Control and Machine Learning

Umberto Biccari
Chair of Computational Mathematics, Universidad de Deusto, Bilbao, Spain
umberto.biccari@deusto.es
cmc.deusto.es

based on works by:
Enrique Zuazua - Friedrich-Alexander-Universität, Erlangen
Universidad de Deusto, Bilbao
Universidad Autónoma de Madrid
Rafael Orive-Illera - Universidad Autónoma de Madrid and ICMAT
Domènec Ruíz-Balet - Université Paris Dauphine
Antonio Álvarez-López - Universidad Autónoma de Madrid

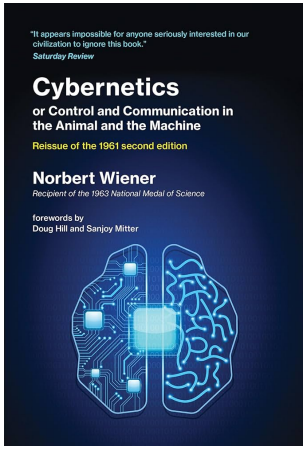
Control, Problemas Inversos y Aprendizaje Automático
Almagro, December 2-3 2024

European Commission
European Research Council
Established by the European Commission
Deusto
Universidad de Deusto
Deustuko Unibertsitatea
University of Deusto

Control and ML: two neighboring fields

CYBERNETICS: the science of communication and control in animals and machines.

Norbert Wiener, 1948



Two essential conceptual binomials involved in this definition

control-communication: sufficient and quality information about the state of a system is needed to make the right decisions, to reach a given objective or simply to avoid risky regimes to be eluded.

animal-machine: the human being, a rational animal, aims to build machines to carry out those tasks that prevent him from dedicating his time and energy to more sublime activities.

The close link between control and/or cybernetics and Machine Learning is built in Wiener's own definition.

Control and ML: two neighboring fields

It is often hard to observe the interconnections between different mathematical disciplines since they are split by conceptual and technical mountain ranges, and often have evolved in different communities. Building the connecting paths requires an important effort of abstraction.

Exploring the notion of controllability helps disclosing one of those gateways.

CONTROLLABILITY: driving a dynamical system from an initial configuration to a final one, in a given time horizon, using skillfully designed and viable controls.

LINEAR FINITE DIMENSIONAL=KALMAN

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$\Downarrow$$

$$\text{rank}\left[B|AB|A^2B|\dots|A^{N-1}B\right] = N$$

The **size of the control** depends on the **length of the time horizon**: in short time horizons the control is enormous while in longer time horizons it has a smaller amplitude.

Are these ideas and methods relevant at all in Machine Learning?

Supervised learning

Goal: classification

Match points (images) to respective labels.

Technique

Approximate a function $f_\theta : \mathbb{R}^d \to \mathbb{R}^d$, depending on some unknown parameters $\theta \in \Theta$, using a dataset

$$\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N \subset \mathbb{R}^d \times \mathbb{R}^d \quad \text{s.t. } \boldsymbol{y}_i = f_\theta(\boldsymbol{x}_i)$$

This is typically done by **training a neural network**. We will do it through **simultaneous or ensemble control of Neural Ordinary Differential Equations(NODEs)**.

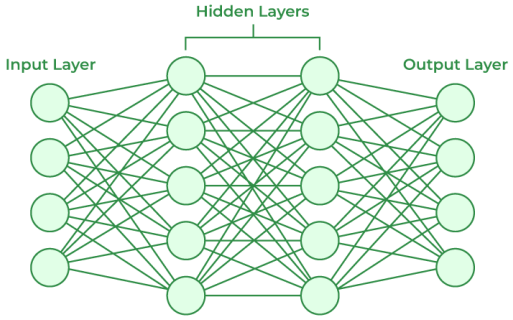
Supervised learning

Residual Neural Network (ResNet)

For each item $i \in \{1, \dots, N\}$

$$\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^k + h\boldsymbol{W}^k \sigma(\boldsymbol{A}^k \boldsymbol{x}_i^k + \boldsymbol{b}^k), \quad k \in \{1, \dots, L-1\}$$

