Neurocomputing 559 (2023) 126785

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Improved deep neural network performance under dynamic programming mode

Julia García Cabello *

Department of Applied Mathematics, Andalusian Research Institute in Data Science and Computational Intelligence, University of Granada, Spain

ARTICLE INFO

Communicated by F.G. Guimaraes

Keywords:
Separable function
Principle of optimality
Composition of parametric functions
Universal approximators of continuous functions

ABSTRACT

For Deep Neural Networks (DNN), the standard gradient-based algorithms may not be efficient because of the raised computational expense resulting from the increase in the number of layers. This paper offers an alternative to the classic training solutions: an in-depth study to find conditions under which the underlying Artificial Neural Networks ANN minimisation problem can be addressed from a Dynamic Programming (DP) perspective. Specifically, we prove that *any* ANN with monotonic activation is separable when regarded as a parametric function. Particularly, when the ANN is viewed as a network representation of a dynamical system (as a coupled cell network), we also prove that the transmission-of-signal law is separable provided the activation function is a monotone non-decreasing function. This strategy may have a positive impact on the performance of ANNs by improving their learning accuracy, particularly for DNNs. For our purposes, ANNs are also viewed as universal approximators of continuous functions and as abstract compositions of an even number of functions. This broader representation makes it easier to analyse them from many other perspectives (universal approximation issues, inverse problem solving) leading to a general improvement in knowledge on NNs and their performance.

1. Introduction

The appeal of Artificial Neural Networks (ANN) lies in their remarkable success in performing various artificial intelligence tasks (forecasting, classifying). On a more practical level, natural language understanding, image classification, speech recognition and video processing have developed greatly thanks to the superiority of ANNs in different areas. Computational advances (resources such as GPUs) and the profusion and availability of databanks used for training deep learning methodologies are in part responsible for their success. Academically, however, the explanation for their success can be found in the fact that they are universal approximators of continuous functions. The need to give theoretical support to deep learning structures, which were born with the reputation of being black boxes due to the apparent opacity of their operation, is still valid today, [1,2]. Coupled with the growing complexity of real problems, ANN architecture and structure are becoming increasingly complex, giving way to Deep Neural Networks (DNNs) (ANNs with a large number of neurons and layers). For them, although the traditional learning algorithms were initially efficient, the rapid increase in computational requirements has demonstrated that this should be replaced (or complemented) by learning solutions which require considerably less computational effort.

Dynamic Programming (DP, [3]) is a fundamental element of Optimisation Theory and Decision Making (DM) as long as a substantial majority of the decision-making processes follow a dynamic pattern. The advantages of employing DP methods are wide ranging. In general terms, DP is suitable for linear or non-linear settings, for discrete or continuous variables and it could be applied to either deterministic or stochastic contexts. Moreover DP determines global rather than local optima. More substantively, DP is an optimisation technique which decomposes complex optimisation problems into simpler sequential ones, each of which can be solved by using Bellman's equation. For this, the original optimisation problem must satisfy the Principle of Optimality (PO), [4]. The successful implementation of DP principles in many fields suggests that the application of the ANN optimisation problem (i.e., minimising the error/cost) should lead to an improvement in their overall performance thus correcting malfunctions in high complexity cases (DNNs).

This paper offers an in-depth study to find conditions under which the underlying ANN minimisation problem can be addressed from a Dynamic Programming (DP) perspective. Specifically, we prove that

* Correspondence to: Department of Applied Mathematics, University of Granada, FCEE, Campus Cartuja s/n, 18071, Granada, Spain.
E-mail address: cabello@ugr.es.

https://doi.org/10.1016/j.neucom.2023.126785
Received 10 March 2023; Received in revised form 21 June 2023; Accepted 11 September 2023
Available online 19 September 2023

any ANN is separable when regarded as a parametric function. Particularly, when the ANN is viewed as a network representation of a dynamical system (coupled cell network, CCN, see [5]), we also prove that the transmission-of-signal law is separable. This strategy may have a positive impact on the performance of ANNs particularly for Deep Neural Networks (DNN), for which the standard gradient-based algorithms may not be efficient because of the raised computational expense resulting from the increase in the number of layers. For our purposes, ANNs are also viewed as both universal approximators of continuous functions and as an abstract composition of an even number of parametric functions.

Only a few papers address the potential usefulness of implementing the DP principles on the ANN learning algorithm. The lack of such research is surprising given the widespread applicability of DP in a wide range of fields. Actually, to this author's knowledge, only two previous works have been published, [6,7]. In the first one, [6], the author provides formulas for explicitly computing a minimum error by running the learning algorithm under a DP philosophy, i.e., by computing the minimum at each stage. Similarly, in [7] the author describes a specific procedure based on the differential dynamic programming method (DDP), which requires computing first and second partial derivatives of the function that interrelates the layers. The optimal training error is found recursively from the DP equation. Importantly, the author suggests that the competitiveness of the DP mode increases "when the number of hidden layers becomes larger and larger" (sic).

Compared to previous surveys, we approach the problem from a different viewpoint. Our major objective was to determine conditions under which the Principle of Optimality could be applied to *the ANN context* since neither [6] nor [7] explored such specifications. We thus review the requirements under which such a principle can be used in *any context* in order to select those that best fit our goal of representing ANNs as a composition of an even number of parametric functions: for us the best option is a form of separability and monotonicity. Given the disparity in the treatment of this concept, a review of the "separable" notion is performed, from the original additive sense [3], the multiplicative one [8], the two-variable notion of [9–11], by finally adopting the extension to the general multivariate case given in [12,13]. From here, first we focus on determining the conditions of the ANN functional components of the problem which enable the use of the Principle of Optimality. Secondly, we prove results that guarantee that the ANN optimisation algorithm can be run in a DP mode.

As mentioned, our proposal also includes an abstract representation of ANNs as composition of an even number of parametric functions. This broader perspective of ANNs provides an additional theoretical setting which makes it easier to analyse them from many other aspects such as universal approximation issues or inverse problem solving. In sum, our work will hopefully contribute to an improvement in the knowledge on and the performance of ANNs.

The rest of the paper is structured as follows. Section 2 is devoted to formulating the problem accompanied by a complete survey of separable functions. In Section 3, the definition of ANNs as CNNs is related to their representation as a composition of parametric functions. Sections 4 and 5 are devoted to finding pre-conditions under which the ANN learning algorithm can be run in a DP mode, focusing on ANNs as a whole (Section 4) and on their functional components (Section 5). In Section 6, the main result is finally stated and demonstrated. Section 7 concludes the paper.

2. Problem formulation

The aim of this paper is to develop a theoretical framework where the minimisation problem that underlies any ANN is treated from a DP perspective. There are disparities in how the foundations of DP are notated and treated in the literature, so let us first set them up according to our problem formulation.

2.1. DP multi-stage single-objective optimisation problems

DP is a technique for solving constrained nonlinear optimisation problems. It examines multivariate DM problems which may be decomposed into lower dimensional subproblems whose solution recursively generates the total solution.

The formulation of a DP multi-stage single-objective maximisation problem is

$$\begin{array}{ll} \underset{(u,x)}{Max} & g(x,u) \\ \text{subject to} & x(t+1) = f(x(t),u(t),t), \quad t = 0,\dots,T-1 \\ & x(0) = x_0 \text{ for } x_0 \in \mathbb{R}^n \text{ the initial condition} \\ & \mathbf{x} = (x(0),\dots,x(T)) \quad \text{the state variables} \\ & \mathbf{u} = (u(0),\dots,u(T-1)) \text{ the control variables.} \end{array} \tag{1}$$

The solution of problem Eq. (1) is usually written $(\mathbf{x}^*,\mathbf{u}^*) \in \mathbf{arg\ max}_{\mathbf{x},\mathbf{u}}\, g(\mathbf{x},\mathbf{u})$ with a parallel formulation for the minimisation case. When the stage dependency does not have to be emphasised, we shall used the bold type $\mathbf{x} \in \mathbb{R}^n, \mathbf{u} \in \mathbb{R}^m$. Otherwise, $x(t), u(t)$ shall be used instead. Specifically, a general DP multi-stage maximisation problem is a tuple $(g, f, \{X_t\}_{0\le t\le T}, U, T)$ where

- $g : \mathbb{R}^{n\times(T+1)} \times \mathbb{R}^{m\times T} \to \mathbb{R}$, $(\mathbf{x},\mathbf{u}) \longmapsto g(\mathbf{x},\mathbf{u})$, is the objective function.
  By the isomorphisms $\mathbb{R}^{n\times(T+1)} \cong \mathfrak{M}_{n\times(T+1)}(\mathbb{R})$ and $\mathbb{R}^{m\times T} \cong \mathfrak{M}_{m\times T}(\mathbb{R})$, we will use for $\mathbf{x},\mathbf{u}$ the vector and matrix notation interchangeably:
  $$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1(0) & \dots & x_1(T) \\ \vdots & \vdots & \vdots \\ x_n(0) & \dots & x_n(T) \end{pmatrix} = (x(0),\dots,x(T)),$$
  where each $x(t) \in X_t \subseteq \mathbb{R}^n, t = 0,\dots,T$ is $x(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{pmatrix}$
  $$\mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix} = \begin{pmatrix} u_1(0) & \dots & u_1(T-1) \\ \vdots & \vdots & \vdots \\ u_m(0) & \dots & u_m(T-1) \end{pmatrix} = (u(0),\dots,u(T-1)),$$
  where each $u(t) \in U \subseteq \mathbb{R}^m, t = 0,\dots,T-1$ is $u(t) = \begin{pmatrix} u_1(t) \\ \vdots \\ u_m(t) \end{pmatrix}$.
- $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{N} \to \mathbb{R}^n$ is the transition function (also "dynamics").
- $X_{t+1} = \{x(t+1) = f(x(t),u(t),t) | \mathbf{x} \in X_t, \mathbf{u} \in U, t \in \mathbb{N}\} \subseteq \mathbb{R}^n$ contains the feasible state variables (i.e., each $x(t) \in X_t$).
- $U \subseteq \mathbb{R}^m$ contains the control variables, (i.e., each $u(t) \in U$).
- $T \in \mathbb{N}$ is the time horizon.

