Control-based classification with neural ordinary differential equations

X Partial differential equations, optimal design and numerics

Antonio Álvarez-López

Joint work with Rafael Orive Illera and Enrique Zuazua

Department of Mathematics,
Universidad Autónoma de Madrid

October 14, 2024

Supervised Learning

Goal

Input space $(\mathcal{X}, \mu^*) \subset \mathbb{R}^d \xrightarrow{\quad F^* \quad}$ Output space $\mathcal{Y} \subset \mathbb{R}^m$

Approximate (*learn*) F^* from a dataset $\mathcal{D} := \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N \subset \mathcal{X} \times \mathcal{Y}$:

$$\mathbf{x}_n \sim \mu^*, \qquad \mathbf{y}_n = F^*(\mathbf{x}_n), \qquad n = 1, \dots N.$$

Classification

$\downarrow$

range(F^*) is finite

$x_1 \in \mathbb{R}^{3N_{pixels}}$

$x_2 \in \mathbb{R}^{3N_{pixels}}$

F^*

$y_1 = (1, 0) \equiv$ Do not enter

$y_2 = (0, 1) \equiv$ Proceed

Motivation

E. Weinan, "A proposal on machine learning via dynamical systems" (2017).

In this note, we go one step further, by exploring the possibility of producing nonlinear functions using continuous dynamical systems, pushing the compositional approach to an infinitesimal limit. In the framework of supervised learning, this gives rise to a new class of control problems. In this view, the deep neural networks can be thought of as being discrete dynamical systems. Compared with deep neural networks, there are several potential advantages with a continuous approach.

1. Mathematically it is easier to think about and deal with continuous dynamical systems. Continuous formulation offers more flexibility (for example adding constraints, adapting the dynamical system to the problem, imposing structure on the dynamical system), and it is easier to analyze continuous dynamical systems than discrete ones.
2. Deep neural network can be thought of as a discretization of the continuous dynamical systems. However, from the viewpoint of discretizing dynamical systems, there are many possibilities that one can explore. For example, one can use adaptive time step size, which corresponds to choosing the layers adaptively. One can use high order or even implicit discretization, and these do not yet have an analog in deep neural networks. One can also use advanced numerical methods for training, such as the multi-grid method or the parallel shooting method (see [4,5]).
3. Most models in physical sciences (physics, chemistry, etc) are represented using dynamical systems in the form of differential equations. The continuous dynamical systems approach makes it easier to combining ideas from machine learning and physical modeling.
4. The vast majority of the applied mathematics community is familiar with differential equations. The continuous dynamical systems approach to machine learning will be of particular interest to them.

Neural ODEs (one-neuron width)

$$\dot{\mathbf{x}}(t) = \mathbf{w}(t)\,\sigma(\mathbf{a}(t)\cdot\mathbf{x}(t) + b(t)), \qquad \mathbf{x}(t) \in \mathbb{R}^d \ (d \geq 2)$$

Neural ODEs (one-neuron width)

$$\dot{\mathbf{x}}(t) = \mathbf{w}(t)\,\sigma(\mathbf{a}(t)\cdot\mathbf{x}(t) + b(t)), \qquad \mathbf{x}(t) \in \mathbb{R}^d \ (d \geq 2)$$

- **Control**: $\theta := (\mathbf{w}, \mathbf{a}, b)$, $\theta(t) \in L^\infty\left((0,T); \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}\right)$.
- **Activation**: $\underbrace{\sigma(z) = (z)_+}_{\text{ReLU}}$, $\underbrace{\sigma(z) = \min\{(z)_+, 1\}}_{\text{TrunReLU}}$ Lipschitz, nonlinear.
- **Flow map** in time T is well defined:

$$\begin{aligned} \Phi_T(\cdot;\theta) : \mathbb{R}^d &\to \mathbb{R}^d \\ \mathbf{x_0} &\mapsto \mathbf{x}(T;\mathbf{x_0}). \end{aligned}$$

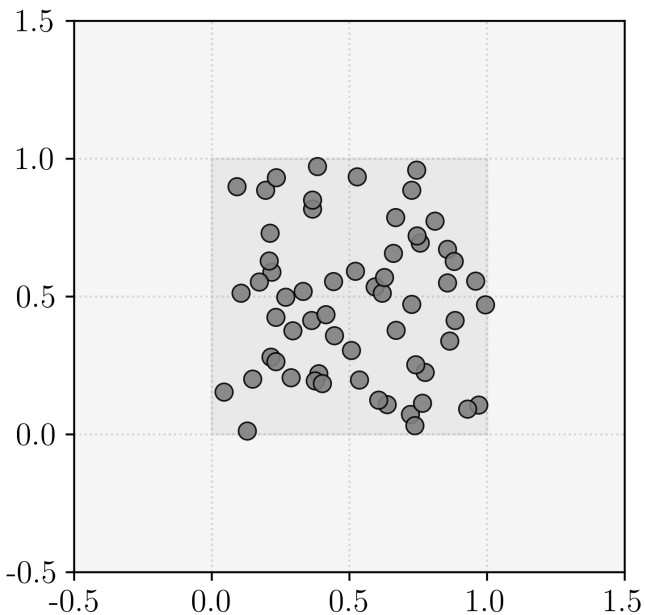
Assume θ piecewise constant in $(0,T)$, $\underbrace{L \text{ discontinuities}}_{\sim \text{ Transitions between layers}}$

$$\dot{\mathbf{x}}(t) = \underbrace{\sum_{j=1}^{L} \mathbf{w_{i,j}}\sigma\left(\mathbf{a_{i,j}}\cdot\mathbf{x} + b_{i,j}\right)\mathbf{1}_{(t_{j-1},t_j)}(t)}_{\text{Finite number of parameters}}, \qquad t \in (0,T).$$

Problem statement

- **Worst-case scenario**:

$$\mathbf{x_n} \sim U([0,1]^d) \quad \text{for all } n.$$



Framework

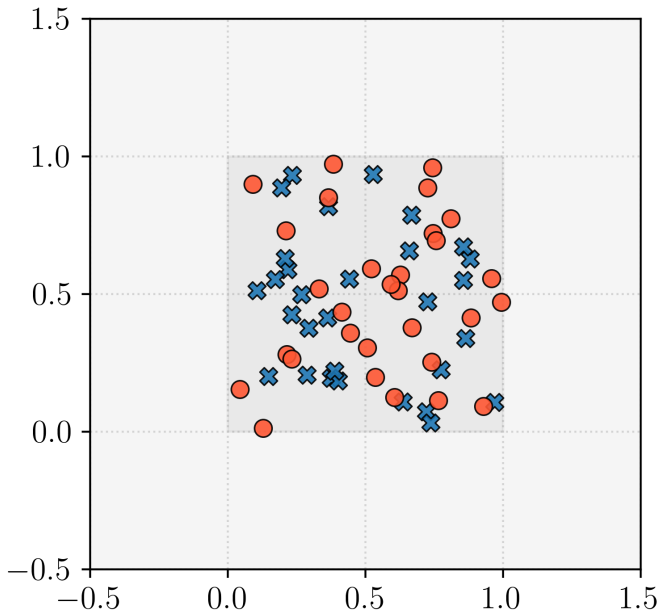
- **Worst-case scenario**:

$$\mathbf{x_n} \sim U([0,1]^d) \quad \text{for all } n.$$

- **Binary classification**:

$$y_n \in \{1,0\} \quad \text{for all } n \longrightarrow \underbrace{\{(\mathbf{x_n},1)\}}_{R}, \underbrace{\{(\mathbf{x_m},0)\}}_{B} \subset \mathcal{D}, \quad |R| = |B| = N.$$

-



Framework

- **Worst-case scenario**:

