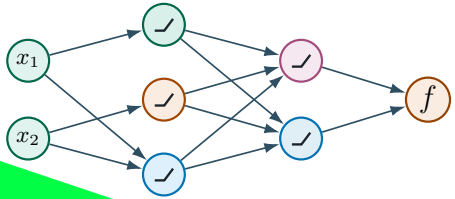
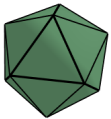
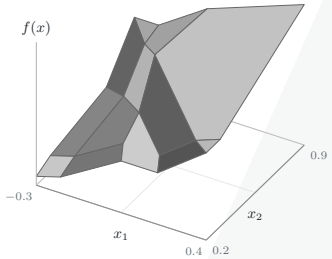
The Polyhedral Geometry of ReLU Neural Networks and Neural Network Verification

Moritz Stargalla
University of Technology Nuremberg

Workshop: Geometric Viewpoints on Learning, Data, and Neural Networks

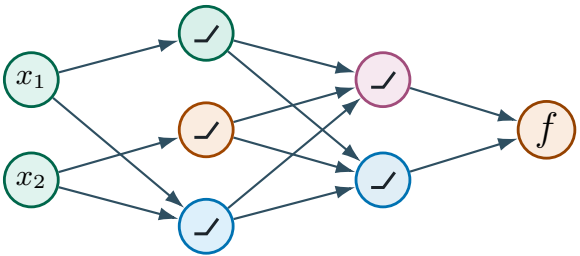






UTN

ReLU neural networks

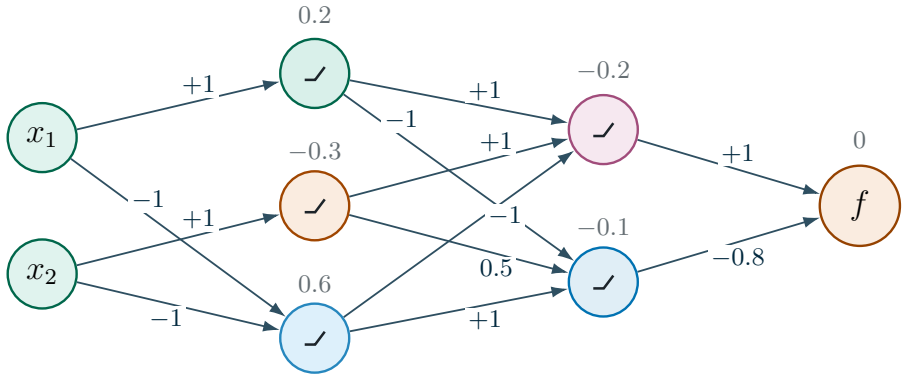


d: input dimension