When explicit vector notation is needed for $x(t), u(t)$, the array notation $x(t) = (x_1(t),\dots,x_n(t))$ shall be used instead of column notation $x(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{pmatrix}$ without distinction between the vector and its transpose in order to avoid confusion between t transpose and t stage. Then, we shall abuse notation when writing $g(\mathbf{x},\mathbf{u}) = g(x(0),\dots,x(T),u(0),\dots,u(T-1))$ for $(\mathbf{x},\mathbf{u}) = (x(0),\dots,x(T),u(0),\dots,u(T-1)) \in X_0 \times \cdots \times X_T \times \overbrace{U \times \cdots \times U}^{T}$. Formally, $D(t,x) = \{u(t) \in U | f(x(t),u(t),t) \in X_t\} \subseteq \mathbb{R}^m$ are those sets which contains the feasible control variables, $D(t,x) \subseteq U \subseteq \mathbb{R}^m$. Any function $D : \mathbb{N} \times \mathbb{R}^n \to \mathbb{D}$ (where $\mathbb{D} \subseteq \mathbb{R}^m$ is called the decision space) that maps any pair stage-state $(t,x) \mapsto D(t,x)$ the set $D(t,x)$ of feasible control variables at stage t and state x is called a decision (or "strategy or policy"). Thus, the set $D(t,x)$ of feasible control variables at stage t and state x is also referred to as the set of feasible decisions at stage t and state x. D^* is an optimal decision for the problem Eq. (1) if $u^* = (D^*(0,x_0),\dots,D^*(T-1,x(T-1)))$ so is.

Although $\{D(t,x)\}_{\substack{0\le t\le T \\ x\in X_t}}$ are supposed to be non-empty, we shall write $u \in U$ for simplicity.

2.2. Principle of Optimality (PO)

In Bellman's words [3] "in each process, the functional equation governing the process was obtained by an application of the following *intuitive principle*: an optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision". This is known as Principle of Optimality (hereinafter PO):

Theorem 2.1 (*Principle of Optimality, [9,10]*). *If*

$x^* = (x(0), \dots, x(T))$
$u^* = (u(0), \dots, u(T-1))$ *are a solution of Eq. (1) for* $(0, x(0) = x_0) \Rightarrow$

$(x(t), \dots, x(T))$
$(u(t), \dots, u(T-1))$ *are a solution of Eq. (1) for any intermediate stage* t, $0 \le t < T$.

In standard DP theory, the objective function $g(x,u)$ is assumed to be additively separable into subfunctions which depend on a single time-stage $(c_t(x(t),u(t)), t = 0, \dots, T-1)$ which are referred as return functions:

$$g(\mathbf{x},\mathbf{u}) = \sum_{t=0}^{T-1} c_t(x(t),u(t)) + c_T(x(T)).$$

Under this assumption (the additively separable case) Bellman's equation provides a recursive solution to Eq. (1), as follows: a function defined as the optimal of problem Eq. (1) $F(x,t) = g_t(x^*,u^*)$, satisfies the following equation (the Bellman's equation),

$$\begin{aligned} F(x,t) &= \sup_{D(t,x)} \{c_t(x,u) + F(f(x,u,t),t+1)\}, \forall x \in X_t, \\ &\quad \forall t \in \{0,..,T-1\}, \\ F(x,T) &= c_T(x), \forall x \in X_T. \end{aligned}$$

$F(x,t)$ may be reached backward recursively by maximising on u. Additionally, given a solution $F(x,t)$, a optimal policy for the problem Eq. (1) can be derived as

$$D^*(t,x) \in \underset{D(t,x)}{argmax}\{c_t(x,u) + F(f(x,u,t),t+1)\}.$$

According to the line of reasoning, when PO is satisfied, a solution $F(x,t)$ of the problem Eq. (1) may be reached recursively backwards in time. Since it enables recursively to recover an optimal solution from the solutions of the subproblems in earlier stages, PO is considered the essence of DP.

Remark 2.2. The assumption "additively separable" for the objective function is a sufficient condition for PO.

In order to determine conditions that allows us to enable PO *for the ANN learning algorithm*, we thus review the conditions under which such principle can be used *in any context*. It is worth mentioning that [8] contains a list of authors and findings related to setting sufficient and necessary-and-sufficient conditions for PO use in any context. For our goal of representing (in Section 3) ANN as composition of parametric functions, the best option is a particular form of separability.

2.3. Separable functions

In classic DP, the problem setup is conducted by converting the original optimisation problem into a functional equation where the objective function is assumed to be a sum/integral (discrete/continuous case) of functions. Although the term "separable" is not yet present in Bellman's work [3], the objective function is assumed to be additively separable in the usual sense: g is said to be additively separable if it can de decomposed as $g(\mathbf{x},\mathbf{u}) = \sum_{t=0}^{T-1} c_t(x(t),u(t)) + c_T(x(T))$ for some single time-state functions $c_t(x(t),u(t)), 0 \le t \le T-1$. Later on, separability is considered in the multiplicative sense: $g(\mathbf{x},\mathbf{u}) = \Pi_{t=0}^{T-1} c_t(x(t),u(t)) \cdot c_T(x(T))$ for some single time-state functions $c_t(x(t),u(t)), 0 \le t \le T-1$, giving rise to the idea of factorable functions and, even more, a procedure for separating into single-variable components any "factorable" function by introducing auxiliary variables and additional constraints, [8].

There are other kind of separable functions for which PO holds. In [8], separability is equivalent to satisfy the following two properties:

1. the final t stages of a process of T stages relies only on state x_{T-t} and the final ones t $u(T-t+1), u(T-t+2), \dots, u(T)$ for all t, and
2. Markovian property[1]: in stage $t+1$, after decision $u(t+1)$, the resulting state $x(t+1)$ depends only on $x(t)$ and $u(t+1)$ and in no case depends upon previous ones $x(0), x(1), \dots, x(t-1)$.

Both 1. and 2. are considered in [8] as sufficient conditions to PO "to be invoked and lead to recurrence relations and hence the valid application of DP" (sic). It is a constant in the literature that the separability (in any form) of the objective function is a sufficient condition for the use of PO.

In [9], a notion of separability based on *composition* of functions is considered. Let X and Y be two nonempty sets and let $Y(x)$ be a nonempty subset of Y which depends on x. Hence $Y(x) \in P(Y)$ where $P(Y)$ denotes the power set of Y (i.e., the set of all nonempty subsets of Y). From the equivalence between $P(Y)$ and the set of characteristic functions, the subset $Y(x)$ can be also viewed as a mapping $Y(x) \to \{0,1\}$ defined as usual:

$$\chi_{Y(x)}(y) = 1 \text{ if } y \in Y(x) \quad \text{and} \quad \chi_{Y(x)}(y) = 0 \text{ if } y \notin Y(x),$$

from which $Im\, Y(x)$ denotes the image of the corresponding characteristic function. Let

$$Gr(Y) = \{(x,y) \mid x \in X, y \in Y(x)\} = X \times Im\, Y(x) \subset X \times Y$$

be the graph of the mapping $Y(x)$. Note that this is the formal development for introducing abstract constrains in the optimisation problem.

Next Theorem 2.3 states that separability and monotonicity are sufficient conditions for PO to be satisfied, as in [10,11]:

Theorem 2.3 (*Maximax Theorem [9]*). *Let* $h : G_r(Y) \to \mathbb{R}$ *any function and* $g : X \times \mathbb{R} \to \mathbb{R}$ *be a function such that* $g(x,\cdot) : \mathbb{R} \to \mathbb{R}$ *is nondecreasing for each* $x \in X$. *If* $\mathrm{Max}_{x \in X}\ g(x, \mathrm{Max}_{y \in Y(x)}\ h(x,y))$ *exists, then* $\mathrm{Max}_{(x,y) \in \mathrm{Gr}(Y)} g(x,h(x,y))$ *exists and both are equal:*

$$\max_{(x,y) \in \mathrm{Gr}(Y)} g(x,h(x,y)) = \max_{x \in X}\ g(x, \max_{y \in Y(x)}\ h(x,y))$$

Assumptions of Theorem 2.3 are equivalent to considering an objective function that can be expressed as a composition of the form $g(x,h(x,y))$ where $g : X \times Im\ h \to \mathbb{R}$ is nondecreasing in its third argument and for each $x \in X$, $h(x,\cdot)$ is an element in the dual space $h \in Y^*$, i.e., $h(x,\cdot) : Y \to \mathbb{R}$.