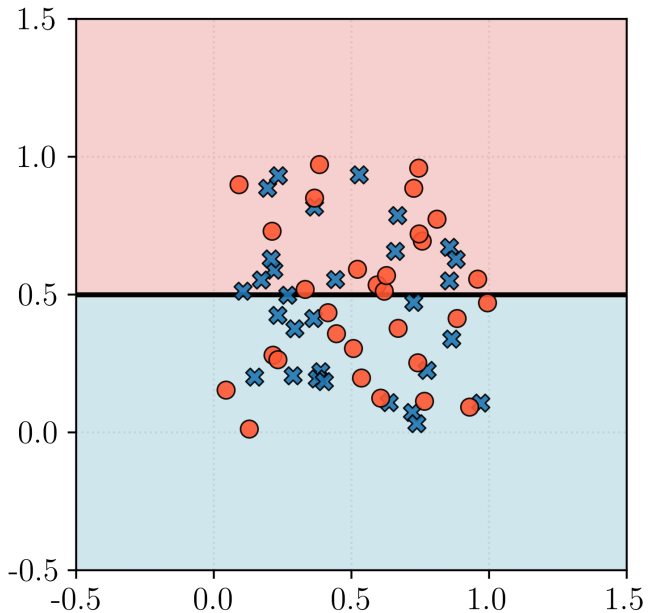
$$\mathbf{x_n} \sim U([0,1]^d) \quad \text{for all } n.$$

- **Binary classification**:

$$y_n \in \{1,0\} \quad \text{for all } n \longrightarrow \underbrace{\{(\mathbf{x_n},1)\}}_{R}, \underbrace{\{(\mathbf{x_m},0)\}}_{B} \subset \mathcal{D}, \quad |R| = |B| = N.$$

- **Target regions**:

$$1 \longleftrightarrow \Omega_1 := \{x^{(1)} > 0.5\}, \qquad 0 \longleftrightarrow \Omega_0 := \{x^{(1)} < 0.5\}.$$



Problem statement

$$\dot{\mathbf{x}}(t) = \mathbf{w}(t)\,\sigma(\mathbf{a}(t)\cdot\mathbf{x}(t) + b(t)) \tag{1}$$

For any $T > 0$, find a control $\theta = (\mathbf{w}, \mathbf{a}, b)$ for (1) such that

$$\Phi_T(\mathbf{x_n}; \theta) \in \Omega_{y_n} \qquad \text{for all } n$$

with the minimum possible complexity (number of discontinuities L).

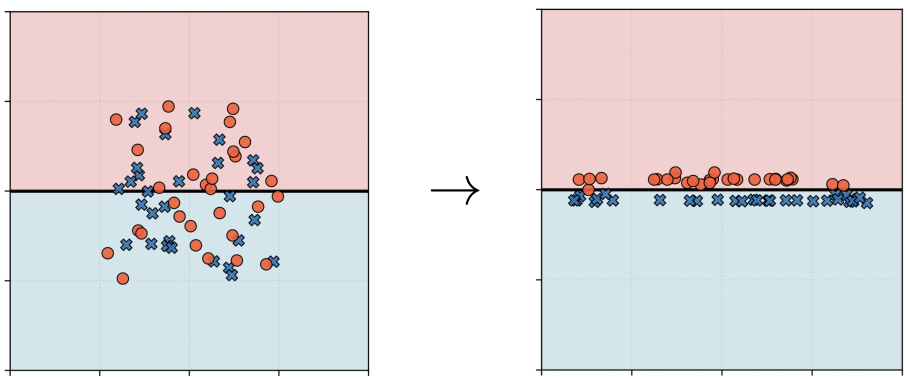
Problem statement

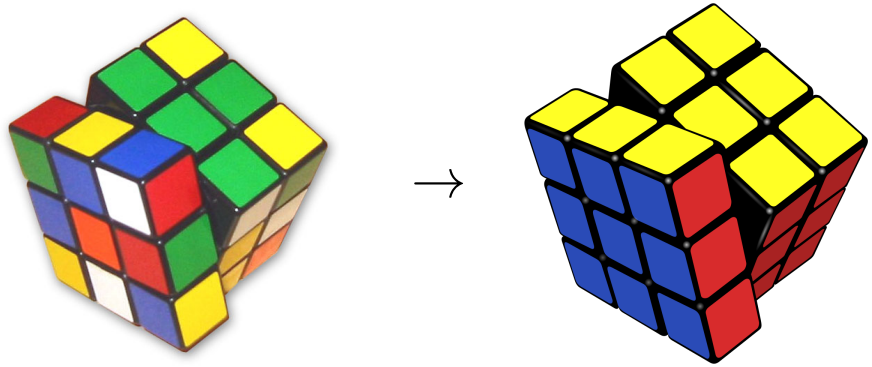
$$\dot{\mathbf{x}}(t) = \mathbf{w}(t)\,\sigma(\mathbf{a}(t)\cdot\mathbf{x}(t) + b(t)) \tag{1}$$

For any $T > 0$, find a control $\theta = (\mathbf{w}, \mathbf{a}, b)$ for (1) such that

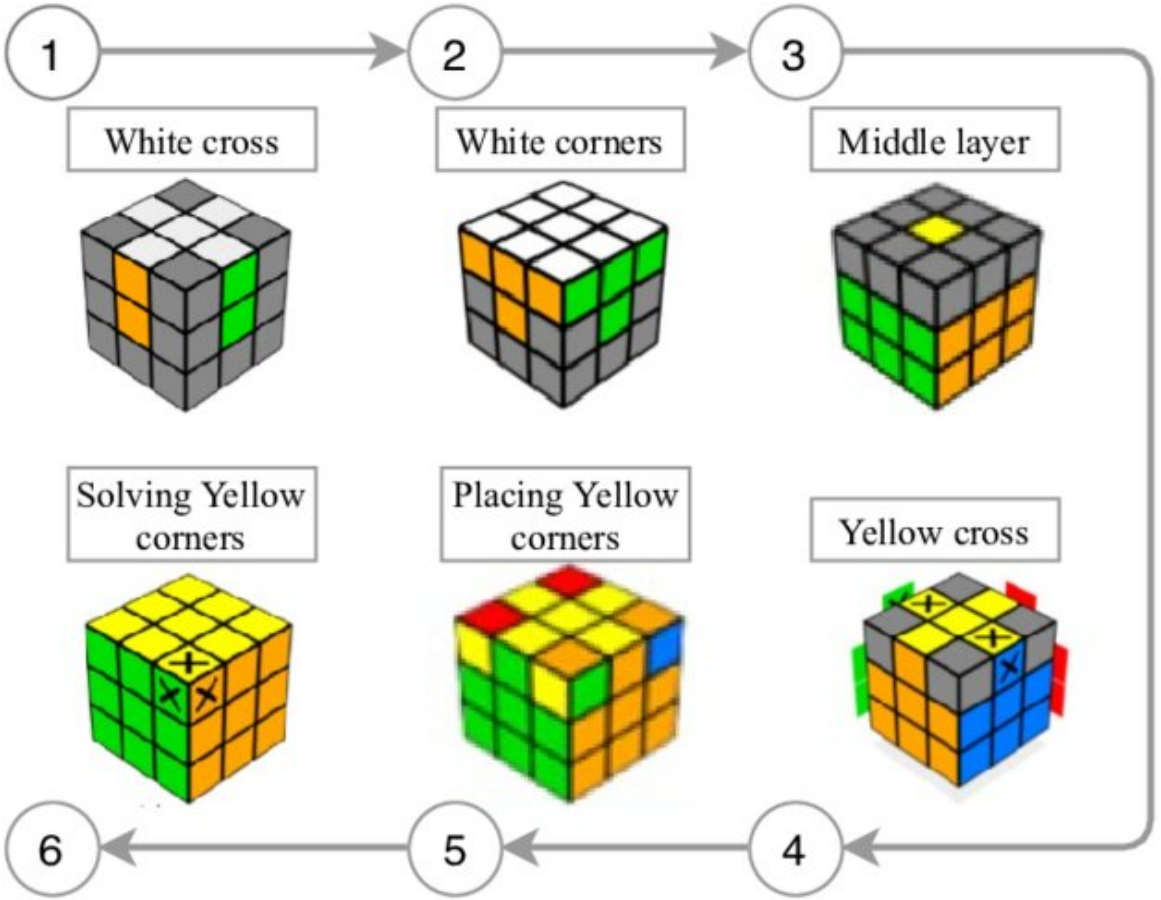
$$\Phi_T(\mathbf{x_n}; \theta) \in \Omega_{y_n} \qquad \text{for all } n$$

with the minimum possible complexity (number of discontinuities L).