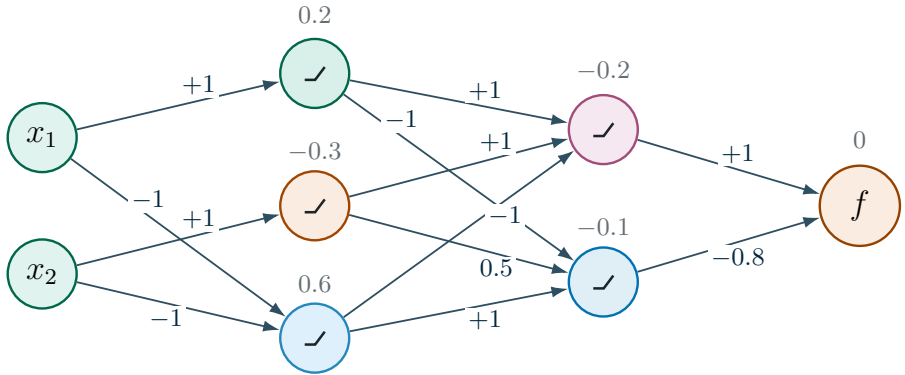
n: width

ℓ: hidden layers

ReLU neural networks

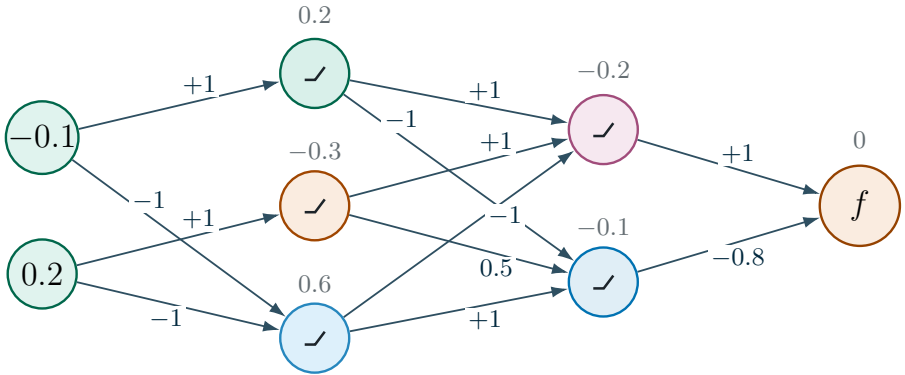


ReLU neural networks



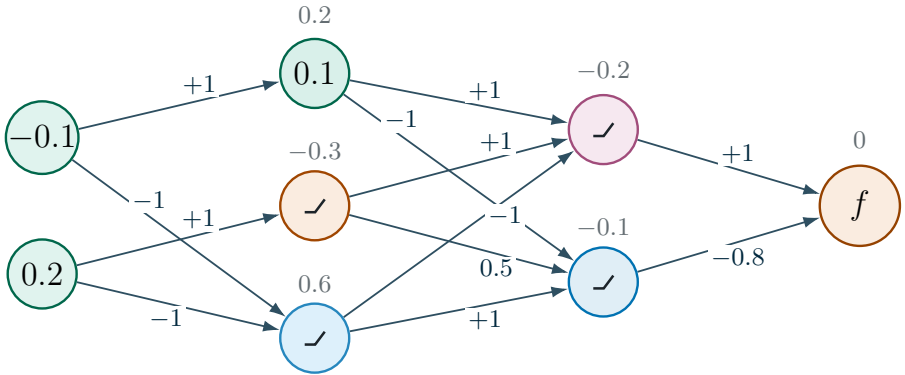
$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

ReLU neural networks



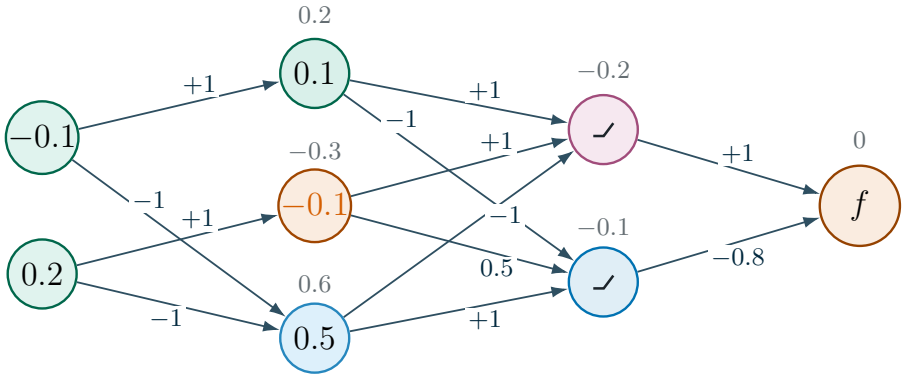
$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

ReLU neural networks



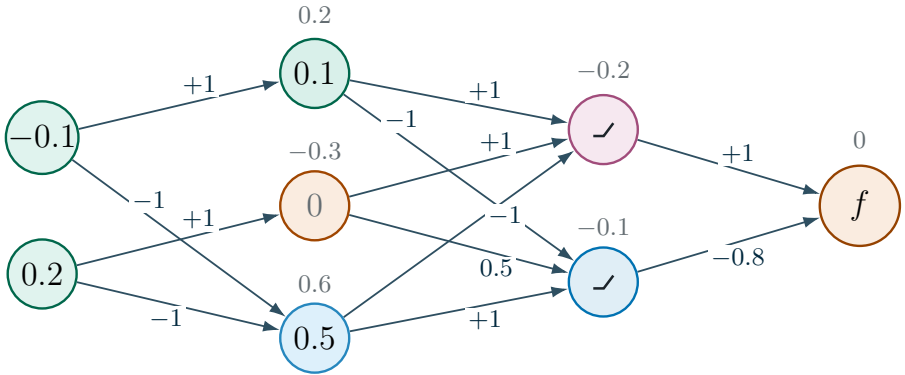
$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