$(\boldsymbol{W}^k, \boldsymbol{A}^k, \boldsymbol{b}^k) \in \mathbb{R}^{d \times p} \times \mathbb{R}^{p \times d} \times \mathbb{R}^p$: parameters



σ: (nonlinear) **activation function**

$L > 1$: number of layers (depth)

$p \geq 1$: neurons per layer (width)

GOAL: find $(\boldsymbol{W}^k, \boldsymbol{A}^k, \boldsymbol{b}^k)$ such that $\boldsymbol{x}_i^L = \boldsymbol{y}_i$

In the limit $L \to +\infty$, this constitutes a simultaneous control problem for the **Neural Ordinary Differential Equation**

$$\dot{\boldsymbol{x}}_i(t) = \boldsymbol{W}(t)\sigma\Big(\boldsymbol{A}(t)\boldsymbol{x}_i(t) + \boldsymbol{b}(t)\Big), \quad t \in (0, T)$$

where $(\boldsymbol{W}, \boldsymbol{A}, \boldsymbol{b}) \in L^\infty((0,T), \mathbb{R}^{d \times p} \times \mathbb{R}^{\times d} \times \mathbb{R}^p)$ are control functions that must steer each data point to its corresponding label: $\boldsymbol{x}_i(T) = \boldsymbol{y}_i$ for all $i \in \{1, \dots, N\}$.

Supervised learning

How do we achieve that? **Empirical risk minimization**: minimize under the ResNet dynamics

$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\text{loss}(\boldsymbol{x}_i,\boldsymbol{y}_i)}_{\text{empirical risk}}+\lambda\underbrace{\sum_{k=1}^{L}\|(\boldsymbol{W}^k,\boldsymbol{A}^k,\boldsymbol{b}^k)\|^2}_{\text{regularization}}$$

$$\downarrow L\to+\infty$$

$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\text{loss}(\boldsymbol{x}_i(T),\boldsymbol{y}_i)}_{\text{empirical risk}}+\lambda\underbrace{\int_0^T\|(\boldsymbol{W}(t),\boldsymbol{A}(t),\boldsymbol{b}(t))\|^2\,dt}_{\text{regularization}}$$

Difficult problem due to

computational complexity: N is most-often very large

lack of convexity coming form the non-linearity of the neural network

curse of dimensionality: as the data dimension increases, the amount of data needed to properly minimize empirical risk grows exponentially.

Supervised learning

Why does it work? The answer is **universal approximation**: finite combinations of rescaled and shifted activation functions

$$\sum_{k=1}^{L} W_k \sigma\left(A_k x + b_k\right) \leftarrow \text{Cybenko ansatz}$$

are dense in a variety of functional classes.

G. Cybenko, Approximation by superpositions of a sigmoidal function, 1989.

By universal approximation, whatever the data set is, it can be classified simply because we can approximate with the Cybenko ansatz any characteristic function taking value 1 in one set of items and 0 in the complementary one, allocating in this way the right label to each item.

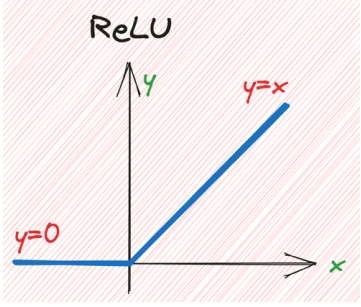
Given that universal approximation guarantees that all goals can be achieved by identifying the right parameters in Cybenko's ansatz, we may adopt the least squares point of view and look for those parameter values minimizing the distance to the needed function, in the so-called **training phase**.

Classification by simultaneous control

Theorem

Consider the NODE $\dot{\boldsymbol{x}}(t) = \boldsymbol{w}(t)\sigma\big(\boldsymbol{a}(t)\cdot\boldsymbol{x}(t) + b(t)\big)$ with $t \in (0, T)$, $\boldsymbol{w}, \boldsymbol{a} \in \mathbb{R}^d$, $b \in \mathbb{R}$ and ReLU activation function

$$\sigma(x) = \mathrm{ReLU}(x) = \max\{x, 0\}$$



Then, in any time horizon $[0, T]$, a finite arbitrary number N of items can be driven to an arbitrary number of open subsets of the Euclidean space corresponding to its labels. Moreover