Seen in this way, separability can be generalised to the n-multi variate case (Definition 2.5). Before giving the general case, the following particular cases will help in understanding it:

[1] This property is called Markovian (or "memoryless") since it stresses that the only information available to use about past states is contained in $x(t)$ when we are about to make decision $u(t+1)$.

Remark 2.4 (*Separability at Low Stage-Dimensions*).

- Case $T = 1$: $g : X_0 \times X_T \times U \to \mathbb{R}$ is said to be separable if there exist functions

$$\begin{array}{l} g_0 : X_0 \times U \times \ Im\, g_T \to \mathbb{R} \quad \text{nondecreasing in its third argument} \\ g_T : X_T \to \mathbb{R} \qquad \text{such that } g \text{ can be expressed as} \end{array}$$

$$\begin{aligned} g(\mathbf{x}, \mathbf{u}) &= g(x(0), x(T), u(0)) = \\ &= g_0(x(0), u(0), g_T(x(T))) \end{aligned}$$

- Case $T = 2$: $g : X_0 \times X_1 \times X_T \times U \times U \to \mathbb{R}$ is said to be separable if there exist functions

$$\left.\begin{array}{l} g_0 : X_0 \times U \times \ Im\, g_1 \to \mathbb{R} \\ g_1 : X_1 \times U \times \ Im\, g_T \to \mathbb{R} \end{array}\right\} \text{nondecreasing in its third argument}$$

$$\begin{array}{l} g_T : X_T \to \mathbb{R} \qquad \text{such that } g \text{ can be expressed as} \end{array}$$

$$\begin{aligned} g(\mathbf{x}, \mathbf{u}) &= g(x(0), x(1), x(T), u(0), u(1)) = \\ &= g_0(x(0), u(0), g_1(x(1), u(1), g_T(x(T)))) \end{aligned}$$

The following is the definition of separability that we adopt:

Definition 2.5 (*[12,13]*). Let $g : X_0 \times \cdots \times X_T \times U \times \overbrace{\quad \cdots \quad}^{T} \times U \to \mathbb{R}$ be the objective function of a multistage optimisation problem Eq. (1). g is said to be separable if there exists a sequence of real-valued functions $\{g_t : X_t \times U \times Im\, g_{t+1} \to \mathbb{R}\}_{0 \le t \le T-1}$ and $g_T : X_T \to \mathbb{R}$ such that

- g can be expressed (ordered backwards or forwards in time) as the composition of $\{g_t\}_{t=0}^{T}$, $g(\mathbf{x}, \mathbf{u}) = g_0\left(x(0), u(0), g_1\left(x(1), u(1), \ldots, g_T(x(T))\ldots\right)\right)$.
- Each g_t is non-decreasing in its third argument: for $z, w \in Im g_{t+1}$ are such that $z \ge w \Rightarrow g_t(x, u, z) \ge g_t(x, u, w)$.

3. Neural networks as composition of an even number of parametric functions

There are several portraits of ANNs. Here, we shall focus on their architecture. A complete review of other aspects (stochastic gradient, back propagation, examples) can be found in [1].

Due to their learning abilities, ANNs are often described as learning systems. Then, a multi-layer neural network is a mathematical learning system with a network-structure where the basic functioning units (neurons) are organised into layers such that neurons in an i-layer receive signal from neurons in a previous $i-1$-layer and send the corresponding output to neurons in an $i+1$-layer. Inner layers (that is, those that are neither the input nor the output layer) are called hidden layers. Thus, a multilayered $n-L-m$ ANN consists of n inputs, m outputs and L *hidden* layers. Note that an $n-L-m$ ANN indeed has $L+2$ layers, i.e., L hidden layers denoted as $L_i, i = 2, \ldots, L+1$ plus the input and output layers, written as L_1 and L_{L+2} respectively. Alternatively, an L_i-layer is hidden if $i \notin 1, L+2$.

The main purpose of this section is to configure a simple approach for neural networks which makes it easier to analyse them from any standpoint, particularly from a DP perspective. For this, we propose an initial definition based on coupled cell networks (CCNs). This description will soon evolve to a definition based on composition of an even number of parametric functions.

For this, recall that a CCN is an abstract arrangement of cells and links. According to the theory formalised by Stewart et al. [14–16] a CCN is a network representation of a dynamical system (either of difference or differential equations) such that the equations are coherent with a network structure: nodes represent the dynamical system laws and edges appear for interactions between nodes. The main advantage of visualising a dynamical system by using a CCN is that it allows to reach theoretical conclusions on joint dynamics based on the network structure. Additionally, from this standpoint, the close relationship between networks and (set of) functions is becoming evident. Hence, neural networks are initially defined as follows:

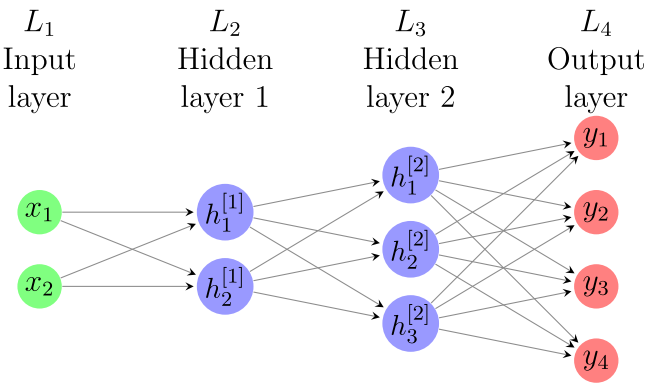


Fig. 1. Multilayered $2-2-4$ ANN.

Definition 3.1. A multilayered $n-L-m$ ANN is a coupled cell network CCN consisting of n inputs, L hidden layers and m outputs.

This definition will be further revised (Definition 3.6) in order to complete it with the dynamic system which describes de transmission of signals.

Particularly, a multilayered $2-2-4$ ANN is a CCN consisting of 2 inputs, 2 hidden layers and 4 outputs according to Fig. 1:

For our objectives of representing ANNs as a composition of parametric functions, let us first examine affine functions. A mapping is said to be affine if it is the composition of a linear map and a translation. If we denote $f : \mathbb{R}^n \to \mathbb{R}^m$ a function whose domain is a subset of $\mathbb{R}^n$ and whose range is a subset of $\mathbb{R}^m$, f is affine if it is $f(x) = Wx + b$ for matrices $W \in \mathfrak{M}_{m\times n}(\mathbb{R})$ and $b \in \mathfrak{M}_{m\times 1}(\mathbb{R})$. As mentioned in Section 2, we identify x and x^t with no distinction between the vector and its transpose.

Now, ANNs can be described as a composition of an even number of parametric functions. Let us anticipate a particular case.

Proposition 3.2 (*Multilayered $2-2-4$ ANN*). *A multilayered $2-2-4$ ANN is a parametric function $F : \mathbb{R}^2 \to \mathbb{R}^4$ of the form*

$$F = \varphi \circ f_3 \circ \varphi \circ f_2 \circ \varphi \circ f_1$$

for $f_i, i = 1, 2, 3$ affine maps $f_i(x) = W^{[i+1]}x + b^{[i+1]}, x = (x_1, x_2)$ and φ an (univariate) non-linear activation function.

Proof. Note that a multilayered $2-2-4$ ANN can be represented by the composition of the following functions after identifying each layer with a subset of the corresponding euclidean space, $L_1 \subseteq \mathbb{R}^2$, $L_2 \subseteq \mathbb{R}^2$, $L_3 \subseteq \mathbb{R}^3$, $L_4 \subseteq \mathbb{R}^4$:

$$\mathbb{R}^2 \xrightarrow{f_1} \mathbb{R}^2 \xrightarrow{\varphi} \mathbb{R}^2 \xrightarrow{f_2} \mathbb{R}^3 \xrightarrow{\varphi} \mathbb{R}^3 \xrightarrow{f_3} \mathbb{R}^4 \xrightarrow{\varphi} \mathbb{R}^4$$

where each f_i is an affine function, for i a layer counter with the usual notation of superscripts for both matrices of weights, and for biases which refer to the corresponding layer:

$$\begin{array}{llll} f_1 : \mathbb{R}^2 \to \mathbb{R}^2, & f_1(x) = & W^{[2]}x + b^{[2]}, & W^{[2]} \in \mathbb{R}^{2,2}, b^{[2]} \in \mathbb{R}^{2,1} \\ f_2 : \mathbb{R}^2 \to \mathbb{R}^3, & f_2(x) = & W^{[3]}x + b^{[3]}, & W^{[3]} \in \mathbb{R}^{3,2}, b^{[2]} \in \mathbb{R}^{3,1} \\ f_3 : \mathbb{R}^3 \to \mathbb{R}^4, & f_3(x) = & W^{[4]}x + b^{[4]}, & W^{[4]} \in \mathbb{R}^{4,3}, b^{[4]} \in \mathbb{R}^{4,1} \end{array}$$

That completes the proof.