ReLU neural networks



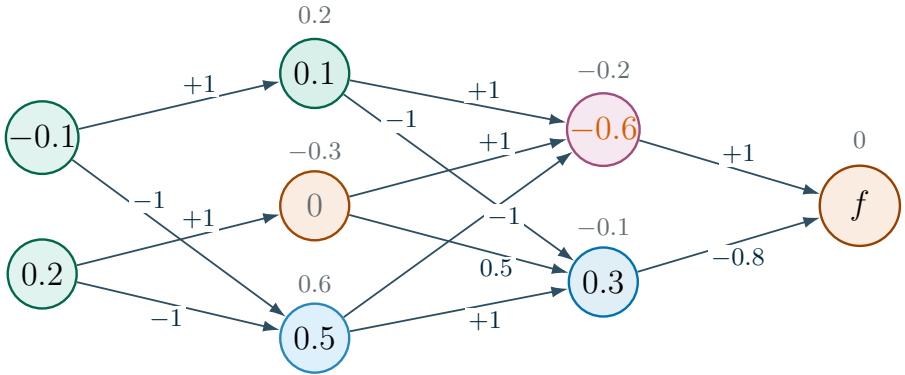
$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

ReLU neural networks



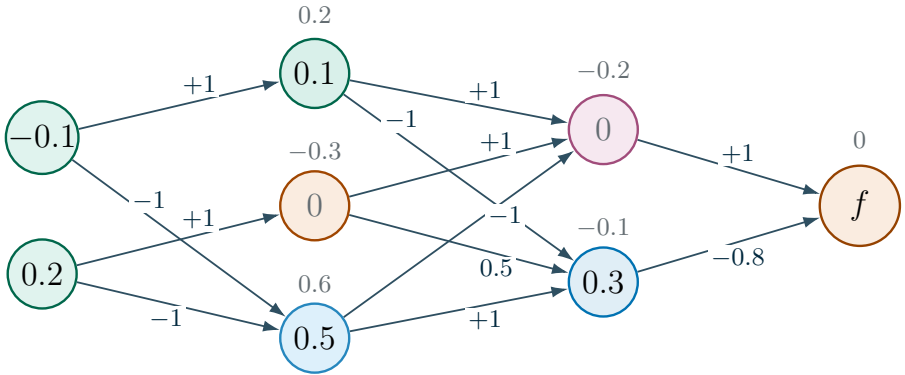
$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

ReLU neural networks



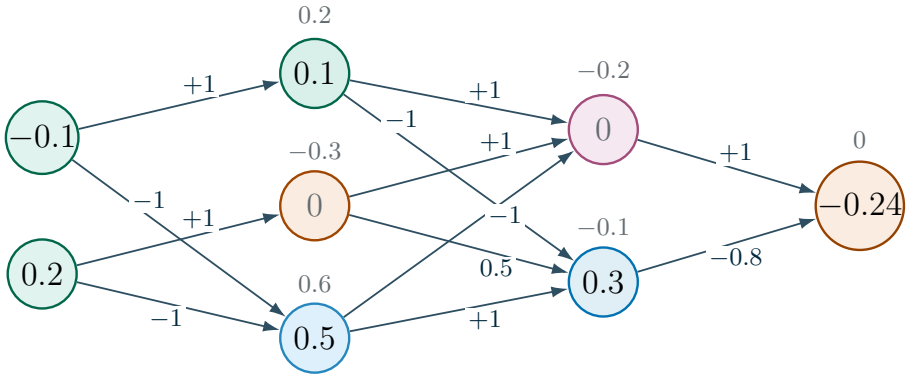
$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