Human's algorithm vs God's algorithm



Source: "Two approaches in solving Rubik's cube with Hardware-Software Co-design", E. Barucija et al.

Human's algorithm vs God's algorithm

God's algorithm is a notion originating in discussions of ways to solve the Rubik's Cube puzzle,[1] but which can also be applied to other combinatorial puzzles and mathematical games.[2] It refers to any algorithm which produces a solution having the fewest possible moves. The allusion to the deity is based on the notion that an omniscient being would know an optimal step from any given configuration.

> An algorithm to determine the minimum number of moves to solve Rubik's Cube was published in 1997 by Richard Korf.[10] While it had been known since 1995 that 20 was a lower bound on the number of moves for the solution in the worst case, Tom Rokicki proved in 2010 that no configuration requires more than 20 moves.[11] Thus, 20 is a sharp upper bound on the length of optimal solutions. Mathematician David Singmaster had "rashly conjectured" this number to be 20 in 1980.[4]

Source: Wikipedia

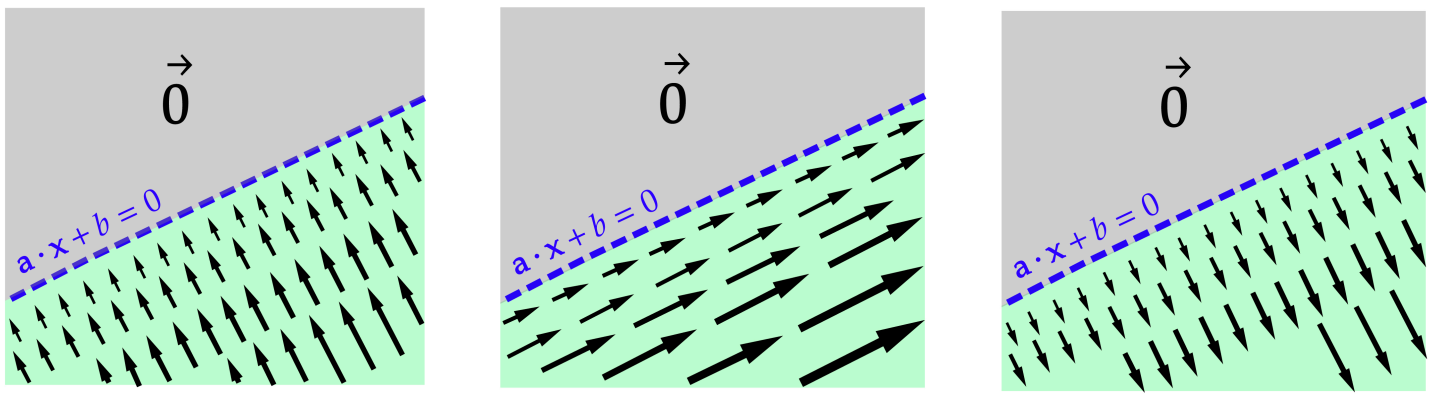
How to move our Rubik's cube

$$\dot{\mathbf{x}}(t) = \mathbf{w}(t)\,\sigma(\mathbf{a}(t)\cdot\mathbf{x}(t) + b(t))$$

- $\mathbf{a}(t), b(t)$ determine the hyperplane in $\mathbb{R}^d$ given by

$$H(\mathbf{x}) = \mathbf{a}(t)\cdot\mathbf{x} + b(t) = 0.$$

- $\sigma(z) = (z)_+$ "activates" $H(\mathbf{x}) > 0$ and "freezes" $H(\mathbf{x}) \leq 0$.
- $\mathbf{w}(t)$ determines the direction of the field in $H(\mathbf{x}) > 0$.



From left to right: Compression, laminar motion, expansion.

Prior work

Theorem (D. Ruiz-Balet, E. Zuazua)

Let $\{\mathbf{x_n}, \mathbf{x_n}\}_{n=1}^N \subset \mathbb{R}^d$. *For any* $T > 0$, *there exists a control*

$$\theta \in L^\infty\left((0,T); \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}\right)$$

that classifies the dataset in the sense that for all $1 \leq n \leq N$:

$$\Phi_T(\mathbf{x_n}; \theta)^{(1)} > 0.5 \quad \text{and} \quad \Phi_T(\mathbf{x_n}; \theta)^{(1)} < 0.5.$$

Furthermore, θ *is piecewise constant and the number of discontinuities is*

$$L = 6N + 1$$

Prior work

Theorem (D. Ruiz-Balet, E. Zuazua)

Let $\{\mathbf{x_n}, \mathbf{x_n}\}_{n=1}^N \subset \mathbb{R}^d$. *For any* $T > 0$, *there exists a control*

$$\theta \in L^\infty\left((0,T); \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}\right)$$

that classifies the dataset in the sense that for all $1 \leq n \leq N$:

$$\Phi_T(\mathbf{x_n}; \theta)^{(1)} > 0.5 \quad \textit{and} \quad \Phi_T(\mathbf{x_n}; \theta)^{(1)} < 0.5.$$

Furthermore, θ *is piecewise constant and the number of discontinuities is*

$$L = 6N + 1$$

Question 1 (Minimax). What is the **minimum** value of L that ensures **any** given dataset can be classified? We improve the prior bound to

$$L = 4\left\lceil \frac{N}{d} \right\rceil$$

Question 2 (Probabilistic estimation). What is the probability that a given dataset can be classified using $L = k$ discontinuities, for any $k \geq 0$?

Prior work: Point-by-point algorithm

Click to play in YT

D. Ruiz-Balet, E. Zuazua, ‘‘Neural ODE control for classification, approximation, and transport’’.

Question 1

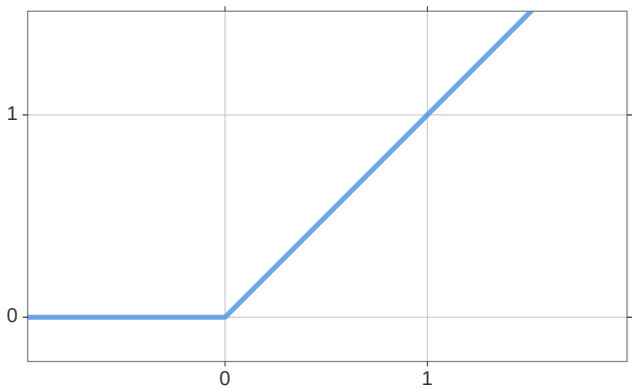
Theorem (A. Á-L, R. Orive-Illera, E. Zuazua)

Let $\{\mathbf{x_n}, \mathbf{x_n}\}_{n=1}^N \subset \mathbb{R}^d$. *For any* $T > 0$, *there exists a control*

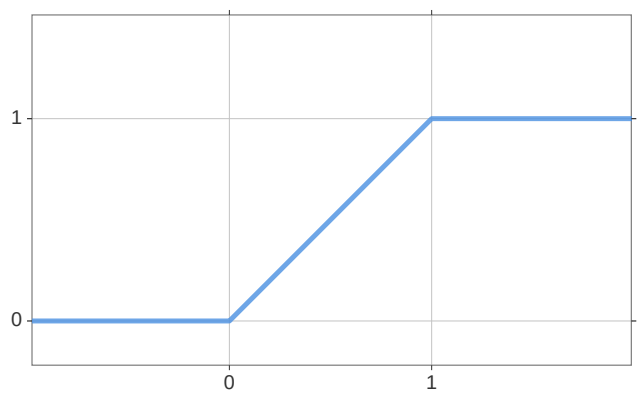
$$\theta \in L^\infty\left((0,T); \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}\right)$$

that classifies the dataset.
Furthermore, θ *is piecewise constant and the number of discontinuities is*

$$L = 4\left\lceil \frac{N}{d} \right\rceil \quad \text{if } \sigma = \text{ReLU}, \qquad L = 2\left\lceil \frac{N}{d} \right\rceil \quad \text{if } \sigma = \text{TrunReLU}.$$



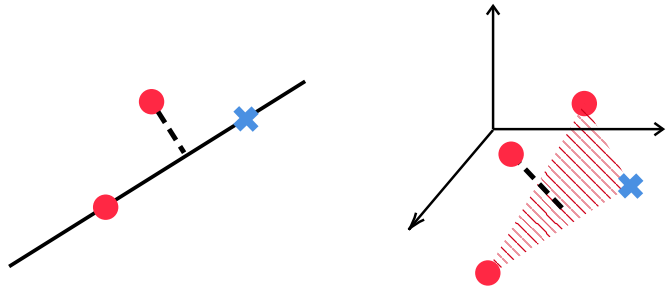
ReLU



Truncated ReLU

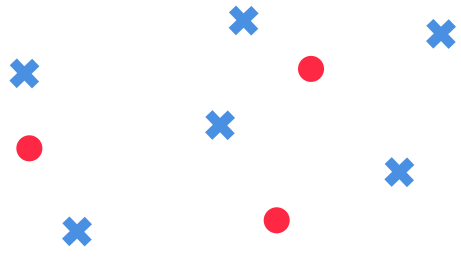
Idea of the proof. Step 1: General position

Any set in $\mathbb{R}^d$ is in **general position** if no subset of $d+1$ points lies on a common hyperplane.