Thus, a general multilayered $n - L - m$ ANN is a composition of parametric functions as follows:

Proposition 3.3 (*Multilayered $n-L-m$ ANN*). *A multilayered $n-L-m$ ANN (n inputs, L hidden layers and m outputs) is a parametric function (with parameters $W^{[\cdot]}, b^{[\cdot]}$) $F : \mathbb{R}^n \to \mathbb{R}^m$ composition of an even number of functions, of the form*

$$F = \varphi \circ f_{L+1} \circ \varphi \circ f_L \circ \ldots \circ \varphi \circ f_1$$

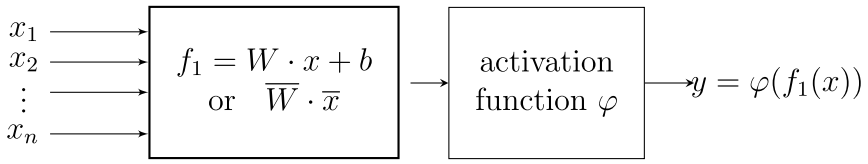


Fig. 2. Perceptron $n-1-1$ ANN.

where $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}, i = 1, \dots, L+1$ *are affine maps* $f_i(x^i) = W^{[i+1]}x + b^{[i+1]}$, $W^{i+1} \in \mathfrak{M}_{r_i \times k_i}, x^i \in \mathbb{R}^{k_i}, b^{i+1} \in \mathfrak{M}_{r_i \times 1}$, *and* φ *is a non-linear function.*

Proof. First note that, for simplicity, φ denotes both the multivariate and the univariate case. The result can be proven by mathematical induction on the number of hidden layers L. For the case of the $n-1-1$ ANN (perceptron), the single affine component transmits the input $x = (x_1, \dots, x_n)$ to the single neuron by using n weights denoted by $w_j, j = 1, \dots, n$ (that are entries of a matrix $W = (w_{1j}) = (w_{11}, \dots, w_{1n}) \in \mathbb{R}^{1,n}$ denoted $(w_1, \dots, w_n)$ for simplicity) as

$$F(x) = (w_1, \dots, w_n) \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + b = \sum_{i=1}^{n} w_i x_i + b = W \cdot x + b \tag{2}$$

or alternatively, $F(x) = \underbrace{(b, w_1, \dots, w_n)}_{\overline{W}} \cdot \underbrace{\begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{pmatrix}}_{\overline{x}} = \sum_{i=0}^{n} w_i x_i = \overline{W} \cdot \overline{x}$,

through expanded matrices $\overline{W}, \overline{x}$. Then, the perceptron is the composition of functions $F = \varphi \circ f_1$ as shown in Fig. 2:

Let us suppose that the property is satisfied for $n-L-m$ ANN. Then, the property holds for $n-(L+1)-m$ by composing the $n-L-m$ ANN with the perceptron.

Corollary 3.4 (*Multilayered* $n-L-1$ *ANN*). *A multilayered* $n-L-1$ *ANN (*n *inputs,* L *hidden layers and* 1 *output) is a parametric function (with parameters* $W^{[\cdot]}, b^{[\cdot]}$*)* $F : \mathbb{R}^n \to \mathbb{R}$ *composition of a even number of function, the form*

$$F = \varphi \circ f_{L+1} \circ \varphi \circ f_L \circ \dots \circ \varphi \circ f_1$$

where $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}, i = 1, \dots, L$ *are affine maps* $f_i(x) = W^{[i+1]}x + b^{[i+1]}$, $i = 1, \dots, L$, $f_{L+1} : \mathbb{R}^{r_{L+1}} \to \mathbb{R}$ *and* φ *is a non-linear function.*

Remark 3.5. From Proposition 3.3, the potential applicability of the DP principles to ANNs will depend on the properties of both affine and activation functions.

Since multilayered $n-L-m$ ANN as parametric functions from $\mathbb{R}^n$ to $\mathbb{R}^m$, should be studied within the scope of DP multi-stage *multi-objective* optimisation problems [4], we shall restrict our attention to multilayered $n-L-1$ ANN, deliberately procrastinating the potential generalisation for future research. Thus, hereinafter, ANN without further specification will mean $n-L-1$ ANN.

We will end this section as we started it, by recalling that multilayered ANNs can be defined as CCNs due to the specific layer architecture by which an i-layer receives signal from neurons in an $i-1$-layer and sends the output to neurons in next layer, $i+1$. Now, we can complete Definition 3.1 by explicitly describing the dynamical system of differentiate equations which governs any ANN:

Definition 3.6 (*Definition 3.1 revisited*). A multilayered $n-L-m$ ANN is a coupled cell network CCN consisting of n inputs, L hidden layers and m outputs, under the dynamical system on the transmission of signals,

$$x^{i+1} = \varphi \circ f_i(x^i) = \varphi(W^{i+1}x^i + b^{i+1})$$

from layer i (input x^i) to layer $i+1$ (output x^{i+1}), with f_i affine and φ a non-linear function, i = 1, ..., L + 1.

Note that in this definition the layer counter coincides with the counter of transition of states of neurons from one layer to the next one. This fact means that ANNs can be considered as multistage sequential decision processes.

4. Study on $n-L-1$ ANNs to use PO

Sections 4 and 5 are devoted to finding conditions under which the $n-L-1$ ANN learning algorithm can be run in a DP mode. Section 4 is focus on the ANN as a whole while in Section 5, attention is paid to their functional components. Here, we will derive conditions from the analysis of ANNs regarded as universal approximators (Section 4.1) and as a composition of functions (Section 4.2).

4.1. $n-L-1$ *ANNs as universal approximators*

Recall here that ANNs are universal approximators of continuous functions. The mathematical origin of this idea is the Hilbert's thirteenth problem in which the question arises whether a continuous function of two variables could be decomposed into continuous functions of one variable. Thus, this insight and that of separability –whether a multivariate function could be or not decomposed into univariate functions– are quite similar. This similarity provides more evidence that supports the idea of relating the properties of the target/objective function of both ANNs and DP.

The classical Universal Approximation Theorem is as follows:

Theorem 4.1 (*UAT, [17]*). *Let* α *be a continuous function defined on a compact subset* $K \subset \mathbb{R}^n$*. Thus, for* $\epsilon > 0$*, there is a neural network* F *such that, under smooth assumptions on its activation function, can approximate* α*:*

$$\|F - \alpha\|_\infty = sup_{x \in K} |F(x) - \alpha(x)| < \epsilon.$$

The continuous function α, to which the ANN approximates, is known as the target function.

Let us also recall briefly the fundamentals of the minimisation process that underlies all ANN (see [1] for detailed information). The ANN learning process, called "gradient descent minimum algorithm (GDM)" or "Taylor expansion approach", is a first-order iterative method which computes a local minimum for a differentiable objective function by adjusting the parameters weights $W^{[\cdot]}$ and biases $b^{[\cdot]}$. This function (also called cost or loss) is the committed error in approximating the target α: $E = \|F - \alpha\|_\infty$ which represents the distance between the ANN prediction and the associated target. The following result takes advantage of the error structure:

Lemma 4.2. *The cost function is a linear combination of the ANN (viewed as in Proposition 3.3) with unitary coefficients.*

Proof. By Proposition 3.3 and considering that the cost function is of the form $E(x) = |F(x) - \alpha(x)|, x \in K \subseteq \mathbb{R}^n$ ($\Rightarrow E = \|F - \alpha\|_\infty = sup_{x \in K \subseteq \mathbb{R}^n} |F(x) - \alpha(x)|$), the property holds.

The target function α should be continuous and defined in a compact subset of $\mathbb{R}^n$. Note also that, unlike the ANN (F by Corollary 3.4), the target function is independent of the parameters weights $W^{[\cdot]}$ and biases $b^{[\cdot]}$. In any case, being F, the error function is also dependent on weights and biases.

4.2. $n-L-1$ ANNs as composition of functions

Since ANNs can be regarded as *composition* of (an even number of) functions (according to Corollary 3.4 from which a multilayered $n-L-1$ ANN is of the form $F = \varphi \circ f_{L+1} \circ \ldots \circ \varphi \circ f_1$), the main goal for this subsection is to study properties which are *preserved under composition* of functions, particularly separability.

Proving that separability is closed under composition will be accomplished in Corollary 4.5. Meanwhile we shall prove a property with weaker hypothesis.

Theorem 4.3. *The (left) composition of any non-decreasing function ϕ with a separable function g, $\phi \circ g$, is separable.*

Proof. Let $g : X_0 \times \cdots \times X_T \times U^T \to \mathbb{R}$ be a separable function. Hence, by Definition 2.5, there exist a sequence of functions $\{g_t : X_t \times U \times Im\ g_{t+1} \to \mathbb{R}\}_{0 \le t \le T-1}$ and $g_T : X_T \to \mathbb{R}$ such that

- g can be expressed as the composition of $\{g_t\}_{t=0}^{T}$,

  $$g(\mathbf{x},\mathbf{u}) = g_0\left(x(0),u(0),g_1\left(x(1),u(1),\ldots,g_T(x(T))\ldots\right)\right), \quad \text{and}$$

- each g_t is non-decreasing in its third argument: for $z, w \in Im\ g_{t+1}$ are such that $z \ge w \Rightarrow g_t(x,u,z) \ge g_t(x,u,w)$.