ReLU neural networks



$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

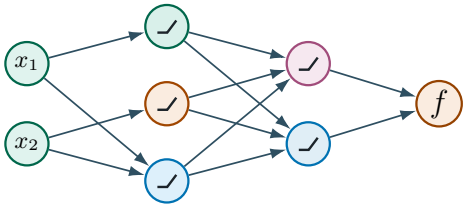
ReLU neural networks



$$x \longmapsto \max\{0, \sum_i w_i x_i + b\}$$

$$f(-0.1, 0.2) = -0.24$$

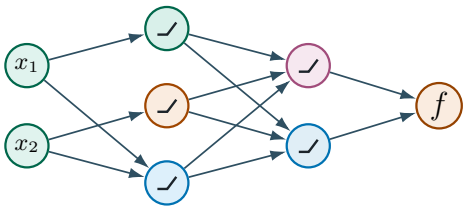
What do ReLU networks compute?



Here: $\ell = 2$

$$f = A^{(\ell+1)} \circ \mathrm{ReLU} \circ \cdots \circ \mathrm{ReLU} \circ A^{(1)}$$

What do ReLU networks compute?

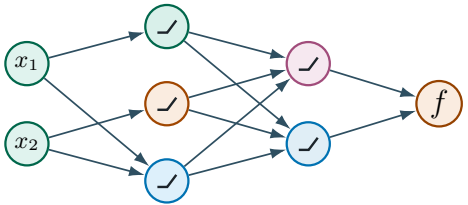


Here: $\ell = 2$

$$f = A^{(\ell+1)} \circ \mathrm{ReLU} \circ \cdots \circ \mathrm{ReLU} \circ A^{(1)}$$

$A^{(i)}$ is affine with $A^{(i)}(x) = W^{(i)}x + b^{(i)}$, $\mathrm{ReLU}(x) = \max\{0, x\}$

What do ReLU networks compute?



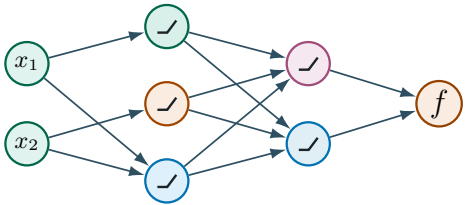
Here: $\ell = 2$

$$f = A^{(\ell+1)} \circ \mathrm{ReLU} \circ \cdots \circ \mathrm{ReLU} \circ A^{(1)}$$

$A^{(i)}$ is affine with $A^{(i)}(x) = W^{(i)}x + b^{(i)}$, $\mathrm{ReLU}(x) = \max\{0, x\}$

$$W^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{pmatrix} \quad b^{(1)} = \begin{pmatrix} 0.2 \\ -0.3 \\ 0.6 \end{pmatrix} \quad W^{(2)} = \begin{pmatrix} 1 & 1 & -1 \\ -1 & 0.5 & 1 \end{pmatrix} \quad b^{(2)} = \begin{pmatrix} -0.2 \\ -0.1 \end{pmatrix} \quad W^{(3)} = \begin{pmatrix} 1 & -0.8 \end{pmatrix} \quad b^{(3)} = 0$$

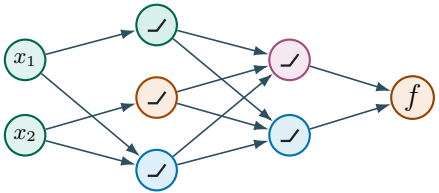
What do ReLU networks compute?