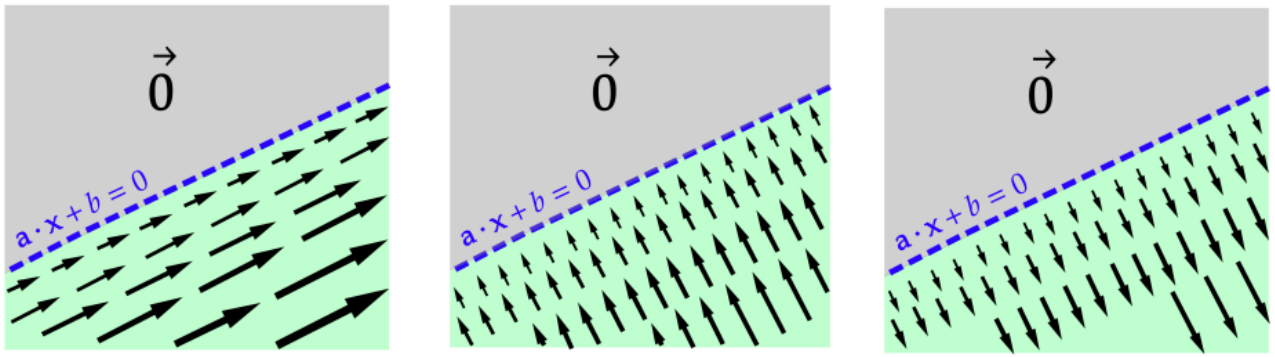
- Controls are piecewise constant with at most $\mathcal{O}(N)$ switches.
- The control time $T > 0$ can be made arbitrarily small (absorbed into $\|w\|$ by scaling).

D. Ruíz-Balet and E. Zuazua, *Neural ODE control for classification, approximation, and transport*, SIAM Rev., 2023

What is the ResNet doing? Basic control actions

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{w}(t)\sigma\left(\boldsymbol{a}(t)\cdot\boldsymbol{x}(t) + b(t)\right) \rightarrow \text{flow map } \phi_T : \mathbb{R}^d \rightarrow \mathbb{R}^d$$

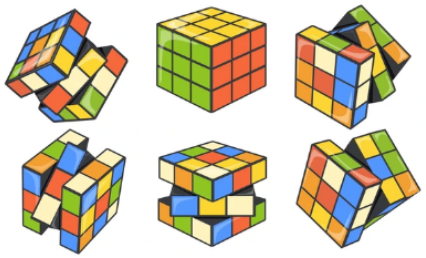
- Control functions $(\boldsymbol{w}, \boldsymbol{a}, b) \longrightarrow$ piecewise constant
- Each time discontinuity $\sim$ one change of layer
- $\boldsymbol{a}(t)$ and $b(t)$ define a hyperplane $H(x) = \boldsymbol{a}(t)\cdot\boldsymbol{x}(t) + b(t) = 0$ in $\mathbb{R}^d$
- $\sigma(x) = \max\{x, 0\}$ activates $H(x) > 0$ and freezes $H(x) \leq 0$
- $\boldsymbol{w}(t)$ gives the direction of the field inside the half-space $H(x) > 0$

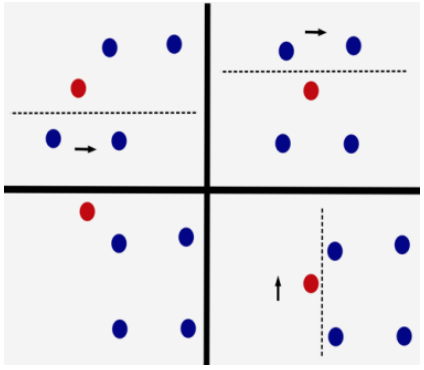


What is the ResNet doing? Basic control actions

The very features of the ReLU enable the realization of the simultaneous control goal. The fact that σ preserves one half-space invariant while deforming the other half-space introduces dynamics not found in mechanics, where such simultaneous control is unlikely.

The NODE driven by the ReLU behaves like a Rubik Cube, **solvable in a finite number of smart operations** in which part of the cube is frozen, while the other one rotates in the appropriate direction and sense. The goal in the Rubik Cube game is to ensure that all faces are homogeneous in color, similar to the task that NODEs have to perform: drive each initial item to a given target.





