Algebraically-informed deep networks for associative evolution algebras

Desamparados Fernández-Ternero*[1], Víctor Manuel Gómez-Sousa†[1], Juan Núñez-Valdés‡[1], and Eduardo Paluzo-Hidalgo§[2]

[1]Department of Geometry and Topology. Faculty of Mathematics, University of Seville, Seville, Spain.
[2]Department of Applied Mathematics I. University of Seville, Seville, Spain.

Abstract

In this paper, we extend the application of the algebraically-informed deep networks (AIDNs) introduced in [4] in the case of evolution algebras. For the associative evolution algebra, the performance of AIDNs is tested for known theoretical results. This is a first step towards neural-network-aided classification of evolution algebras.

1 Introduction

Evolution algebras were firstly introduced by Tian in his Ph.D. Thesis [6] in 2004 and later published in a book in 2008 [7]. These types of algebras belong to the family of genetic algebras and have direct applications in non-Mendelian genetics [8]. In addition, its applications to other branches of mathematics are numerous, being connected, among others, with graph theory, stochastic processes and Markov chains, group theory and mathematical physics.

Classification problems in evolution algebras generally involve a large number of nonlinear equations. This makes classifying evolution algebras a difficult task. For instance, the classification of those evolution algebras whose evolution operator (the main operator of the algebra) is a homomorphism is still incomplete [2]. The purpose of this paper is to help in this task by searching for representations of evolution algebras through neural networks, which will give us an approximation of possible solutions to the problem that we can later prove theoretically. To verify the effectiveness of this method, we will use it for classifications already achieved theoretically, such as the classification of associative evolution algebras [1] or that of those evolution algebras whose evolution operator is a derivation [3].

Algebraically-informed deep networks were introduced in [4]. There, the authors provide a correspondence between a representation of an algebraic structure and feed-forward neural networks where each generator is associated with a neural network. To find an appropriate representation, those networks are trained using a traditional training algorithm which error function is based on the relations needed to be satisfied by the generators. In the case of associative evolution algebras, it is known that its structure matrix is diagonal for any dimension. In this paper, we adapt the AIDN implementation for associative evolution algebras and test its performance.

This paper is organized as follows: in Section 2, the basic concepts of evolution algebras, neural networks and AIDN are provided. Next, in Section 3, AIDNs are extended for associative evolution algebras and experiments are depicted. Finally, in Section 4, some conclusions and future work are described.

2 Background

This section provides a brief introduction to evolution algebras, neural networks and algebraically-informed deep networks over the real field. However, it can be easily extended to the complex field.

2.1 Evolution algebras

An algebra $E \equiv (E, +, \cdot)$ is said to be an *evolution algebra* if there exists a basis $\mathcal{B} = \{e_i\}_{i=1}^n$ of E such that $e_i \cdot e_j = 0$, for all $i \neq j$. Since $\mathcal{B}$ is a basis, the product $e_j \cdot e_j = e_j^2$ can be written as $\sum_{i\in\Lambda} a_{ij} e_i$, for some *structure constants* $a_{ij} \in \mathbb{R}$. So, the product on E is determined by the *structure matrix* $A = (a_{ij})$.

In general, evolution algebras are non-associative. However, in this paper we will work with those that are. This type of evolution algebras have been studied in depth in [1], where it is shown the following theorem

Theorem 1 *Let E be an evolution algebra. It is equivalent:*

1. *E is associative.*

*Email: desamfer@us.es
Research supported by national projects PID2020-118753GB-100 and PID2020-114474GB-I00, autonomic project ProyExcel-00827 and autonomic P.A.I. Group FQM-326.
†Email: victor.manuel.gomez.sousa@gmail.com
Research supported by national project PID2020-118753GB-100 and autonomic P.A.I. Group FQM-326.
‡Email: jnvaldes@us.es
Research supported by autonomic P.A.I. Group FQM-326.
§Email: epaluzo@us.es
Research partially supported by national project PID2019-107339GB-I00.

2. *$e_i \cdot e_j^2 = 0$, for all $i \neq j$.*

3. *$a_{ij}a_{ki} = 0$, for all $i \neq j$ and for all k.*

4. *There exists a rearrangement of the basis $\mathcal{B}$ such that the structure matrix has the form*

$$\begin{pmatrix} D_{r\times r} & 0_{r\times s} \\ M_{s\times r} & 0_{s\times s} \end{pmatrix},$$

where $r + s = n$ and

- *$D_{r\times r}$ is a diagonal matrix of order r.*
- *$M_{s\times r}$ is a $s \times r$ matrix.*
- *$0_{r\times s}$ and $0_{s\times s}$ are null matrices of dimension $r \times s$ and $s \times s$, respectively.*

An evolution algebra is said to be *non-degenerate* if $e_j^2 \neq 0$, for all j. In this case, the previous theorem turns out to be

Corollary 2 *Let E be a non-degenerate evolution algebra with evolution operator L. Then, the following assertions are equivalent*

1. *E is associative.*

2. *$e_i \cdot e_j^2 = 0$, for all $i \neq j$.*

3. *$a_{ij}a_{ki} = 0$, for all $i \neq j$ and for all k.*

4. *The structure matrix is a diagonal matrix with nonzero diagonal elements.*

For example, the evolution algebra with basis $\{e_1, e_2, e_3\}$ and product defined by

$$e_1^2 = e_1,$$
$$e_2^2 = 2e_2,$$
$$e_3^2 = 3e_3,$$

is a non-degenerate associative evolution algebra, since its structure matrix is $Diag(1, 2, 3)$.