Thus, the following sequence of functions

$$\begin{aligned} \phi_0 &= \phi \circ g_0 \\ \phi_t &= g_t, \quad 1 \le t \le T-1 \\ \phi_T &= g_T \end{aligned}$$

proves that $\phi \circ g$ is separable. Indeed,

$$\phi \circ g(\mathbf{x},\mathbf{u}) = \phi \circ g_0\left(x(0),u(0),g_1\left(x(1),u(1),\ldots,g_T(x(T))\ldots\right)\right) \text{ and}$$

for $0 \le t \le T$, the functions ϕ_t are non-decreasing in its third argument. The property holds for $1 \le t \le T$. As for $t = 0$, we shall prove it by equivalently proving that the corresponding partial derivative with respect to its third argument is non-negative. Let us denote the arguments of both g_t and $\phi \circ g_t$ as $(x(t),u(t),g_{t+1}(\cdot)), x(t) \in X_t, u(t) \in U, g_{t+1}(\cdot) \in Im\ g_{t+1}$. Thus, by the chain rule, the partial derivative is non-negative since both factors are:

$$\begin{aligned} \frac{\partial(\phi \circ g_t)}{\partial g_{t+1}(\cdot)} &= \underbrace{\phi'(g_t(x(t),u(t),g_{t+1}(\cdot)))}_{\ge 0} \cdot \underbrace{\frac{\partial g_t}{\partial g_{t+1}(\cdot)}(x(t),u(t),g_{t+1}(\cdot))}_{\ge 0} \\ &\ge 0. \end{aligned}$$

This concludes the proof.

Theorem 4.4. *Any separable function is non-decreasing in its last argument.*

Proof. Let $g : X_0 \times \cdots \times X_T \times U^T \to \mathbb{R}$ be a separable function. Hence, by Definition 2.5, there exist a sequence of functions $\{g_t : X_t \times U \times Im\ g_{t+1} \to \mathbb{R}\}_{0 \le t \le T-1}$ and $g_T : X_T \to \mathbb{R}$ such that g can be expressed as the composition of $\{g_t\}_{t=0}^{T}$,

$$g(\mathbf{x},\mathbf{u}) = g_0\left(x(0),u(0),g_1\left(x(1),u(1),\ldots,g_T(x(T))\ldots\right)\right)$$

and each g_t is non-decreasing in its third argument.

Remember that the projections p_t are those functions which take the t-coordinate from a tuple in a fold cartesian product of sets, $p_t : X_1 \times \cdots \times X_t \times \cdots \times X_T \to X_t, \quad t = 1,\ldots,T$ defined as $p_t(x_1,\ldots,x_T) = x_t$. In order to match the subscripts, we will consider $p_t : X_0 \times X_1 \times \cdots \times X_t \times \cdots \times X_T \times U^T \to X_t, \quad p_t(x_0,\ldots,x_T) = x_t, \quad t = 0,1,\ldots,T$. Its jacobian matrix $J(p_t)$ is the constant gradient $J(p_t) = \vec{\nabla} p_t = (0,\ldots,0,\underset{(t)}{1},0,\ldots,0)$.

Thus, any separable function g can be written as the composition of projections p_t and $\{g_t : X_t \times U \times Im\ g_{t+1} \to \mathbb{R}\}_{0 \le t \le T-1}$, $g_T : X_T \to \mathbb{R}$, as follows:

$$\begin{aligned} g(\mathbf{x},\mathbf{u}) &= g_0\left(x(0),u(0),g_1\left(x(1),u(1),\ldots,g_T(x(T))\ldots\right)\right) \\ &= \underbrace{g_0 \circ (p_0 \times p_{T+1} \times g_1 \circ (p_1 \times p_T)\ldots)}_{g} \\ &\times \quad (x(0),\ldots,x(T),u(0),\ldots,u(T-1)) \end{aligned}$$

Before computing the partial derivative of g with respect to the last argument, it should be noticed that the domain of $p_0 \times p_{T+1} \times g_1$ is

$$(X_0 \times \cdots \times X_T \times U^T) \times (X_0 \times \cdots \times X_T \times U^T) \times (X_1 \times U \times Im\ g_2) \subseteq \mathbb{R}^{2((T+1)+T)+3}$$

and its range is $X_0 \times X_{T+1} \times \mathbb{R} \subseteq \mathbb{R}^3$. Hence, its jacobian matrix has dimension $3 \times (2((T+1)+T)+3)$. Now, the jacobian matrix of g is

$$J(g) = \left(\frac{\partial g_0}{\partial x(0)}, \frac{\partial g_0}{\partial u(0)}, \frac{\partial g_0}{\partial g_1(\cdot)}\right) \begin{pmatrix} \frac{\partial p_0}{\partial x(0)} \cdots & \frac{\partial p_0}{\partial u(0)} \cdots & \frac{\partial p_0}{\partial g_2(\cdot)} \\ \frac{\partial p_T}{\partial x(0)} \cdots & \frac{\partial p_T}{\partial u(0)} \cdots & \frac{\partial p_T}{\partial g_2(\cdot)} \\ \frac{\partial g_1}{\partial x(0)} \cdots & \frac{\partial g_1}{\partial u(0)} \cdots & \frac{\partial g_1}{\partial g_2(\cdot)} \end{pmatrix}.$$

Particularly, the partial derivative of g with respect to the last argument is non-negative:

$$\frac{\partial g}{\partial g_2(\cdot)} = \left(\frac{\partial g_0}{\partial x(0)}, \frac{\partial g_0}{\partial u(0)}, \frac{\partial g_0}{\partial g_1(\cdot)}\right) \begin{pmatrix} \frac{\partial p_0}{\partial g_2(\cdot)} \\ \frac{\partial p_T}{\partial g_2(\cdot)} \\ \frac{\partial g_1}{\partial g_2(\cdot)} \end{pmatrix} \ge 0,$$

as we wanted to demonstrate.

Thus, previous theorems allow us finally to prove that separability is closed under composition.

Corollary 4.5. *Composition of separable functions is also separable as long as the composition exists.*

Proof. First note that it is sufficient to prove the property for a composition of only two functions. Indeed, we assume that the property holds for a composition of two functions and let us prove it for a composition of L functions by using mathematical induction on L. By hypothesis the property holds for a composition of L functions $h_1 \circ \ldots \circ h_L$ and it have to be shown for $L+1$. For the associate law for composition of functions,

$$h_1 \circ \ldots \circ h_L \circ h_{L+1} = (h_1 \circ \ldots \circ h_L) \circ h_{L+1}$$

and $(h_1 \circ \ldots \circ h_L)$ to be separable by hypothesis, the property is satisfied for $L+1$.

Let us then prove the theorem for a composition of two functions: let h_1, h_2 two separable functions such that $h_1 \circ h_2$ exists. Both are non-decreasing in their last argument by Theorem 4.4, particularly h_1 is. Thus, by applying Theorem 4.3, the result follows.

5. Study on ANN functional components to use PO

As mentioned, by Corollary 3.4, a multilayered $n-L-1$ ANN is a parametric function $F : \mathbb{R}^n \to \mathbb{R}$ of the form $F = \varphi \circ f_{L+1} \circ \varphi \circ f_L \circ \ldots \circ \varphi \circ f_1$. Hence, F can be expressed in terms of two functional components:

- affine maps $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}, f_i(x) = W^{[i+1]}x + b^{[i+1]}, i = 1,\ldots,L$, $f_{L+1} : \mathbb{R}^{r_{L+1}} \to \mathbb{R}$ and
- activation functions $\varphi : \mathbb{R} \to \mathbb{R}$, which is applied component-wise when necessary.

Both types of functionals will be analysed independently.

5.1. Conditions on affine functions

In this subsection, the us analyse the properties of affine functions regarding separability. Our first target is to show that any additively separable function is separable. This result will then allow us to finally prove that affine maps will become separable functions (Proposition 5.2).

Proposition 5.1. *Any additively separable function is separable.*

Proof. To be consistent with the objective function of Eq. (1), we consider additively separable functions $g : \mathbb{R}^{n\times(T+1)} \times \mathbb{R}^{m\times T} \to \mathbb{R}.$ $(\mathbf{x},\mathbf{u}) \longmapsto g(\mathbf{x},\mathbf{u})$ of the form

$$g(\mathbf{x},\mathbf{u}) = \sum_{t=0}^{T-1} c_t(x(t),u(t)) + c_T(x(T)).$$

Define $g_t : X_t \times U \times Im\, g_{t+1} \to \mathbb{R}$, $g_T : X_T \to \mathbb{R}$ as in [12]: $g_t(x(t),u(t),z) = c_t(x(t),u(t)) + z$, $g_T = c_T$. Let us prove the property for $T = 1$ in order to see how it works:

$$\begin{aligned} g(x(0),x(1),u(0)) &= g(x(0),x(T),u(0)) = \\ &= c_0(x(0),u(0)) + c_T(x(T)) \\ &= g_0(x(0),u(0),g_T(x(T))), \end{aligned}$$

with g_0 is nondecreasing in its third argument since the third partial derivative is $\frac{\partial g_0}{\partial z} = 1 \geq 0$. In general,

$$\begin{aligned} &g(\mathbf{x},\mathbf{u}) = g(x(0),\dots,x(T-1),x(T),u(0),\dots,u(T-1)) = \\ &= c_0(x(0),u(0)) + \cdots + \underbrace{c_{T-1}(x(T-1),u(T-1)) + c_T(x(T))}_{g_{T-1}((x(T-1),u(T-1)),c_T(x(T)))} = \\ &= \cdots = \\ &= g_0\left(x(0),u(0),g_1\left(x(1),u(1),\dots,g_T(x(T))\dots\right)\right) \end{aligned}$$

Moreover, for each g_t $t = 1,\dots,T$ is nondecreasing in its third argument since the third partial derivative is $\frac{\partial g_t}{\partial z} = 1 \geq 0$. For functions with range $\mathbb{R}^m$, the proof is performed component-wise. This ends the proof.