A strategic **inductive choice** of the different hyperplanes (via the controls $\boldsymbol{a}$ and $\boldsymbol{b}$) and of the direction of the dynamics, through the control $\boldsymbol{w}$, guarantees classification in a finite number of steps

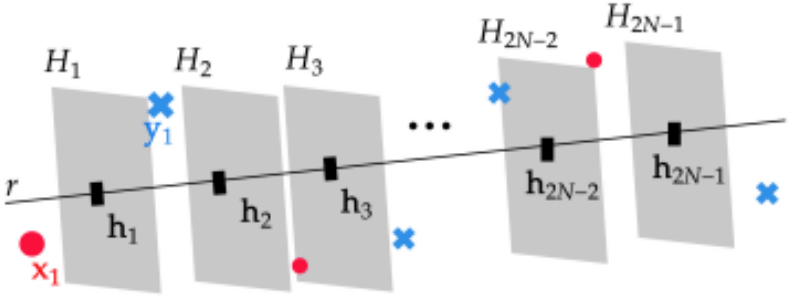
Can we do better?

Key idea for an optimal classification: identifying a **minimal set of hyper-planes** that can separate the points in the dataset according to their labels. The required number of hyperplanes is a **measure of the complexity of the classifier**.

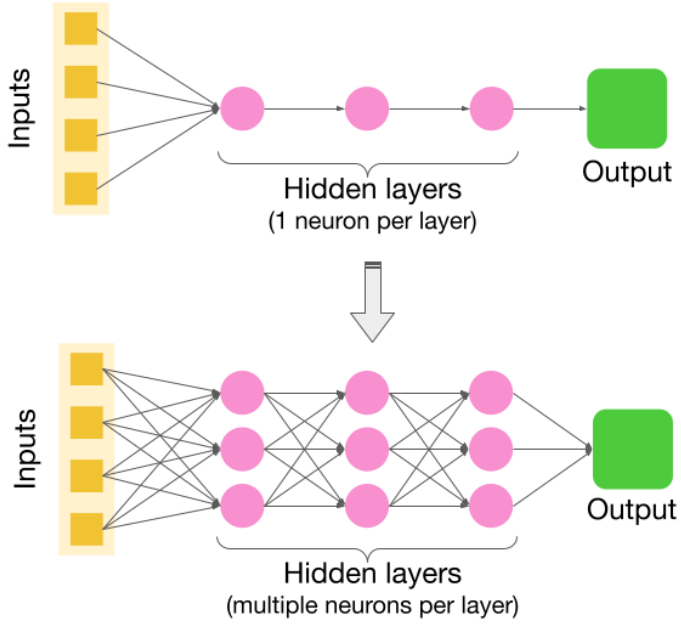
Two possible way of reducing complexity:

Optimize the classification algorithm:

define sharply the hyperplanes and controls that accomplish the classification task



Increase the width p: use more neurons



Optimize the classification algorithm

Classification through NODEs is intrinsically related with organizing the data points according to their labels using hyper-planes.

WARNING!!

Not all hyper-planes families is adequate for classification. For example, if we were to divied our domain in polygonal cells separating the points of different classes, it would be not at all clear how to move them.

Classification via NODEs is **more restrictive** than separating the points and must be done properly. Actually, we can ensure the existence of an effective control $\boldsymbol{w}$ only when the hyper-planes have an appropriate structure.

Optimize the classification algorithm

Theorem

Let $2 \leq d \leq 2N$. Consider a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{2N} \subset \mathbb{R}^d \times \{1, 0\}$ such that $\{\boldsymbol{x}_i\}_{i=1}^{N} \subset \mathbb{R}^d$ and $\{\boldsymbol{x}_{i+N}\}_{i=i}^{N} \subset \mathbb{R}^d$ are in **general position**. Then

- For any $T > 0$, there exists piecewise constant controls

$$(\boldsymbol{w}, \boldsymbol{a}, \boldsymbol{b}) \in L^\infty((0, T); \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R})$$

  such that

$$\phi_T(\boldsymbol{x}_i) > 1 \quad \text{and} \quad \phi_T(\boldsymbol{x}_{i+N}) < 1, \quad \text{for all } i =\in \{1, \dots, N\}$$