2.2 Neural networks

A *neural network* is a function $Net : \mathbb{R}^{d_{in}} \to \mathbb{R}^{d_{out}}$ defined by a composition of *layer functions* $f_i : \mathbb{R}^{n_i} \to \mathbb{R}^{m_i}$, that is to say, $Net = f_L \circ \cdots \circ f_1$. The layer functions are of the form $f_i(x) = \alpha_i(W_i(x) + b_i)$, where W_i is a $m_i \times n_i$ matrix, b_i is a vector in $\mathbb{R}^{m_i}$ and α_i is the *activation function*, a chosen function (generally non linear) applied coordinate-wisely to an input vector.

We will denote the set of neural networks of the form Net: $\mathbb{R}^n \to \mathbb{R}^n$ as $\mathcal{N}(\mathbb{R}^n)$. This set is closed under composition of functions and hence has a natural algebraic structure $(\mathcal{N}(\mathbb{R}^n), +, \circ)$.

The set of weights of a neural network are trained using a gradient-based training algorithm induced by a loss function $\mathcal{L}$ that measures how far is the output of the neural network from the desired output. There exist plenty of different hyperparameters to be tuned in a neural network and a training algorithm. To name a few: the number of layers and nodes, the loss function, the learning rate, the number of epochs, among others.

2.3 Algebraically-informed deep networks (AIDN)

Let $S = \{s_i\}_{i=1}^n$ be a set of formal symbols (generators) and $R = \{r_i\}_{i=1}^k$ a formal set of equations satisfying these generators. The system $\langle S \mid R\rangle$ is called a *presentation*.

Presentations can encode different algebraic objects depending on the algebraic operations that we are willing to allow while solving the algebraic equations of R. For example, if we allow operations of addition and scalar multiplication by elements of a field satisfying the axioms implied in the definition of vector space, in addition to a bilinear product, then the resulting algebraic structure induced by the presentation $\langle S \mid R\rangle$ is an algebra. Depending on the properties that we allow in the product, this algebra can be, for example, associative or unitary.

We are interested in finding neural networks $\{f_i(x;\theta_i)\}_{i=1}^n$, where $\theta_i \in \mathbb{R}^{k_i}$ is the parameter vector of the network f_i, such that these neural networks correspond to the generators of S and satisfy the same relations of R. Formally, this is equivalent to finding a homomorphism from the algebraic structure $\langle S \mid R\rangle$ to $(\mathcal{N}(\mathbb{R}^n), +, \circ)$.

The AIDN algorithm finds the weights $\{\theta_i\}_{i=1}^n$ of the networks $\{f_i(x;\theta_i)\}_{i=1}^n$ by defining the loss function as follows

$$\mathcal{L}(f_1, \ldots, f_n) = \sum_{i=1}^{k} \parallel \mathcal{F}(r_i) \parallel_2^2,$$

where $\mathcal{F}(r_i)$ is the relation r_i written in terms of the networks $\{f_i(x;\theta_i)\}$ and $\|\cdot\|_2$ is the L^2 norm. This loss function is minimized using any known neural network training algorithm such as gradient descent or RMSprop [5].

3 AIDN for associative evolution algebras

In the case of associative evolution algebras, we considered a fixed canonical basis $\mathcal{B}$ in dimension n, hence we want to find a set of generators corresponding to the structure constants $a_{ij} \in \mathbb{R}$. Let us describe, firstly, the set of relations needed to be satisfied. The third statemen of Theorem 1 specifies the rest of the relations needed for the desired representation.

Finally, we desired to find those algebras that are not trivial (i.e. its structure matrix is not null), so we can add a final set of relations for the representation, guaranteeing that a column in the structure matrix is not null by minimizing $\frac{1}{\sum_{i=1}^n |a_{ij}|}$ for all $j \in [\![1, n]\!]$[1].

[1]Let us denote $\{1, \ldots, n\}$ by $[\![1, n]\!]$.