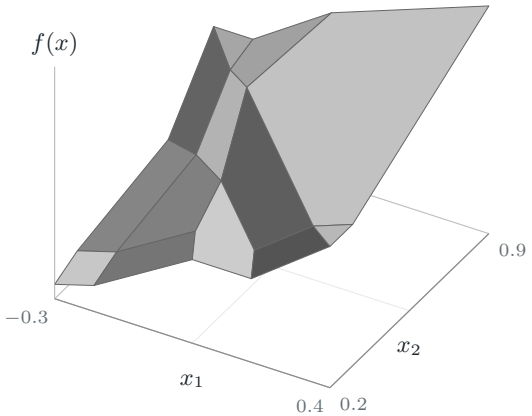
Here: $\ell = 2$

$$f = A^{(\ell+1)} \circ \mathrm{ReLU} \circ \cdots \circ \mathrm{ReLU} \circ A^{(1)}$$

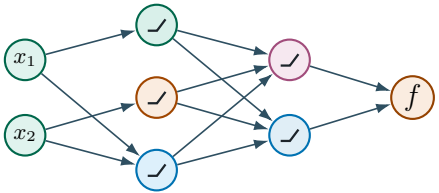
What do ReLU networks compute?



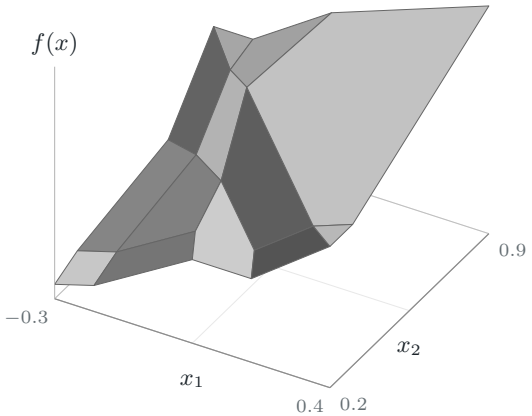
- A ReLU network computes a continuous piecewise linear function.



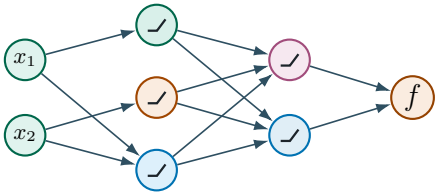
What do ReLU networks compute?



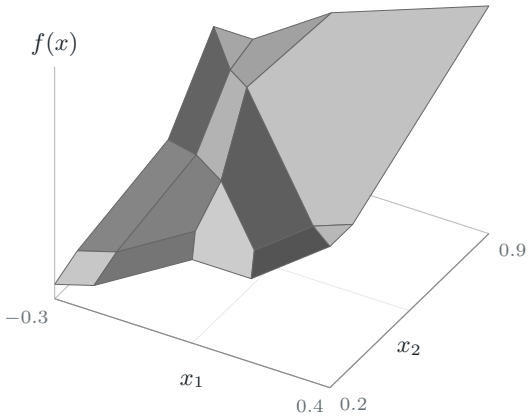
- A ReLU network computes a CPWL function.



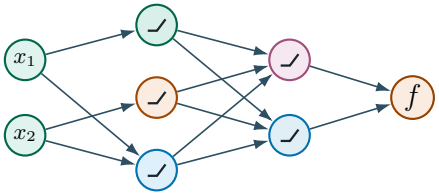
What do ReLU networks compute?



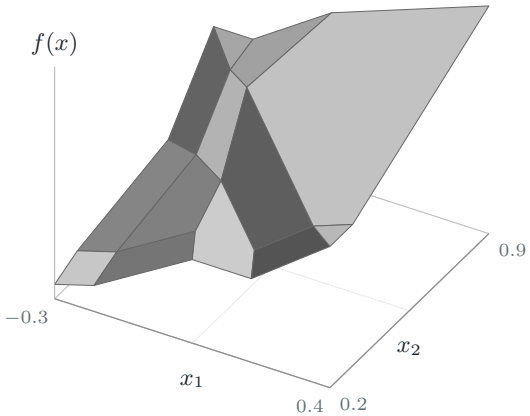
- A ReLU network computes a CPWL function.
- $\mathbb{R}^d$ is partitioned into polyhedral linear regions.



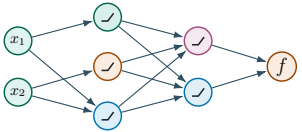
What do ReLU networks compute?



- A ReLU network computes a CPWL function.
- $\mathbb{R}^d$ is partitioned into polyhedral linear regions.
- Every CPWL function can be computed by a ReLU network.