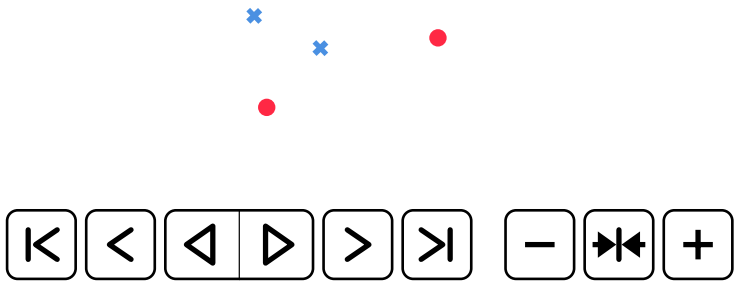


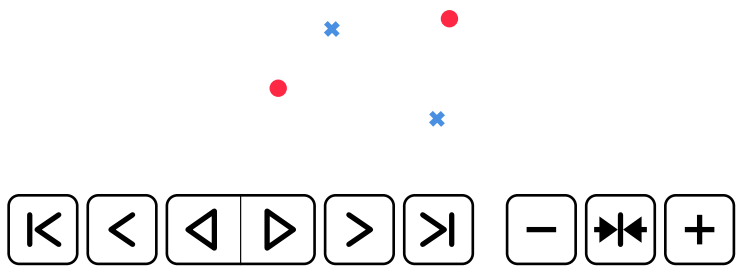
Examples in $\mathbb{R}^2$ and $\mathbb{R}^3$.

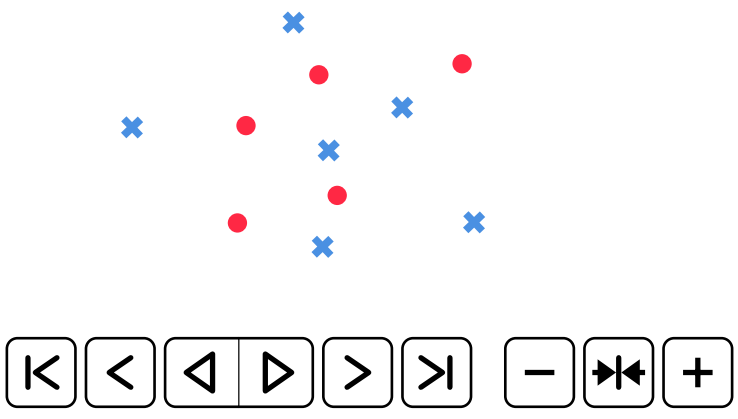
Preconditioning: $\{\mathbf{x_n}, \mathbf{x_n}\}_{n=1}^N$ can be evolved through (1) into general position for almost any $\theta \in \mathbb{R}^{2d+1}$ in an arbitrarily small time.