Therefore, an associative algebra representation is given by n^2 generators $\{a_{ij}\}_{i,j\in[\![1,n]\!]}$ subject to the following relations:

1. For all $i \neq j$ and for all k: $a_{ij}a_{ki} = 0$.

2. For all $j \in [\![1,n]\!]$: minimize $\frac{1}{\sum_{i=1}^{n}|a_{ij}|}$.

Let us remark that the first set of relations is composed of $(n^2 - n)\cdot n$ equations, and the second set is composed of n equations. As mentioned above, the basis of the associative evolution algebra will be fixed as the canonical basis which is composed of one-hot vectors and the definition of the product induced by the obtained generators. Therefore, the AIDN representation of the algebra will be based on finding a set of feed-forward neural networks $\{f_{ij}\}_{i,j\in[\![1,n]\!]} \subset \mathcal{N}(\mathbb{R})$ associated to each of the structure constants which satisfy the conditions stated above. The architecture of those neural networks can be tuned by the user. In our case, we decided to use very simple feed-forward neural networks with just one hidden layer $(1 \times 12 \times 1)$ and a linear activation function. In the case of higher dimensional representations, the number of layers and nodes can be increased. The neural networks were trained using the RMSprop training algorithm and the structure matrices were computed for dimensions 2, 3, and 4. Each of the generators is the result of applying neural networks to a specific real number parameter that was used during the training process as input data. In Table 1, examples of the matrices obtained are displayed with the error values of the error function which history is depicted in Figure 1. Let us remark that, as expected, the structure matrices obtained are diagonal matrices, satisfying the already known theoretical results.

4 Conclusions

In this paper, we have applied AIDNs to associative evolution algebras. These networks were trained for different dimensions. After the training process, the matrices obtained satisfied the known theoretical results for the structure matrix of associative evolution algebras, known to be diagonal. In future work, we plan to apply AIDNs towards patterns discovery for other types of evolution algebras.

Code availability The code of the experiments is available in the following GitHub repository `https://github.com/Cimagroup/AIDN-for-Evolution-Algebras`.

Dimension	Structure matrix
2	$\begin{pmatrix} 145.22 & 0 \\ 0 & -143.69 \end{pmatrix}$
3	$\begin{pmatrix} 19.94 & 0 & 0 \\ 0 & -20.81 & 0 \\ 0 & 0 & 20.68 \end{pmatrix}$
4	$\begin{pmatrix} -23 & 0 & 0 & 0 \\ 0 & -23.93 & 0 & 0 \\ 0 & 0 & -22.74 & 0 \\ 0 & 0 & 0 & -22.27 \end{pmatrix}$

Table 1: Examples of structure matrices obtained after training AIDNs are depicted together with the mean of the loss values for all relations for different dimensions. The reached loss values were of the order of 10^{-4} to 10^{-5}.

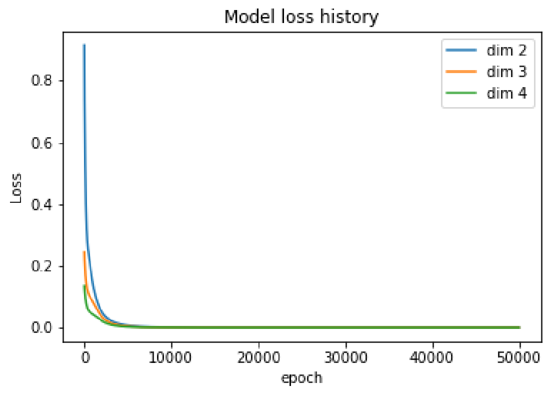

Figure 1: Neural networks were trained for 50000 epochs with a learning rate of 10^{-3}. As depicted, the evolution of the loss function during the training algorithm was similar in the three cases.

References

[1] Desamparados Fernández-Ternero, Víctor M. Gómez-Sousa, and Juan Núñez-Valdés. "A Characterization of Associative Evolution Algebras". In: *Contemp. Math.* 4.1 (2023), pp. 42–48.

[2] Desamparados Fernández-Ternero, Víctor M. Gómez-Sousa, and Juan Núñez-Valdés. "Evolution algebras whose evolution operator is a homomorphism". In: *Comput. Math. Methods.* 3.6 (2021). e1200.

[3] Desamparados Fernández-Ternero, Víctor M. Gómez-Sousa, and Juan Núñez-Valdés. "Using the Evolution Operator to Classify Evolution Algebras". In: *Math. Comput. Appl.* 26.3 (2021). Art. 57.

[4] Mustafa Hajij et al. *Algebraically-Informed Deep Networks (AIDN): A Deep Learning Approach to Represent Algebraic Structures*. 2021. arXiv: `2012.01141v3 [cs.LG]`.

[5] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2017. arXiv: `1609.04747v2 [cs.LG]`.

[6] Jianjun Paul Tian. "Evolution Algebra Theory". PhD thesis. Riverside, CA: University of California, 2004.

[7] Jianjun Paul Tian. *Evolution Algebras and Their Applications*. Berlin, Germany: Springer, 2008.

[8] Jianjun Paul Tian and Petr Vojtechovsky. "Mathematical concepts of evolution algebras in non-Mendelian genetics". In: *Quasigr. Relat. Syst.* 14.1 (2006), pp. 111–122.