Topic of this talk



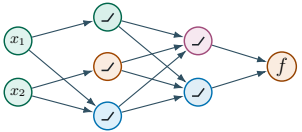
Expressivity

Which functions can a given ReLU network architecture compute?

Verification

How efficiently can we verify properties of a given ReLU network?

Topic of this talk



Expressivity

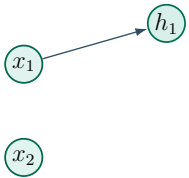
Which functions can a given ReLU network architecture compute?

Verification

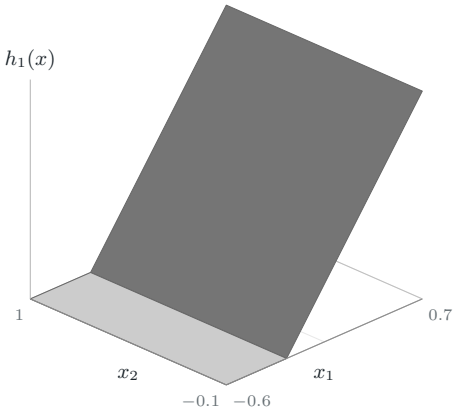
How efficiently can we verify properties of a given ReLU network?

- Linear regions

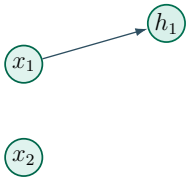
First layer: Hyperplanes



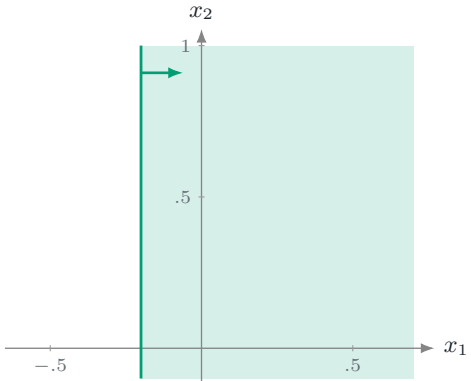
$$h_1(x) = \mathrm{ReLU}(x_1 + 0.2)$$



First layer: Hyperplanes

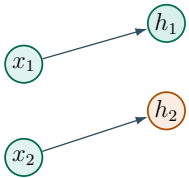


$$h_1(x) = \mathrm{ReLU}(x_1 + 0.2)$$

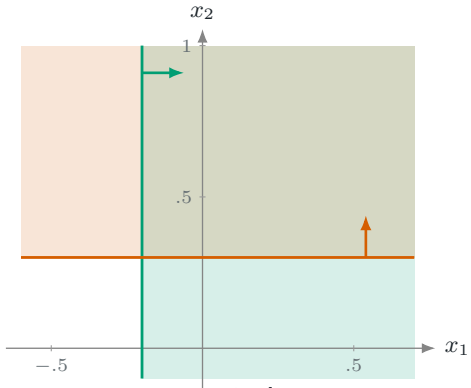


neurons induce
oriented hyperplanes

First layer: Hyperplanes

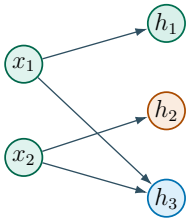


$$h_1(x) = \mathrm{ReLU}(x_1 + 0.2)$$
$$h_2(x) = \mathrm{ReLU}(x_2 - 0.3)$$

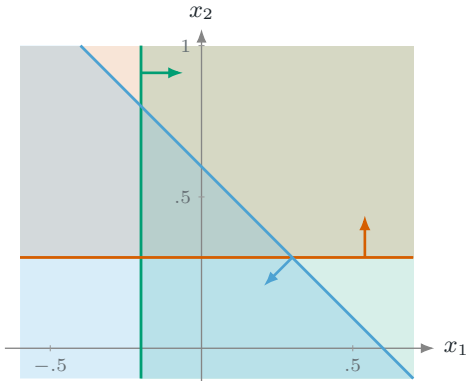


neurons induce
oriented hyperplanes

First layer: Hyperplanes

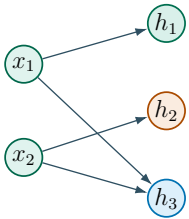


$$h_1(x) = \text{ReLU}(x_1 + 0.2)$$
$$h_2(x) = \text{ReLU}(x_2 - 0.3)$$
$$h_3(x) = \text{ReLU}(-x_1 - x_2 + 0.6)$$