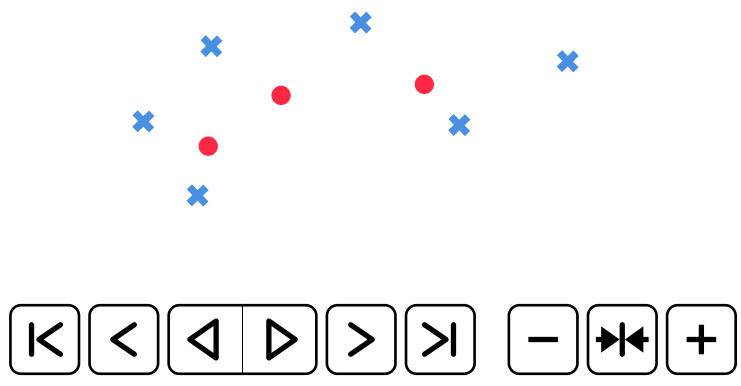


Idea of the proof. Step 2: Separation

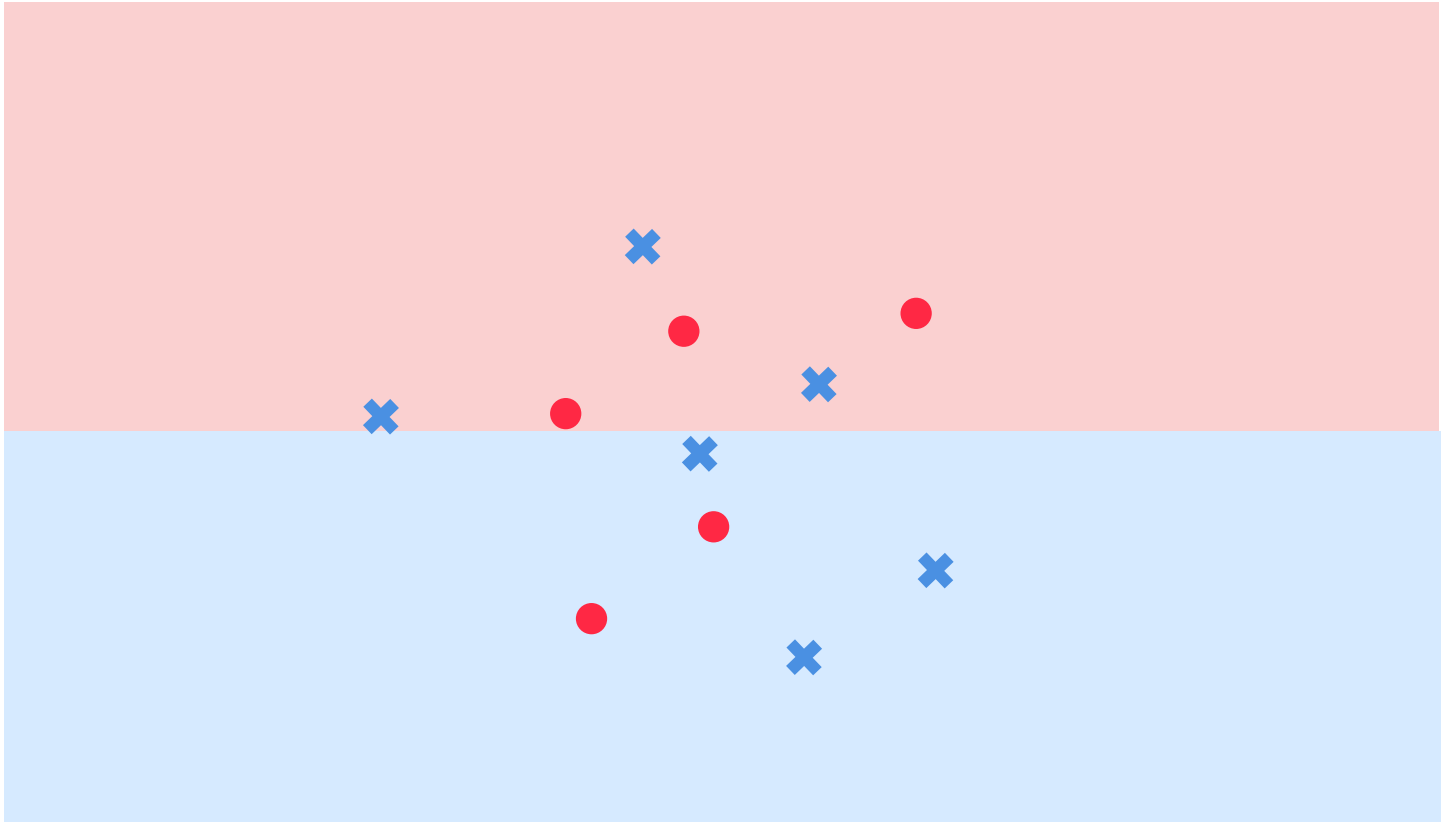




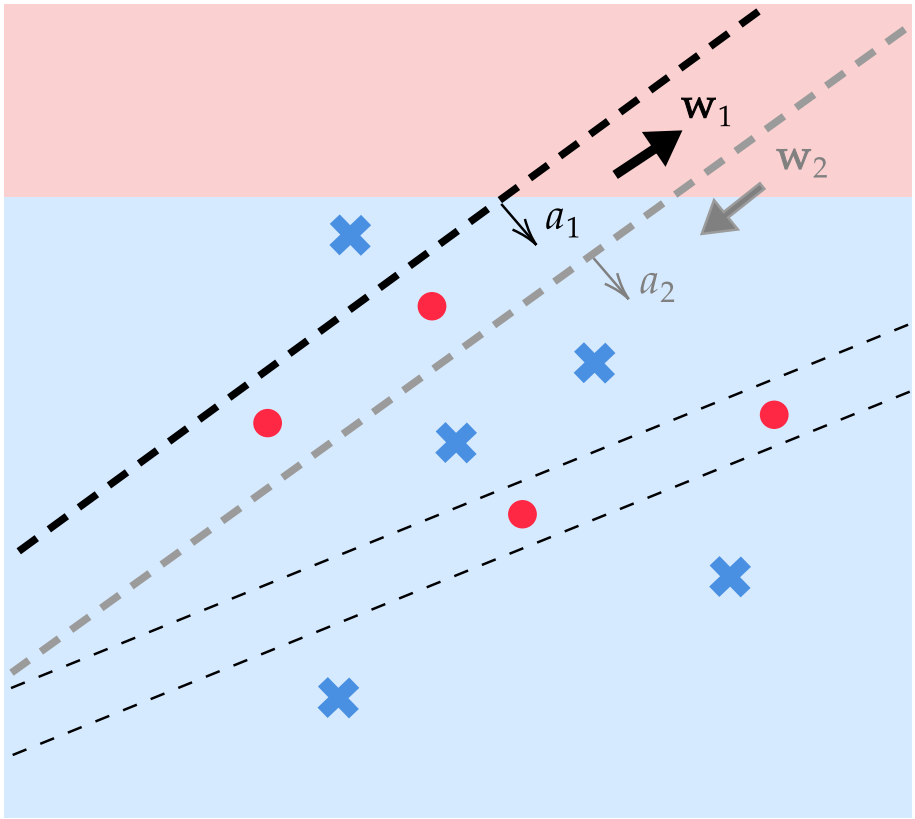




Idea of the proof. Step 3: Dynamics



Idea of the proof: Summary



We have thus overcome (and exploited) the curse of dimensionality!

Question 2

Theorem (A. Á-L, R. Orive-Illera, E. Zuazua)

Assume $\mathbf{x_n}, \mathbf{x_n} \sim U\left([0,1]^d\right)$ *for all n. For any* $T > 0$*, there exists* $\theta = (\mathbf{w}, \mathbf{a}, b)$ *where*

$$\mathbf{a} \in \{\mathbf{e_1}, \dots, \mathbf{e_d}\} \subset \mathbb{S}^{d-1} \quad \textit{const}, \qquad (\mathbf{w}, b) \in L^\infty((0,T); \mathbb{R}^d \times \mathbb{R}),$$

that classifies the dataset.
Furthermore, θ *is piecewise constant with L discontinuities, and satisfies:*

$$\mathbb{P}(L=0) = 1 - \left(\frac{2(N!)^2}{(2N)!}\right)^d \sim 1 - \exp\left\{-\frac{\sqrt{\pi N}}{2^{2N-1}} d\right\} \longrightarrow 1 \quad \textit{iff} \quad d \gtrsim 2^N/\sqrt{N}.$$

Question 2

Theorem (A. Á-L, R. Orive-Illera, E. Zuazua)

Assume $\mathbf{x_n}, \mathbf{x_n} \sim U\left([0,1]^d\right)$ *for all n. For any* $T > 0$, *there exists* $\theta = (\mathbf{w}, \mathbf{a}, b)$ *where*

$$\mathbf{a} \in \{\mathbf{e_1}, \dots, \mathbf{e_d}\} \subset \mathbb{S}^{d-1} \quad \textit{const}, \qquad (\mathbf{w}, b) \in L^\infty((0,T); \mathbb{R}^d \times \mathbb{R}),$$

that classifies the dataset.
Furthermore, θ *is piecewise constant with L discontinuities, and satisfies:*

$$\mathbb{P}(L=0) = 1 - \left(\frac{2(N!)^2}{(2N)!}\right)^d \sim 1 - \exp\left\{-\frac{\sqrt{\pi N}}{2^{2N-1}} d\right\} \longrightarrow 1 \quad \textit{iff} \quad d \gtrsim 2^N/\sqrt{N}.$$

More generally, for $0 \leq k \leq 2N-2$:

$$\mathbb{P}(L \leq k) = 1 - \Big(\sum_{p=\lceil \frac{k}{2}+1 \rceil}^{N} \binom{N-1}{p-1}^2 + \sum_{p=\lceil \frac{k+1}{2} \rceil}^{N-1} \binom{N-1}{p} \binom{N-1}{p-1} \Big)^d \left(\frac{2(N!)^2}{(2N)!}\right)^d.$$

Question 2

Theorem (A. Á-L, R. Orive-Illera, E. Zuazua)

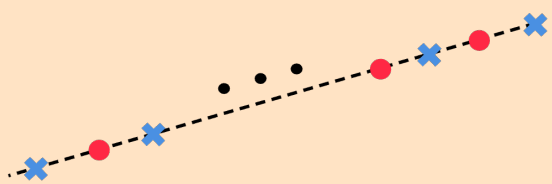
Assume $\mathbf{x_n}, \mathbf{x_n} \sim U\left([0,1]^d\right)$ *for all n. For any* $T > 0$, *there exists* $\theta = (\mathbf{w}, \mathbf{a}, b)$ *where*

$$\mathbf{a} \in \{\mathbf{e_1}, \dots, \mathbf{e_d}\} \subset \mathbb{S}^{d-1} \quad \textit{const}, \qquad (\mathbf{w}, b) \in L^\infty((0,T); \mathbb{R}^d \times \mathbb{R}),$$

that classifies the dataset.
Furthermore, θ *is piecewise constant with L discontinuities, and satisfies:*

$$\mathbb{P}(L=0) = 1 - \left(\frac{2(N!)^2}{(2N)!}\right)^d \sim 1 - \exp\left\{-\frac{\sqrt{\pi N}}{2^{2N-1}} d\right\} \longrightarrow 1 \quad \textit{iff} \quad d \gtrsim 2^N/\sqrt{N}.$$

Worst-case scenario ($k = 2N-2$)
Characterized by $\{\mathbf{x_n}, \mathbf{x_n}\}_{n=1}^N \subset \mathbb{R}^d$ aligned and interspersed by colors.



Very pathological!