This proposition allows us to show finally the desired result:

Proposition 5.2. *Any affine function is separable.*

Proof. We shall show that any affine mapping is additively separable. Hence, it will be separable by Proposition 5.1. Let $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}$ be any affine map. By definition, it is of the form $f_i(x) = W^{[i+1]}x + b^{[i+1]}, x = (x_1,\dots,x_{k_i})$ for $i = 1,\dots,l$.

Specifically, f can be decomposed into sum of univariate (affine) functions, as follows:

$$f_i\begin{pmatrix} x_1 \\ \vdots \\ x_{k_i} \end{pmatrix} = \begin{pmatrix} w_{11}^{i+1} & \cdots & w_{k_i1}^{i+1} \\ \vdots & \ddots & \vdots \\ w_{r_i1}^{i+1} & \cdots & w_{r_ik_i}^{i+1} \end{pmatrix}\begin{pmatrix} x_1 \\ \vdots \\ x_{k_i} \end{pmatrix} + \begin{pmatrix} b_1^{i+1} \\ \vdots \\ b_{r_i}^{i+1} \end{pmatrix} = \tag{3}$$

$$\begin{aligned} &= \begin{pmatrix} w_{11}^{i+1}x_1 + \cdots + w_{k_i1}^{i+1}x_{k_i} + b_1^{i+1} \\ \vdots \\ w_{r_i1}^{i+1}x_1 + \cdots + w_{r_ik_i}^{i+1}x_{k_i} + b_{r_i}^{i+1} \end{pmatrix} = \\ &= \begin{pmatrix} w_{11}^{i+1} \\ \vdots \\ w_{r_i1}^{i+1} \end{pmatrix}x_1 + \cdots + \begin{pmatrix} w_{k_i1}^{i+1} \\ \vdots \\ w_{r_ik_i}^{i+1} \end{pmatrix}x_{k_i} + \begin{pmatrix} b_1^{i+1} \\ \vdots \\ b_{r_i}^{i+1} \end{pmatrix} \\ &= g_1(x_1) + \cdots + g_{k_i}(x_{r_i}), \end{aligned}$$

where the independent column vector can be considered as part of any g_i.

Note also that, as it has been proved that each $g_{k_i}(x_{r_i})$ is affine, the proof could also have been performed by mathematical induction on k_i.

5.2. Conditions on activation functions

In this subsection, attention is paid to activations functions. The objective is to study the properties which guarantee that the learning algorithm can be run in a DP mode.

Monotonicity is the universally required property for activation functions. For descendent-gradient models, the reason for this is that monotonicity both mitigates the vanishing-gradient problem and ensures that the algorithm converges. Recently, there has been a growing demand for monotonic DNN models in response to the large part of real problems which have *dominance* as one of their features[2] (monotonicity is a mathematical translation of dominance). This results into *monotonic classification*, in which a monotonicity relationship between the set of attributes and the class labels of the samples exists, [18,19].

Monotonic DNN models are a particular case of *monotonic networks* [20] one of the most successful tools for monotonic classification. A network $F : \mathbb{R}^n \to \mathbb{R}^m$ is said to be monotonic with respect to the inputs $x_i, i = 1,\dots,n$ if $\frac{\partial F}{\partial x_j} \geq 0$.

Next results provide explicit formulas on the relationship between the partial derivative of F and that of φ and f_i. Since by Corollary 3.4, a multilayered $n - L - 1$ ANN $F : \mathbb{R}^n \to \mathbb{R}$ is of the form

$$F = \varphi \circ f_{L+1} \circ \varphi \circ f_L \circ \dots \circ \varphi \circ f_1$$

$f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}, i = 1,\dots,L$ $f_i(x) = W^{[i+1]}x + b^{[i+1]}$, $i = 1,\dots,L$, $f_{L+1} : \mathbb{R}^{r_{L+1}} \to \mathbb{R}$ and φ non-linear, we will first examine the partial derivatives of the functional pair $\varphi \circ f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}$,

$$\begin{array}{ccccc} \mathbb{R}^{k_i} & \longrightarrow & \mathbb{R}^{r_i} & \longrightarrow & \mathbb{R}^{r_i} \\ & \searrow & & \nearrow & \\ & & \varphi\circ f_i & & \end{array}$$

Proposition 5.3. *For affine functions $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}$ and non-linear activation $\varphi : \mathbb{R}^{r_i} \to \mathbb{R}^{r_i}$, the partial derivatives of the composite function $\varphi \circ f_i$ are linear combination of $\varphi\prime$ and weights $w^{[i+1]}$.*

Proof. In order to apply the chain rule to the composite function, recall that $\varphi : \mathbb{R}^{r_i} \to \mathbb{R}^{r_i}$ is a simplified notation for $\varphi\times\overset{(r_i)}{\dots}\times\varphi : \mathbb{R}^{r_i} \to \mathbb{R}^{r_i}$, the r_ith cartesian product of the univariate real valued activation function $\varphi : \mathbb{R} \to \mathbb{R}$:

$$(\varphi \times \overset{(r_i)}{\dots} \times \varphi)(y_1,\dots,y_{r_i}) = (\varphi(y_1),\dots,\varphi(y_{r_i})).$$

The Jacobian matrix $J\varphi$ is thus $J\varphi = J(\varphi\times\overset{(r_i)}{\dots}\times\varphi) = (\varphi',\dots,\varphi')$.

We will denote $f_{i,k}$ the kth functional component of the affine function $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}$,

$$f_i(x_1,\dots,x_{k_i}) = (f_{i,1}(x_1,\dots,x_{k_i}),\dots,f_{i,r_i}(x_1,\dots,x_{k_i})).$$

Since each f_i-component is

$$\begin{aligned} f_{i,1}(x_1,\dots,x_{k_i}) &= \textstyle\sum_{j=1}^{k_i} w_{1j}^{[i+1]} + b_1^{[i+1]} \\ f_{i,2}(x_1,\dots,x_{k_i}) &= \textstyle\sum_{j=1}^{k_i} w_{2j}^{[i+1]} + b_2^{[i+1]} \\ \dots & \quad \dots \\ f_{i,r_i}(x_1,\dots,x_{k_i}) &= \textstyle\sum_{j=1}^{k_i} w_{r_ij}^{[i+1]} + b_{r_i}^{[i+1]}, \end{aligned}$$

the corresponding partial derivatives are

$$\begin{aligned} &\frac{\partial f_{i,1}}{\partial x_1} = w_{11}^{[i+1]}, \quad \frac{\partial f_{i,1}}{\partial x_2} = w_{12}^{[i+1]} \dots \quad \frac{\partial f_{i,1}}{\partial x_{k_i}} = w_{1k_i}^{[i+1]} \\ &\dots \qquad\qquad \dots \qquad\qquad \dots \\ &\frac{\partial f_{i,r_i}}{\partial x_1} = w_{r_i1}^{[i+1]}, \quad \frac{\partial f_{i,r_i}}{\partial x_2} = w_{r_i2}^{[i+1]} \dots \quad \frac{\partial f_{i,r_i}}{\partial x_{k_i}} = w_{r_ik_i}^{[i+1]} \end{aligned}$$

[2] Dominance of one financial institution over others, of a gene over others ...

By applying the chain rule to the composite function $\varphi \circ f_i$, the relationship between the corresponding jacobian matrices is the following product of matrices for $(y_1, \dots, y_{r_i}) = f_i(x_1, \dots, x_{k_i})$:

$$\begin{aligned} J(\varphi \circ f_i)(x_1, \dots, x_{k_i}) &= J(\varphi \times \overset{(r_i)}{\dots} \times \varphi)(y_1, \dots, y_{r_i}) \cdot J(f_i)(x_1, \dots, x_{k_i}) = \\ &= (\varphi', \dots, \varphi')(y) \cdot \begin{pmatrix} \frac{\partial f_{i,1}}{\partial x_1}(x) & \dots & \frac{\partial f_{i,1}}{\partial x_{k_i}}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{i,r_i}}{\partial x_1}(x) & \dots & \frac{\partial f_{i,r_i}}{\partial x_{k_i}}(x) \end{pmatrix} \\ &= (\varphi', \dots, \varphi')(y) \cdot \begin{pmatrix} w_{11}^{[i+1]} & \dots & w_{1k_i}^{[i+1]} \\ \vdots & \ddots & \vdots \\ w_{r_i 1}^{[i+1]} & \dots & w_{r_i k_i}^{[i+1]} \end{pmatrix} \\ &= (\varphi'(y_1) w_{11}^{[i+1]} + \dots + \varphi'(y_{r_i}) w_{r_i 1}^{[i+1]}, \dots, \\ &\quad \varphi'(y_1) w_{1k_i}^{[i+1]} + \dots + \varphi'(y_{r_i}) w_{r_i k_i}^{[i+1]}), \end{aligned}$$

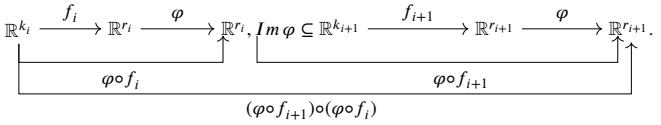
with $x = (x_1, \dots, x_{k_i})$ and $y = (y_1, \dots, y_{r_i})$ in order to shorten the notation. Hence, the partial derivatives of the composition $\varphi \circ f_i$ are:

$$\begin{aligned} \frac{\partial \varphi \circ f_i}{\partial x_1} &= \varphi'(y_1) \cdot w_{11}^{[i+1]} + \dots + \varphi'(y_{r_i}) \cdot w_{r_i 1}^{[i+1]} \\ \dots & \quad \dots \\ \frac{\partial \varphi \circ f_i}{\partial x_{k_i}} &= \varphi'(y_1) \cdot w_{1k_i}^{[i+1]} + \dots + \varphi'(y_{r_i}) \cdot w_{r_i k_i}^{[i+1]} \end{aligned} \tag{4}$$

That concludes the proof.