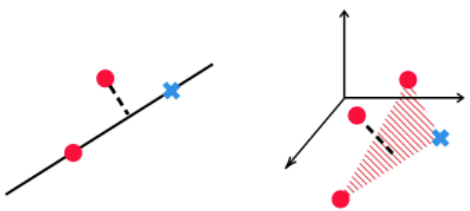
- The number of discontinuities of $(\boldsymbol{w}, \boldsymbol{a}, \boldsymbol{b})$ is of the order $\mathcal{O}\left(1 + \frac{N}{d}\right)$.

A. Alvarez-López, R. Orive and E. Zuazua, Optimized classification with neural ODEs via separability, 2023

Optimize the classification algorithm

General position

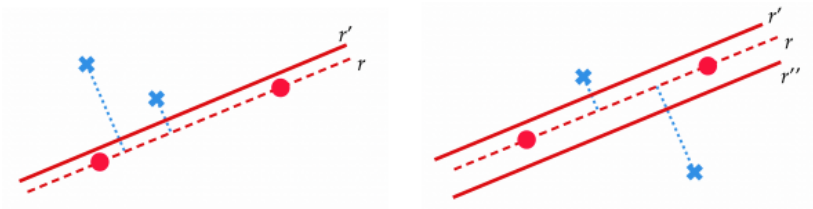
A set X of N points in $\mathbb{R}^d$, with $N \geq d$, is in **general position** if no hyperplane of dimension $d-1$ contains more than d points of X.



X is in general position in $\mathbb{R}^2$ if no three points lie on the same line

X is in general position in $\mathbb{R}^3$ if there are no four points on the same plane

A dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{2N} \subset \mathbb{R}^d \times \{1, 0\}$ such that $\{\boldsymbol{x}_i\}_{i=1}^{N} \subset \mathbb{R}^d$ and $\{\boldsymbol{x}_{i+N}\}_{i=i}^{N} \subset \mathbb{R}^d$ are in general position can be separated with at most $2\lceil N/2d \rceil$ hyperplanes.



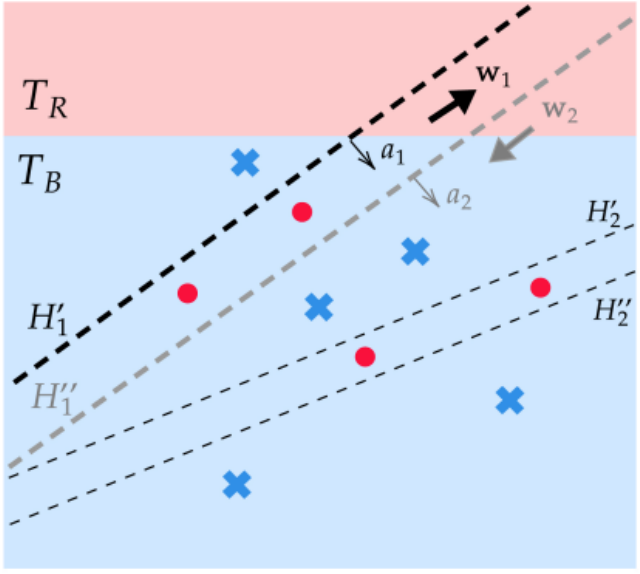
Optimize the classification algorithm

Strategy

Step 1 - isolate points according to their classes: find a family of pairwise parallel hyperplanes

$$\mathcal{H} = \left\{H'_1, H''_2, \ldots, H'_{\lceil N/d \rceil}, H''_{\lceil N/d \rceil}\right\}$$

so that the region bounded by the pair (H'_i, H''_i) contains only points of the same class. This corresponds to find suitable controls $\boldsymbol{a}$ and $\boldsymbol{b}$.
Step 2 - control: move each group of point to its corresponding region by means of suitable designed controls $\boldsymbol{w}$.



To move the two red points between H'_1 and H''_1 to the red region T_R:

1. we move in direction $\boldsymbol{w}_1$ the region defined by $\boldsymbol{a}_1$.
2. we move in direction $\boldsymbol{w}_2$ the region defined by $\boldsymbol{a}_2$.