Optimality computational test: Setting

Goal

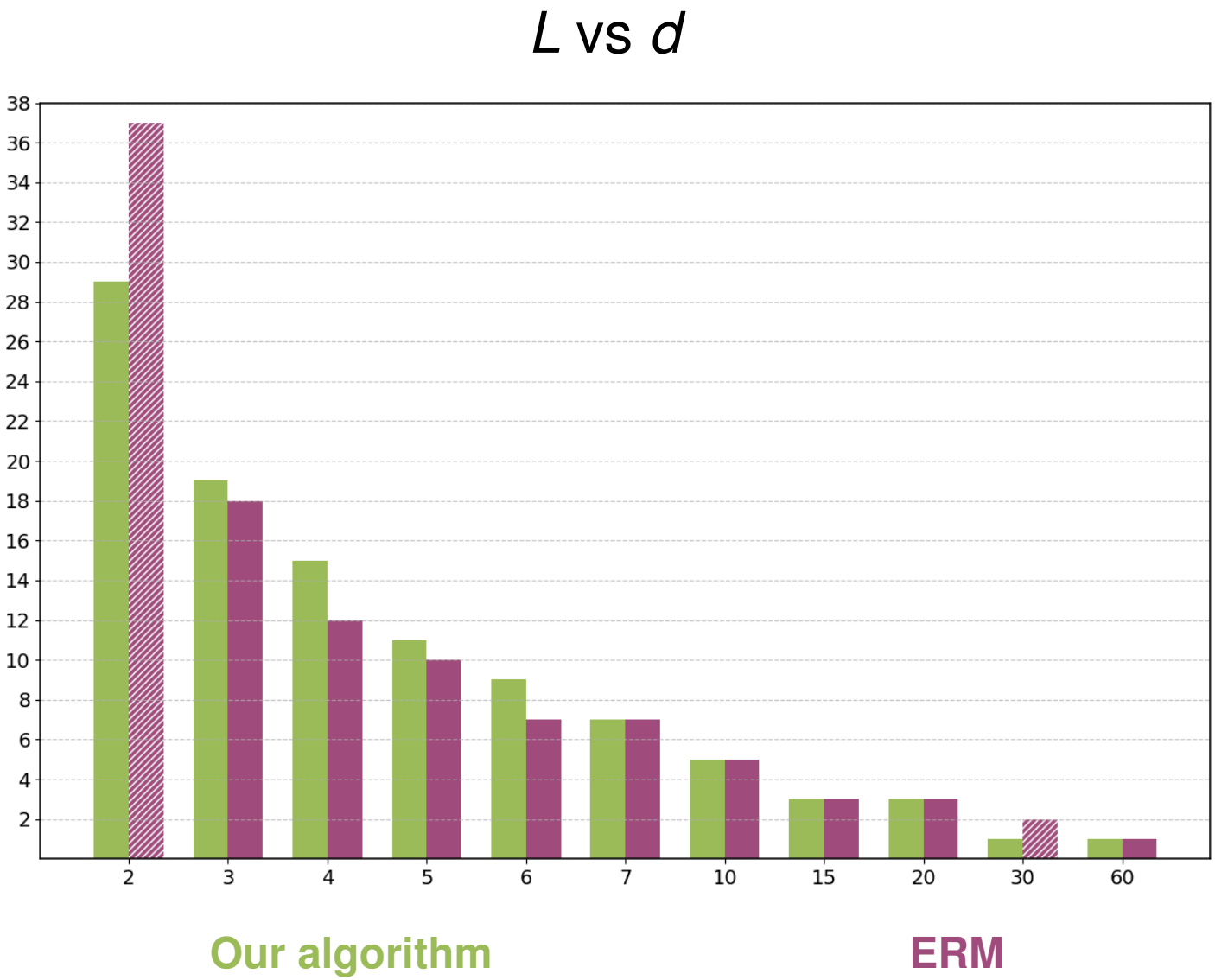
Compare the performance of our algorithm (L_0) against empirical risk minimization (ERM).

- **Data**. 10 datasets sampled from $U([0,1]^d)$ for $N = 30$, and several $2 \leq d \leq 2N$.
- **Model**. $\sigma =$ TrunReLU, so: $\boxed{L_0 = 2\left\lceil \frac{N}{d} \right\rceil \approx \frac{60}{d}.}$
- **Error**. Mean squared loss with margin $h = 0.5$:

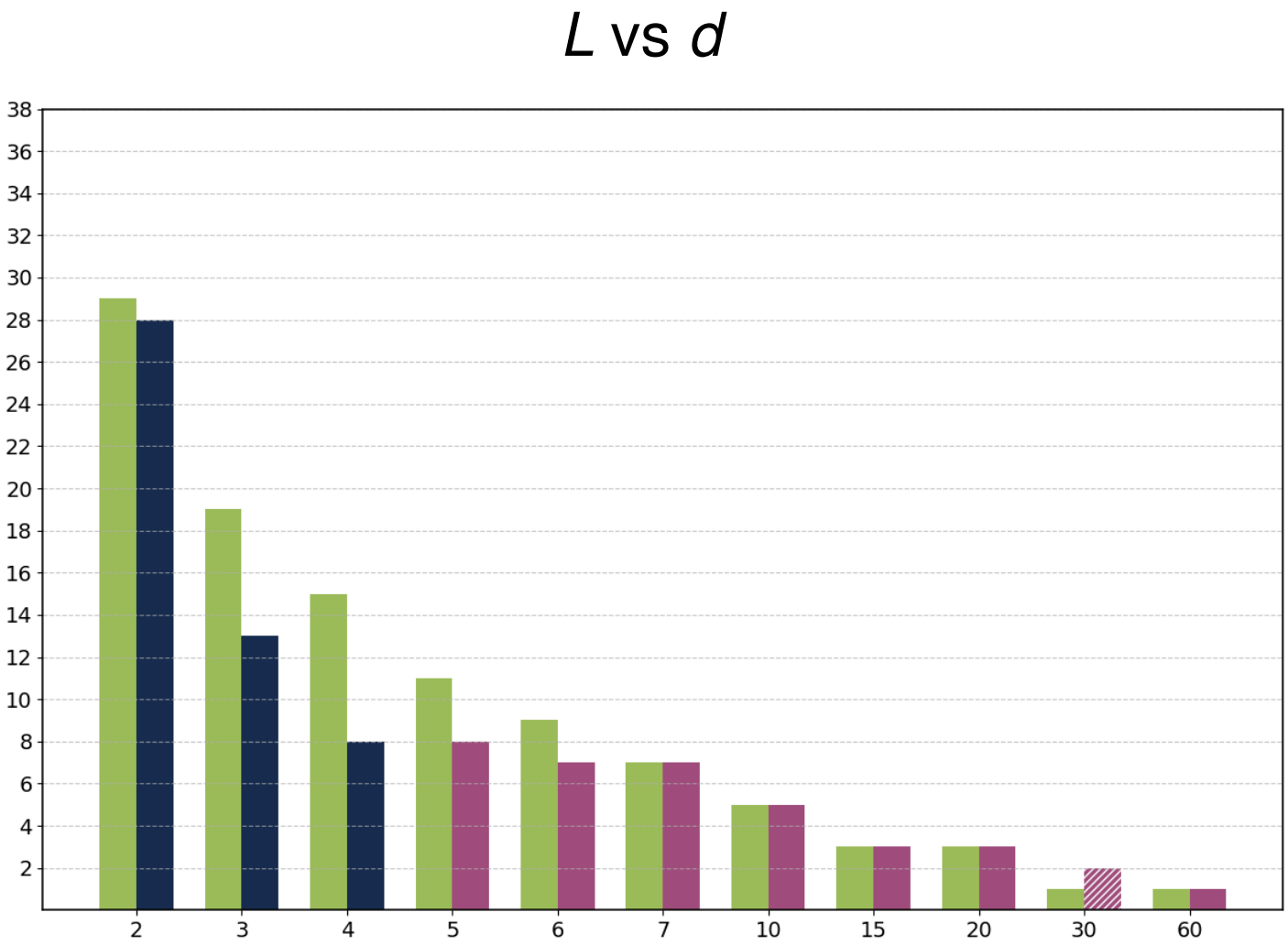
$$\frac{1}{N}\sum_{\mathbf{x}_n \in R} \operatorname{dist}\left(\Phi_T(\mathbf{x}_n), \{x^{(1)} < 0.5 + h\}\right)^2 + \frac{1}{N}\sum_{\mathbf{x}_n \in B} \operatorname{dist}\left(\Phi_T(\mathbf{x}_n), \{x^{(1)} > 0.5 - h\}\right)^2$$

- **Procedure**. Fix d. Verify that $L = L_0$ suffices to classify all datasets.
  Reduce L until any dataset fails to be classified in 20 trials.

Optimality computational test: Results



Optimality computational test: Results



Empirical risk minimization

Click to play in YT

We observe irrelevant movements. Good initialization of parameters is important!