Proposition 5.4. *For a multilayered $n - L - 1$ ANN $F : \mathbb{R}^n \to \mathbb{R}$ is of the form $F = \varphi \circ f_{L+1} \circ \varphi \circ f_L \circ \dots \circ \varphi \circ f_1$, $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}, i = 1, \dots, L$ $f_i(x) = W^{[i+1]}x + b^{[i+1]}$, $i = 1, \dots, L$, $f_{L+1} : \mathbb{R}^{r_{L+1}} \to \mathbb{R}$ and φ non-linear, the partial derivatives of the composite function $\varphi \circ f_i$ are linear combination of $\varphi\prime$ and weights w.*

Proof. We assume that the conditions for the existence of the compositions are met, i.e., for $\varphi : \mathbb{R}^{r_i} \to \mathbb{R}^{r_i}$, $Im\, \varphi \subseteq \mathbb{R}^{k_{i+1}}, i = 1, \dots, L$:



The jacobian matrix of the composition $(\varphi \circ f_{i+1}) \circ (\varphi \circ f_i)$ is

$$\begin{aligned} J((\varphi \circ f_{i+1}) \circ (\varphi \circ f_i))(x) &= J(\varphi \circ f_{i+1})((\varphi \circ f_i)(x)) \cdot J(\varphi \circ f_i)(x) = \\ &= J(\varphi \circ f_{i+1})(z) \cdot J(\varphi \circ f_i)(x) \end{aligned}$$

for $x = (x_1, \dots, x_{k_i})$, $y = f_i(x)$ and $z = (\varphi \circ f_i)(x)$ in order to shorten the notation. By application of the formulae shown in Proposition 5.3, the jacobian is

$$\begin{aligned} &J((\varphi \circ f_{i+1}) \circ (\varphi \circ f_i))(x) = J(\varphi \circ f_{i+1})(z) \cdot J(\varphi \circ f_i)(x) = \\ &(\varphi', \dots, \varphi')(z) \cdot \begin{pmatrix} w_{11}^{[i+2]} & \dots & w_{1k_{i+1}}^{[i+2]} \\ \vdots & \ddots & \vdots \\ w_{r_{i+1}1}^{[i+2]} & \dots & w_{r_{i+1}k_{i+1}}^{[i+2]} \end{pmatrix} \\ &\cdot (\varphi', \dots, \varphi')(y) \cdot \begin{pmatrix} w_{11}^{[i+1]} & \dots & w_{1k_i}^{[i+1]} \\ \vdots & \ddots & \vdots \\ w_{r_i 1}^{[i+1]} & \dots & w_{r_i k_i}^{[i+1]} \end{pmatrix} \end{aligned}$$

This ends the proof.

The most straightforward conclusion from Propositions 5.3 and 5.4 is that a way to generate a monotonic network is to limit its weights to be non-negative and to impose that activation function to be monotonic. The purpose of explicitly stating the relationship of dependence between the partials of F and the derivative of the activation function together with the sign of the weights, is to provide material for other conditions to be demonstrated.

After the exposition in this subsection of the benefits of monotonicity, a further conclusion is that monotonicity can be presumed for activation functions:

Proposition 5.5. *Activation should be a monotonically nondecreasing function.*

6. Main result

As mentioned, our main target is to set conditions under which the ANN minimising algorithm can be run sequentially. To do this, as an application of all the results that have been shown above, we can already prove the following

Theorem 6.1 (*Any $n - L - 1$ multilayered ANN is separable*)**.** *Under the premise that the activation function is non-decreasing, any $n - L - 1$ multilayered ANN is a separable function as composition of parametric functions.*

Proof. Written as a composition of parametric functionals

$$F = \varphi \circ f_{L+1} \circ \varphi \circ f_L \circ \dots \circ \varphi \circ f_1$$

with $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}, i = 1, \dots, L$, $f_{L+1} : \mathbb{R}^{r_{L+1}} \to \mathbb{R}$ affine maps and φ the (activation) function (according to Corollary 3.4) by the associative law for composition of functions, any $n - L - 1$ ANN can be expressed as a composition of functional pairs:

$$F = (\varphi \circ f_{L+1}) \circ (\varphi \circ f_L) \circ \dots \circ (\varphi \circ f_1).$$

Indeed, each pair $(\varphi \circ f_i)$ is a left composition of a non-decreasing function φ (φ must be non-decreasing by Proposition 5.5) with a separable one f_i (according to Proposition 5.2 from which any affine map is separable). In consequence, by Theorem 4.3, each functional pair $(\varphi \circ f_i)$ is a separable function. Finally, by applying Corollary 4.5, F is separable.

As mentioned before, non-decreasing DNNs are most demanded. Hence, a straightforward consequence of the above result is the following:

Corollary 6.2. *As long as the activation function φ is non-decreasing, the F is a monotonic network.*

Let us return for a moment to the original definition of ANN in terms of coupled cell networks (Definition 3.6) where the transition law between one layer and the next one is the dynamical system

$$x^{i+1} = \varphi \circ f_i(x^i) = \varphi(W^{i+1}x^i + b^{i+1}).$$

Thus, as a consequence of Theorem 6.1, one has the following

Corollary 6.3. *In any $n - L - m$ multilayered ANN (viewed as CCN by Definition 3.6) the transition function representing the dynamics between layers, $\varphi \circ f_i$, $i = 1, \dots, L$, is separable as long as the φ is non-decreasing.*

Proof. Throughout the proof Theorem 6.1 it has been proved that any functional pair $(\varphi \circ f_i)$ composed of activation function and affine map is separable.

Table 1 allows us to visualise the potential applicability of DP mode to the ANN underlying minimisation process through the comparison between both general setups. For this, the differentiate equation $x^{i+1} = h(x^i, W^{i+1}, i)$ is a simple way of highlighting the dependence of the transition law, formerly referred to as $x^{i+1} = \varphi \circ f_i(x^i) = \varphi(W^{i+1}x^i + b^{i+1})(= h(x^i, W^{i+1}, i))$ for affine maps $f_i : \mathbb{R}^{k_i} \to \mathbb{R}^{r_i}$, $f_i(x^i) = W^{[i+1]}x + b^{[i+1]}$ and φ a non-linear function. Note that the objective

Table 1
ANN minimisation algorithm on DP mode.

DP multi-stage minimisation problem		ANN underlying minimisation problem	
$\underset{(u,x)}{Min}$	$g(x,u)$	$\underset{W^i,b^i}{Min}$	$E(x) = F(x) - \alpha(x)$
subject to	$x(t+1) = f(x(t),u(t),t)$	subject to	$x^{i+1} = h(x^i, W^{i+1}, i)$ $= \varphi \circ f_i(x^i) =$ $= \varphi(W^{i+1}x^i + b^{i+1})$
	$t = 0,\dots,T-1$		$i = 1,\dots,L$
	$x(0) = x_0$		f_i affine, φ activation
	$\mathbf{x} = (x(0),\dots,x(T))$		$x^i \in K \subseteq \mathbb{R}^{k_i}$
	$\mathbf{u} = (u(0),\dots,u(T-1))$		$W^{i+1} \in \mathfrak{M}_{r_i \times k_i}$

function has been expressed simply as $E(x) = F(x) - \alpha(x)$ (instead of $E(x) = |F(x) - \alpha(x)|$) for the relationship between the extrema of a given function and its opposite (once the minimum m of a function has been found, $-m$ is the minimum of the opposite function).

7. Conclusions

This manuscript presents an alternative to the classical training solutions for Deep Neural Networks (DNN), for which the standard gradient-based algorithms may not be efficient because of the raised computational expense of the increase in the number of layers. Specifically, an in-depth study to find conditions under which the underlying ANN minimisation problem can be addressed from a Dynamic Programming (DP) perspective is provided.

In this line, we have shown that ANNs with monotonic activation are separable when regarded as parametric functions. Additionally, for ANNs viewed as network representations of a dynamical system (CCNs), the transmission-of-signal law has also been proved to be separable as long as the activation function is monotone non-decreasing. In order to reach these results, the theoretical framework used has considered ANNs as a composition of an even number of parametric functions. Such an abstract representation makes it easier to analyse ANNs from many other perspectives (universal approximation issues, inverse problem solving) which has led to a general improvement in the knowledge in and performance of ANNs.

In this work, ANNs have also been regarded as universal approximators of continuous functions. In order to demonstrate the validity of the approach proposed, the error function has been treated in its simplest structure (see Lemma 4.2) for displaying the strong dependence between error E and target function α. This makes it extremely difficult to establish generalistic conditions under which E is separable that go beyond those obtained in Theorem 6.1.