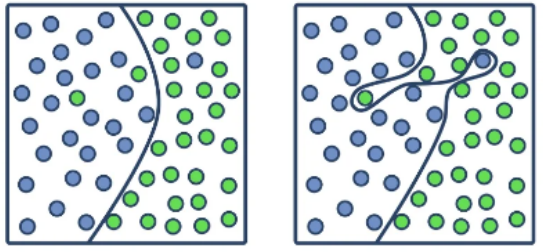
Optimize the classification algorithm

The proposed methodology can classify any set of points, independently of their distribution. At the same time, it may not be useful in real-world classification problems.

The algorithm characterizes the capacity of a NODE model to classify $2N$ points in the worst-case scenario. It provides an upper bound of the controls complexity sufficient to induce overfitting. Said differently, the algorithm can serve to know when overfitting arises, depending on the model and dataset dimensions.

Overfitting happens when a model learns the training data too well, capturing not only the underlying patterns but also the noise and outliers. This leads to poor performance on new unseen data, because the model is too complex.

The good functioning of any real-life algorithm requires **avoiding overfitting**.



Increase the network's width

1 **neuron** (p=1) **multiple neurons** (p>1)

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{w}(t)\sigma(\boldsymbol{a}(t)\cdot\boldsymbol{x}(t) + b(t)) \quad \longrightarrow \quad \dot{x}(t) = \sum_{j=1}^{p} \boldsymbol{w}_j(t)\sigma(\boldsymbol{a}_j(t)\cdot\boldsymbol{x} + b_j) \quad (1)$$

$\boldsymbol{w}_j$: columns of $\boldsymbol{W} \in \mathbb{R}^{d\times p}$
$\boldsymbol{a}_j$: rows of $\boldsymbol{A} \in \mathbb{R}^{p\times d}$
b_j : coordinates of $\boldsymbol{b} \in \mathbb{R}^{d}$

Theorem

Let $d \geq 2$ and $T > 0$ be fixed. Consider a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N} \subset \mathbb{R}^d \times \mathbb{R}^d$ such that $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ and $\boldsymbol{y}_i \neq \boldsymbol{y}_j$ for all $i,j \in \{1,\dots,N\}$. Then

- For any $p \geq 1$, there exists piecewise constant controls

$$(\boldsymbol{W}, \boldsymbol{A}, \boldsymbol{b}) \in L^{\infty}((0,T); \mathbb{R}^{d\times p} \times \mathbb{R}^{p\times d} \times \mathbb{R}^{p})$$

  such that the flow ϕ_T generated by (1) **interpolates** $\mathcal{D}$, i.e., $\phi_T(\boldsymbol{x}_i) = \boldsymbol{y}_i$, for all $i \in \{1,\dots,N\}$.

- The number of discontinuities of $(\boldsymbol{W}, \boldsymbol{A}, \boldsymbol{b})$ is of the order $\mathcal{O}\left(\frac{N}{p} - 1\right)$.

A. Alvarez-López, A. Slimaneb and E. Zuazua, Interplay between depth and width for interpolation in neural ODEs, 2024

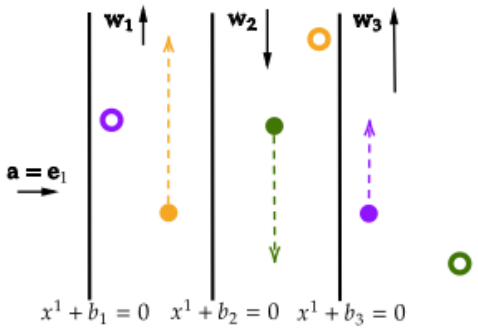
Increase the network's width

Strategy

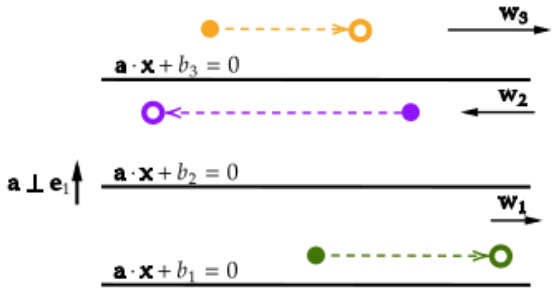
Step 1 - control $d-1$ coordinates: for all $j \in \{1, \ldots, p\}$ we take $\boldsymbol{a}_j = \boldsymbol{e}_1$ and $b_j \in \mathbb{R}$ satisfying $-b_1 < x_1^{(1)} < -b_2 < x_2^{(1)} < \ldots < -b_p < x_p^{(1)}$. This selection is possible up to a change of coordinates.