Conclusions

- We have found a reduced value of $L = 4\lceil N/d \rceil$ that guarantees any dataset in $\mathbb{R}^d$ can be classified.
- We exploit dimensionality: Any dataset in general position in $\mathbb{R}^d$ can be divided into clusters of d points using hyperplanes.
- If $d \gtrsim 2^N/\sqrt{N}$ then, with high probability, any two sets of N points can be classified with $L = 0$ discontinuities (autonomous neural ODE).

Open problems

- Is $L = 4\lceil N/d \rceil$ sharp? If so, for which configurations is maximality attained?
- Role played by the chosen boundary decision? Which is the optimal one?
- Potential application for appropriate initialization of parameters.

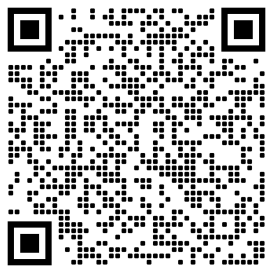

Antonio Álvarez-López, Rafael Orive-Illera, and Enrique Zuazua.
Optimized classification with neural ODEs via separability.
arXiv preprint arXiv:2312.13807, 2023.


Antonio Álvarez-López, Arselane H. Slimane, and Enrique Zuazua.
Interplay between depth and width for interpolation in neural ODEs.
Neural Networks, 180:106640, 2024.


Domènec Ruiz-Balet and Enrique Zuazua.
Neural ODE Control for Classification, Approximation, and Transport.
SIAM Review, 65(3):735--773, 2023.

Paper

Thank you for your attention!



GitHub

Analogy

Further improvements, and finding God's Number [edit]

In 2006, Silviu Radu further improved his methods to prove that every position can be solved in at most 27 face turns or 35 quarter turns.[20] Daniel Kunkle and Gene Cooperman in 2007 used a supercomputer to show that all unsolved cubes can be solved in no more than 26 moves (in face-turn metric). Instead of attempting to solve each of the billions of variations explicitly, the computer was programmed to bring the cube to one of 15,752 states, each of which could be solved within a few extra moves. All were proved solvable in 29 moves, with most solvable in 26. Those that could not initially be solved in 26 moves were then solved explicitly, and shown that they too could be solved in 26 moves.[21][22]

Tomas Rokicki reported in a 2008 computational proof that all unsolved cubes could be solved in 25 moves or fewer.[23] This was later reduced to 23 moves.[24] In August 2008, Rokicki announced that he had a proof for 22 moves.[25]

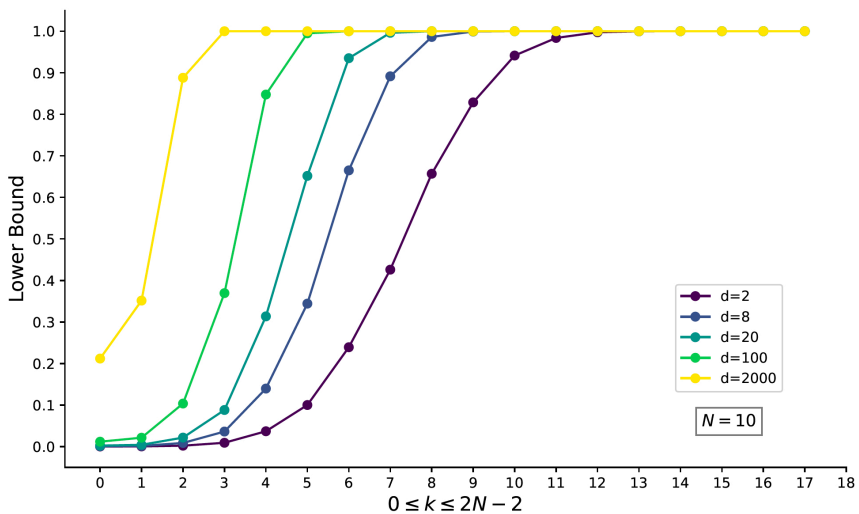
Finally, in 2010, Tomas Rokicki, Herbert Kociemba, Morley Davidson, and John Dethridge gave the final computer-assisted proof that all cube positions could be solved with a maximum of 20 face turns.[2] In 2009, Tomas Rokicki proved that 29 moves in the quarter-turn metric is enough to solve any scrambled cube.[26] And in 2014, Tomas Rokicki and Morley Davidson proved that the maximum number of quarter-turns needed to solve the cube is 26.[3]

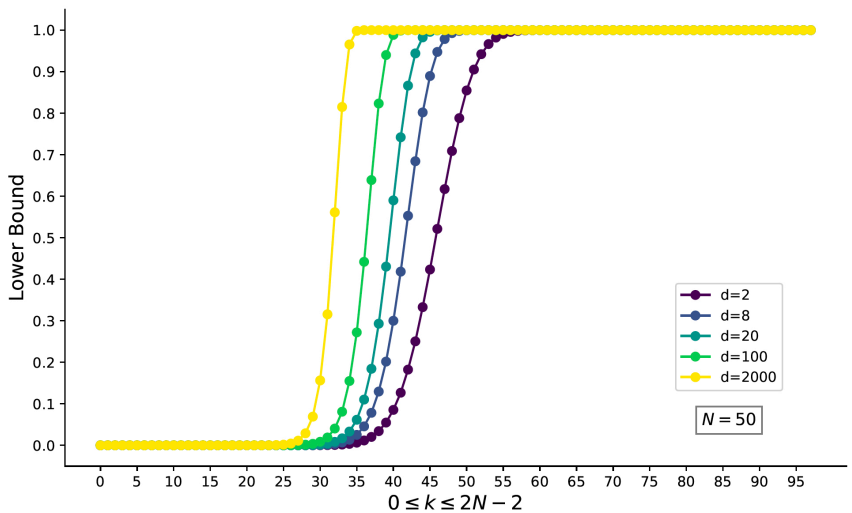
Probabilistic complexity

For $k = 0, \ldots, 2N-2$:

$$\mathbb{P}(L \leq k) = 1 - \left(\sum_{p=\left\lceil \frac{k}{2}+1 \right\rceil}^{N} \binom{N-1}{p-1}^2 + \sum_{p=\left\lceil \frac{k+1}{2} \right\rceil}^{N-1} \binom{N-1}{p} \binom{N-1}{p-1} \right)^d \left(\frac{2(N!)^2}{(2N)!} \right)^d,$$

Representation:



