{\max_r(A_{cr})+1} \left\{ \left( \begin{array}{ccc} A_{11} & \ldots & A_{1N} \\ \vdots & \ddots & \vdots \\ A_{c1} & \ldots & A_{cN} \\ \vdots & \ddots & \vdots \\ A_{N1} & \ldots & A_{NN} \end{array} \right) \right\},
\]

which is a subset of \(A^{N+1}\). Its \(|\nu_c(A)| = \max_r(A_{cr}) + 1\) elements correspond to adjacency matrices of \(N\)-vertex arborescences, that have been extended by an \((N+1)\)th vertex, which is connected to the \(c^{th}\) vertex. For the element with \(b = \max_r(A_{cr}) + 1\), this corresponds to establishing a connection via an additional propagator, that is consequently of second order. Else, we have \(b \in \{1, \ldots, \max_r(A_{cr})\}\), which corresponds to raising the order of the \(b^{th}\) propagator in the \(c^{th}\) 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. (49) is given by

\[
\frac{\partial}{\partial x_{kN+1}}, B_c(A) = \Theta \left( l - \max_r(A_{cr}) - 1 \right) B_c \begin{pmatrix} A_{11} & \ldots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \ldots & A_{cN} & \max_r(A_{cr}) + 1 \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \ldots & A_{NN} & 0 \end{pmatrix} \times B_{N+1} \begin{pmatrix} 0 & \ldots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \ldots & 0 & \max_r(A_{cr}) + 1 \\ 0 & \ldots & 0 & 0 \end{pmatrix} + \sum_{b=1}^{\max_r(A_{cr})} B_c \begin{pmatrix} A_{11} & \ldots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \ldots & A_{cN} & b \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \ldots & A_{NN} & 0 \end{pmatrix} B_{N+1} \begin{pmatrix} 0 & \ldots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \ldots & 0 & b \\ 0 & \ldots & 0 & 0 \end{pmatrix}.
\] (51)

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 \(A'_{\beta_{N+1}(A'),N+1} = b\),

\[
\Omega_{\{q\{c\}_{b}^{(c)}\}_{a=1}^{\nu_c(A)}} q_{b}^{(\beta_{N+1}(A'))} k_{N+1} = \Omega_{\{q\{\beta_{N+1}(A')\}_{a=1}^{(\beta_{N+1}(A'))} k_{N+1}
\begin{pmatrix} A_{11} & \ldots & A_{1N} & 0 \\ \vdots & \ddots & \vdots & \vdots \\ A_{c1} & \ldots & A_{cN} & b \\ \vdots & \ddots & \vdots & \vdots \\ A_{N1} & \ldots & A_{NN} & 0 \end{pmatrix} = B_{N+1} \begin{pmatrix} 0 & \ldots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \ldots & 0 & b \\ 0 & \ldots & 0 & 0 \end{pmatrix}.
\]

Each element of \(\nu_c(A)\) is represented in this sum, which implies that each possible connection from the \(c\)th vertex to the \((N + 1)\)th vertex is established. Note that the first summand, that introduces a new propagator, is the only term that may alter the saturation threshold of the arborescence, namely in the case that \(\max_r(A_{cr}) \geq \alpha(A)\). Therefore, we write
The derivative of the arborescence \( \delta_{m_{k_1 \ldots k_N}}^{(l,N)}(A) \) can be written as

\[
\frac{\partial}{\partial x_{k_{N+1}}} \delta_{m_{k_1 \ldots k_N}}^{(l,N)}(A) = \sum_{q^{(0)}_i} \delta_{m_{q_i}^{(0)}} \sum_{j_{A_0}^{(0)}} \delta_{l_{j_{A_0}^{(0)}}} \left[ \frac{\partial}{\partial x_{k_{N+1}}} \prod_{c=1}^{N} B_c(A) \right].
\] (53)

Using the commutator relation in Eq. (48), we can express \( \delta_{m_{k_1 \ldots k_N}}^{(l,N)}(A) \) in terms of the individual commutators from Eq. (51),

\[
\frac{\partial}{\partial x_{k_{N+1}}} \delta_{m_{k_1 \ldots k_N}}^{(l,N)}(A) = \sum_{q^{(0)}_i} \delta_{m_{q_i}^{(0)}} \sum_{j_{A_0}^{(0)}} \delta_{l_{j_{A_0}^{(0)}}} \left[ \frac{\partial}{\partial x_{k_{N+1}}} \prod_{c=1}^{N} B_c(A) \right].
\] (54)
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:

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

Using Eq. (54), we finally complete the inductive step,

$$\Delta^{(l,N+1)}_{mk_1\ldots k_{N+1}} = \frac{\partial}{\partial x_{k_{N+1}}} \Delta^{(l,N)}_{mk_1\ldots k_N}$$

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

$$= \sum_{A \in \mathbb{A}_{N+1}^N} \Theta(l - \alpha(A)) \delta^{(l,N+1)}_{mk_1\ldots k_{N+1}}(A).$$