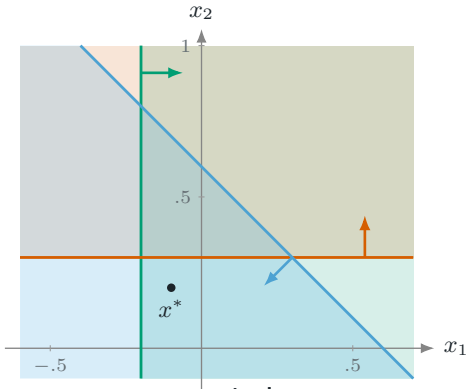


neurons induce
oriented hyperplanes

First layer: Hyperplanes

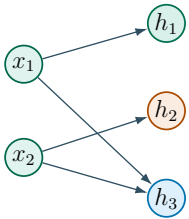


$$h_1(x) = \mathrm{ReLU}(x_1 + 0.2)$$
$$h_2(x) = \mathrm{ReLU}(x_2 - 0.3)$$
$$h_3(x) = \mathrm{ReLU}(-x_1 - x_2 + 0.6)$$

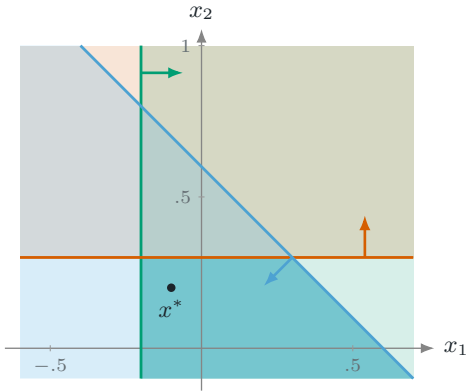


neurons induce
oriented hyperplanes

First layer: Hyperplanes

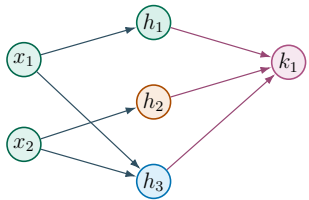


$$h_1(x) = x_1 + 0.2$$
$$h_2(x) = 0$$
$$h_3(x) = -x_1 - x_2 + 0.6$$



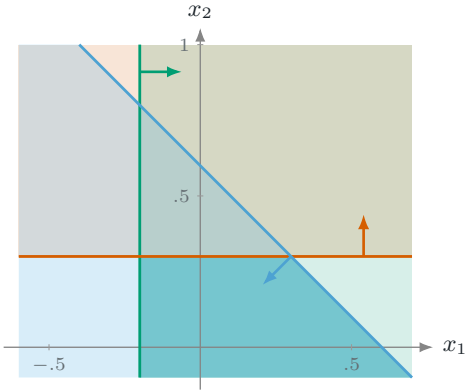
In each cell, every first-layer neuron computes an affine function.

Second layer: Bent Hyperplanes

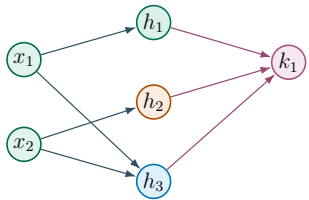


$$k_1(x) =$$

$$\mathrm{ReLU}\big(\mathrm{ReLU}(x_1 + 0.2) + \mathrm{ReLU}(x_2 - 0.3)$$
$$- \mathrm{ReLU}(-x_1 - x_2 + 0.6) - 0.2\big).$$