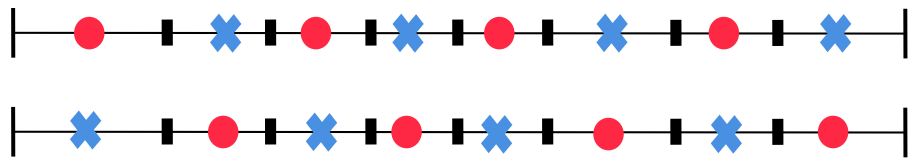


Proof. Step 1: Separability in dimension $d = 1$

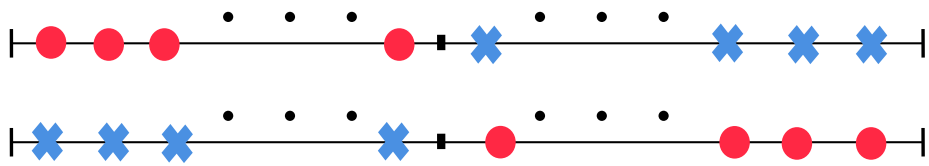
Let

$$Z(R, B) := \text{Minimum number of points needed to separate } (R, B)$$

- $Z = 2N - 1$ if



- $Z = 1$ if

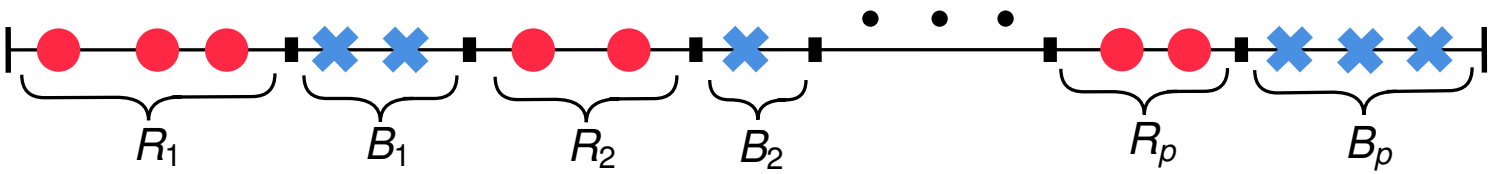


$$\mathbb{P}(Z = 1) = \mathbb{P}(Z = 2N - 1) = \frac{2(N!)^2}{(2N)!}.$$

Step 1: Separability in dimension $d = 1$

Let

$$Z(R, B) := \text{Minimum number of points needed to separate } (R, B)$$



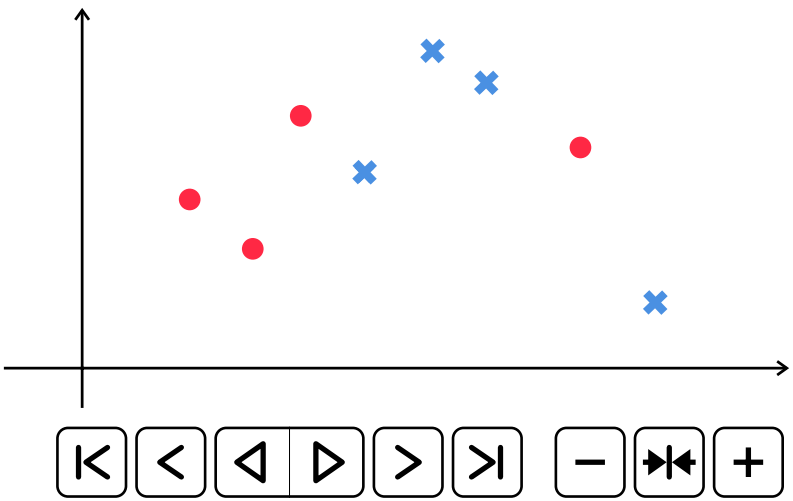
$$\mathbb{P}(Z = k) = \begin{cases} \binom{N-1}{p-1}^2 \frac{2(N!)^2}{(2N)!}, & \text{if} \quad k = 2p-1, \\ \binom{N-1}{p}\binom{N-1}{p-1} \frac{2(N!)^2}{(2N)!}, & \text{if} \quad k = 2p, \end{cases}$$

for $k = 1, \ldots, 2N-1$.

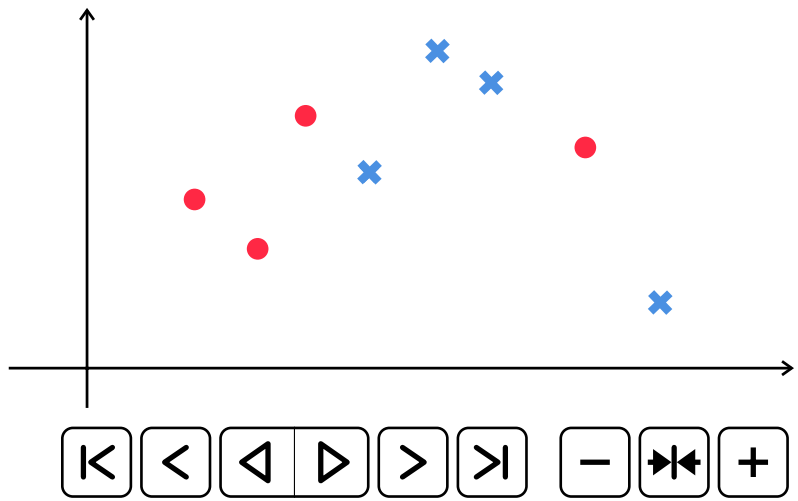
Step 2: Separability in dimension $d > 1$

Let

$$\pi_i := i\text{-th canonical projection in } \mathbb{R}^d, \qquad Z_i(R, B) := Z(\pi_i(R), \pi_i(B))$$



(a) $Z^1(R, B) = 3$



(b) $Z^2(R, B) = 4$

Step 2: Separability in dimension $d > 1$

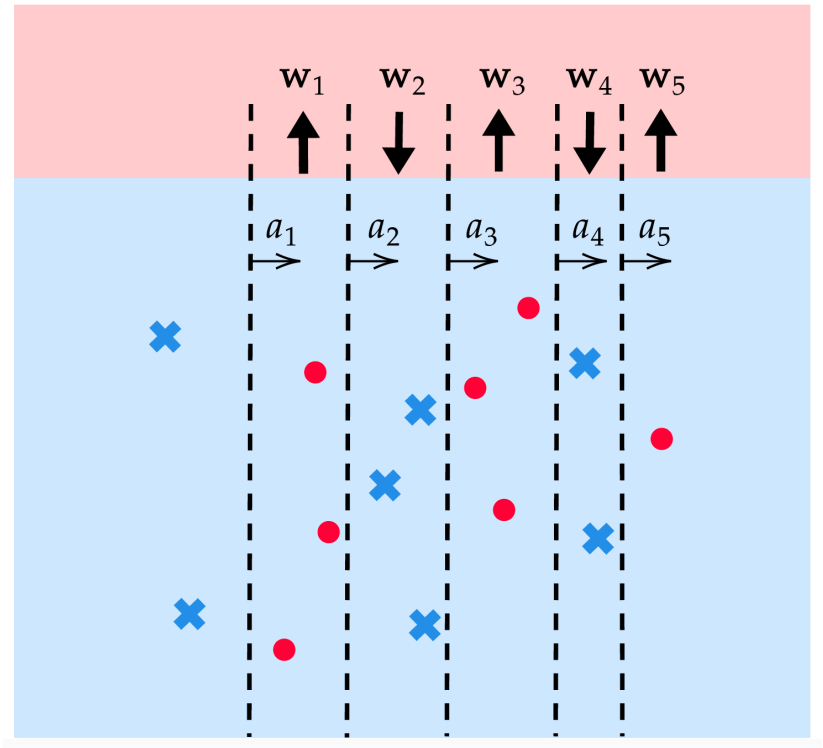
Let

$$\pi_i := i\text{-th canonical projection in } \mathbb{R}^d, \qquad Z_i(R, B) := Z(\pi_i(R), \pi_i(B))$$

Z_i independent and identically distributed to Z for all i, so we can compute:

$$\mathbb{P}\left(\min_{i=1,\dots,d}\{Z_i\} \geq k\right) = \left(\mathbb{P}\left(Z \geq k\right)\right)^d.$$

Step 3: Control



The statement is concluded via $\mathbb{P}(L \leq k) = \mathbb{P}\left(\min_{i=1,\ldots,d} \{Z_i\} \leq k+1\right)$.

Optimality computational test: Complete setting

- **Data**. 10 datasets of $N = 30$ red points and $N = 30$ blue points sampled from $U([0,1]^d)$ for various dimensions d ranging from 2 to $2N$.
- **Model**. Piecewise constant controls over $L+1$ intervals of length $\Delta t = \frac{T}{L+1}$, where $T = 60$. Explicit Euler scheme with step size of $0.25\Delta t$.
- **Complexity**. For each d, we verify that $L = L_0$ discontinuities suffice to classify all datasets. Then we reduce L until any dataset fails to be classified.
- **Error**.

$$\frac{1}{|R|}\sum_{\mathbf{x}_n \in R} \operatorname{dist}\left(\Phi_T(\mathbf{x}_n,\theta), \{x^{(1)} < 1\}\right)^2 + \frac{1}{|B|}\sum_{\mathbf{x}_n \in B} \operatorname{dist}\left(\Phi_T(\mathbf{x}_n,\theta), \{x^{(1)} > 0\}\right)^2.$$

- **Training.** Adam optimizer with a learning rate of 0.01.
- **Stopping criteria**.
  1. The maximum number of 70000 epochs is reached.
  2. Slow convergence, if $\mathcal{L} \geq 0.15$ at 20000 epochs or $\mathcal{L} \geq 0.1$ at 40000 epochs, or if the minimum error does not decrease over 5000 consecutive epochs.
  3. Local minima detection, if the maximum relative error on every 50 consecutive epochs exceeds a threshold of 10^{-20}.