However, for future research, requirements on α (which in the first instance, must be continuous and defined in a compact set in order to apply the UAT) should be explored in the sense that they enable the error to be a separable function. For this, α should intuitively verify the same conditions as any objective function of a DP problem: to be separable, i.e., to decompose –in some way– into certain univariate functions. This is precisely what is established for α in the representation theorems that are the basis of the UAT theorem (Theorem 4.1). In the discrete case, Kolmogorov's Theorem, Sprecher's version, and Irie–Miyake's Theorem for continuous contexts and integrable functions (L^2-integrable) are:

Theorem 7.1 (*Kolmogorov's Theorem*). *Any continuous function* $f : [0,1] \times \overset{(n)}{\dots} \times [0,1] \to \mathbb{R}, n \geq 2$, $f(x_1,\dots,x_n)$, *can be decomposed as*

$$f(x_1,\dots,x_n) = \sum_{j=1}^{2n+1} \chi_j \left(\sum_{i=1}^{n} \varphi_{ij}(x_i) \right),$$

where both χ_j, φ_{ij} *are continuous univariate functions and* φ_{ij} *are monotone.*

Theorem 7.2 (*Sprecher's Theorem*). *Any continuous function* $f : [0,1] \times \overset{(n)}{\dots} \times [0,1] \to \mathbb{R}, n \geq 2$, $f(x_1,\dots,x_n)$, *can be represented as*

$$f(x_1,\dots,x_n) = \sum_{j=1}^{2n+1} \chi \left(\sum_{i=1}^{n} \lambda^i \varphi(x_i + \epsilon(j-1)) + j - 1 \right),$$

for a real, monotonically increasing function $\varphi(x)$ *which has associated a rational number* ϵ, $0 < \epsilon < \delta$, *given such a* δ, *and with the property that* $\varphi[0,1] = [0,1]$.

Theorem 7.3 (*Irie–Miyake's Theorem*). *Let* $f(x_1,\dots,x_n) \in L^2(\mathbb{R}^n)$ *and* $\varphi(x) \in L^1(\mathbb{R})$. *Let* $\Psi(\xi)$, $F(w_1,\dots,w_n)$ *be the Fourier transforms of* $\varphi(x)$ *and* $f(x_1,\dots,x_n)$, *respectively. If* $\Psi(1) \neq 0$, *then*

$$f(x_1,\dots,x_n) = \int_{\mathbb{R}^{n+1}} \psi \left(\sum_{i=1}^{n} x_i w_i - w_0 \right) \frac{1}{(2\pi)^n \Psi(1)} \times F(w_1,\dots,w_n)\, e^{iw_0}\, d\mathbf{w}$$

where $d\mathbf{w} = dw_0 dw_1 \dots dw_n$.

Note that these representation theorems state that any continuous function defined on $[0,1] \times \overset{(n)}{\dots} \times [0,1]$ (a compact set) can be approximated by ANNs, with φ as activation function. In the case of Irie–Miyake's Theorem, w_0 is the bias and w_i are the weights.

CRediT authorship contribution statement

Julia García Cabello: Solely responsible, Writing and preparation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

Financial support from the Spanish Ministry of Universities. "Disruptive group decision making systems in fuzzy context: Applications in smart energy and people analytics" (PID2019-103880RB-I00). Main Investigator: Enrique Herrera Viedma, and Junta de Andalucía. "Excellence Groups", Spain (P12.SEJ.2463) and Junta de Andalucía, Spain (TIC186) are gratefully acknowledged. Research partially supported by the "Maria de Maeztu" Excellence Unit IMAG, reference CEX2020-001105-M, funded by MCIN/AEI/10.13039/501100011033/.

References

[1] C.F. Higham, D.J. Higham, Deep learning: An introduction for applied mathematicians, Siam Rev. 61 (4) (2019) 860–891, http://dx.doi.org/10.1137/18M1165748.
[2] C. Ma, S. Wojtowytsch, L. Wu, et al., Towards a mathematical understanding of neural network-based machine learning: what we know and what we don't, 2020, arXiv preprint arXiv:2009.10713, http://dx.doi.org/10.48550/arXiv.2009.10713.
[3] R. Bellman, Dynamic Programming, Press Princeton, New Jersey, 1957, http://dx.doi.org/10.1126/science.153.3731.34.
[4] M.L. Hussein, M.A. Abo-Sinna, Decomposition of multiobjective programming problems by hybrid fuzzy-dynamic programming, Fuzzy Sets and Systems 60 (1) (1993) 25–32, http://dx.doi.org/10.1016/0165-0114(93)90286-Q.
[5] E. Nijholt, B. Rink, J. Sanders, Center manifolds of coupled cell networks, SIAM Rev. 61 (1) (2019) 121–155, http://dx.doi.org/10.1137/18M1219977.
[6] P. Saratchandran, Dynamic programming approach to optimal weight selection in multilayer neural networks, IEEE Trans. Neural Netw. 2 (4) (1991) 465–467, http://dx.doi.org/10.1109/72.88167.
[7] M. Sun, Training multilayer feedforward neural networks using dynamic programming, in: Proceedings of 28th Southeastern Symposium on System Theory, IEEE, 1996, pp. 163–167, http://dx.doi.org/10.1109/SSST.1996.493491.

[8] L. Cooper, M.W. Cooper, Introduction To Dynamic Programming: International Series in Modern Applied Mathematics and Computer Science, Vol. 1, Elsevier, 2016.
[9] S. Iwamoto, Sequential minimaximization under dynamic programming structure, J. Math. Anal. Appl. 108 (1) (1985) 267–282, http://dx.doi.org/10.1016/0022-247X(85)90023-X.
[10] L. Mitten, Composition principles for synthesis of optimal multistage processes, Oper. Res. 12 (4) (1964) 610–619, http://dx.doi.org/10.1287/opre.12.4.610.
[11] T.L. Morin, Monotonicity and the principle of optimality, J. Math. Anal. Appl. 88 (2) (1982) 665–674, doi:0022 247X/82/080665.
[12] D. Li, Multiple objectives and non-separability in stochastic dynamic programming, Internat. J. Systems Sci. 21 (5) (1990) 933–950, http://dx.doi.org/10.1080/00207729008910422.
[13] D. Li, Y.Y. Haimes, The envelope approach for multiobjeetive optimization problems, IEEE Trans. Syst. Man Cybern. 17 (6) (1987) 1026–1038, http://dx.doi.org/10.1109/TSMC.1987.6499313.
[14] M. Golubitsky, I. Stewart, Nonlinear dynamics of networks: the groupoid formalism, Bull. Am. Math. Soc. 43 (3) (2006) 305–364, http://dx.doi.org/10.1090/S0273-0979-06-01108-6.
[15] M. Golubitsky, I. Stewart, A. Török, Patterns of synchrony in coupled cell networks with multiple arrows, SIAM J. Appl. Dyn. Syst. 4 (1) (2005) 78–100, http://dx.doi.org/10.1137/040612634.
[16] I. Stewart, M. Golubitsky, M. Pivato, Symmetry groupoids and patterns of synchrony in coupled cell networks, SIAM J. Appl. Dyn. Syst. 2 (4) (2003) 609–646, http://dx.doi.org/10.1137/S1111111103419896.
[17] M. Leshno, V.Y. Lin, A. Pinkus, S. Schocken, Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, Neural Netw. 6 (6) (1993) 861–867, http://dx.doi.org/10.1016/S0893-6080(05)80131-5.
[18] J.-R. Cano, P.A. Gutiérrez, B. Krawczyk, M. Woźniak, S. García, Monotonic classification: An overview on algorithms, performance measures and data sets, Neurocomputing 341 (2019) 168–182, http://dx.doi.org/10.48550/arXiv.1811.07155.
[19] W. Kotlowski, R. Slowinski, On nonparametric ordinal classification with monotonicity constraints, IEEE Trans. Knowl. Data Eng. 25 (11) (2012) 2576–2589, http://dx.doi.org/10.1109/TKDE.2012.204.
[20] J. Sill, Monotonic networks, in: Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10, NIPS '97, MIT Press, Cambridge, MA, USA, 1998, pp. 661–667.

Dr. Julia García Cabello Born in Andalusia (Spain). Dr. Julia García Cabello held a Ph.D. in Pure and Applied Mathematics from the University of Granada where she has been teaching since 1990. Prior to arriving at the world of Applied Mathematics, she developed a successful career in Pure Algebra (known as JG Cabello), publishing in flagship journals on Algebraic Homotopy Theory. Today, she is fully tenured professor and full researcher at the Applied Mathematics Department of the University of Granada (Spain), where she teaches undergraduate, MBA and Executive MBA Dr. Julia García Cabello courses and conducts seminars on a wide range of mathematical business-related topics. She has been also Academic Secretary of the Department of Applied Mathematics till 2023.

She is full researcher of Andalusian Research Institute in Data Science and Computational Intelligence. Her current research interests include the application of Applied Mathematics to the Resolution of Real Problems, Decision Making, Theoretical Computer Science and Operational Research. To this regard, her mathematical baggage (from the Pure Algebra to Applied Mathematics) makes Dr. Julia García Cabello's research characterised by using a wide range of mathematical tools, from stochastic processes to dynamic systems, with the purpose of solving the stated real problem. Dr. Julia García Cabello is also regular reviewer in journals of Applied Mathematics and Intelligent and Information Systems.