These controls confine the trajectory of each data point $\boldsymbol{x}_i$ within the hyperplane $x_j^{(1)} = b_j$. By properly choosing the velocities $\boldsymbol{w}_j$, we can control $d-1$ coordinates of these points using $\mathcal{O}\left(\frac{N}{p}\right)$ switches.

Step 2 - control the last coordinate: use the controlled coordinates and suitable designed velocities $\boldsymbol{w}_j$ to adjust $x_i^{(1)}$. This also can be done with $\mathcal{O}\left(\frac{N}{p}\right)$ switches.



Control $d-1$ coordinates



Control the remaining coordinate

Increase the network's width

Remark

If the target points $\{\boldsymbol{y}_i\}_{i=1}^N$ are not distinct, interpolation is not achievable due to the uniqueness of solutions in the NODE. In such cases, we can relax the statement from **exact** to **approximate** controllability by applying the theorem to an ε-perturbation of the targets.

Additional bibliography

- D. Ruiz-Balet and E. Zuazua, *Control of neural transport for normalizing flows*, J. Math. Pres et Appl., 2024.
- Z. Li, K. Liu, L. Liverani and E. Zuazua, *Universal approximation of dynamical systems by semi-autonomous Neural ODEs and applications*, preprint, 2024
- M. Hernández and E. Zuazua, *Deep neural networks: multi-classification and universal approximation*, preprint, 2024
- Y. Song, Z. Wang and E. Zuazua, *Approximate and weighted data reconstruction attack in Federated Learning*, preprint, 2023
- Y. Song, Z. Wang, E. Zuazua, *FedADMM-InSa: an Inexact and self-Adaptive ADMM for Federated Learning*, Neural Networks, 2024
- B. Geshkovski and E. Zuazua, *Turnpike in optimal control of PDEs, ResNets, and beyond*, Acta Num., 2022
- C. Esteve and B. Geshkovski, *Sparsity in long-time control of neural ODEs*, Syst. Control. Lett., 2023
- B. Geshkovski, C. Letrouit, Y. Polyanskiy and P. Rigollet, *The emergence of clusters in self-attention dynamics*, Adv. Neur. Inf. Process. Syst., 2024.
- B. Geshkovski, C. Letrouit, Y. Polyanskiy and P. Rigollet, *A mathematical perspective on transformers*, preprint, 2024
- A. Alcalde, G. Fantuzzi and E. Zuazua, *Clustering in pure-attention hardmax transformers and its role in sentiment analysis*, preprint, 2024

ERC Advanced Grant CoDeFeL

ERC Advanced Grant **CoDeFeL**
PI: Enrique Zuazua

Universidad de Deusto
Friedrich-Alexander-Universität
Universidad Autónoma de Madrid

Develop new tools with a solid mathematical foundation suitable for data-aware/physics-inspired modeling, combining ML and control theoretical ideas.

Four Work Packages:

WP1: Asymptotics and Turnpike for Deep Neural Networks

WP2: Complexity of ResNet dynamics

WP3: Federated Learning

WP4: Modelling through Control and Machine Learning

Pre-doc and **postdoc** students applications are welcome.

THANKS FOR YOUR ATTENTION!!

The **CoDeFeL** project (ERC-2022-ADG) has received funding from the European Union's Horizon ERC Grants programme under grant agreement No. 101096251. The views and opinions expressed are solely those of the author(s) and do not necessarily reflect those of the European Research Council Executive Agency (ERCEA), the European Union or the granting authority who cannot be held responsible for them.

Additional funding

- PID2020-112617GB-C22-**KiLearn** (MINECO)
- PID2023-146872OB-I00-**DyCMaMod** (MINECO)
- TED2021-131390B-I00-**DasEl** (MINECO)

European Research Council

Established by the European Commission