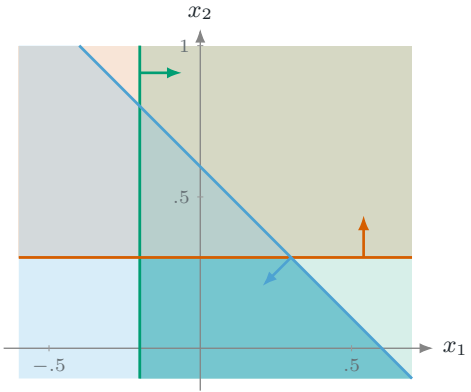


Second layer: Bent Hyperplanes



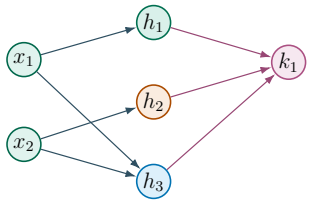
On the highlighted cell:

$$k_1(x) = \mathrm{ReLU}(2x_1 + x_2 - 0.6).$$



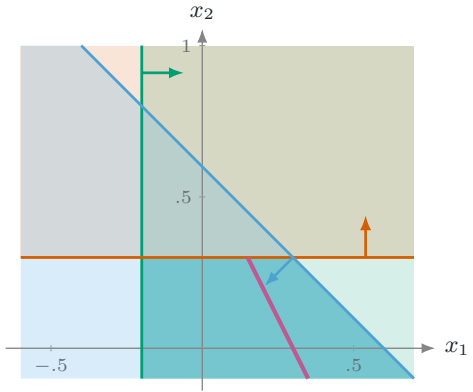
k_1 introduces a nonlinearity in this cell

Second layer: Bent Hyperplanes



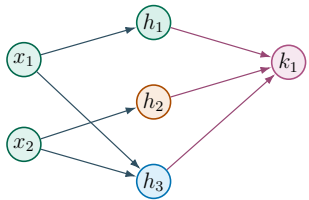
On the highlighted cell:

$$k_1(x) = \mathrm{ReLU}(2x_1 + x_2 - 0.6).$$



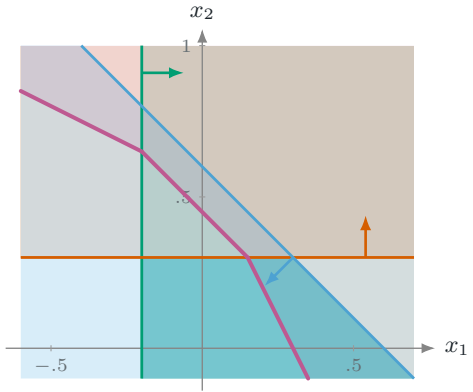
k_1 introduces a nonlinearity in this cell

Second layer: Bent Hyperplanes



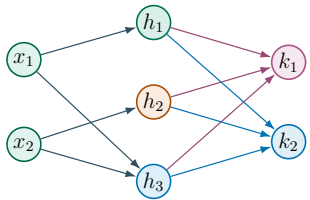
On the highlighted cell:

$$k_1(x) = \mathrm{ReLU}(2x_1 + x_2 - 0.6).$$



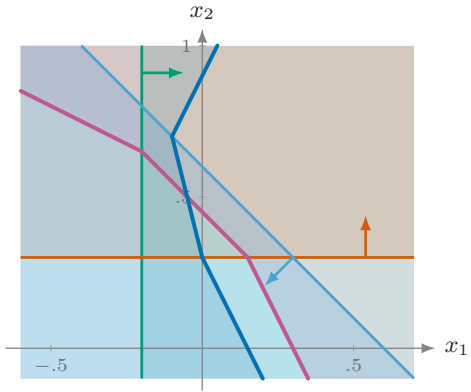
k_1 introduces a nonlinearity in this cell

Second layer: Bent Hyperplanes

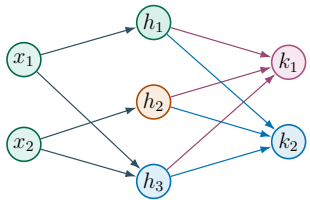


$$k_1 = \mathrm{ReLU}(h_1 + h_2 - h_3 - 0.2)$$

$$k_2 = \mathrm{ReLU}(-h_1 + 0.5h_2 + h_3 - 0.1)$$

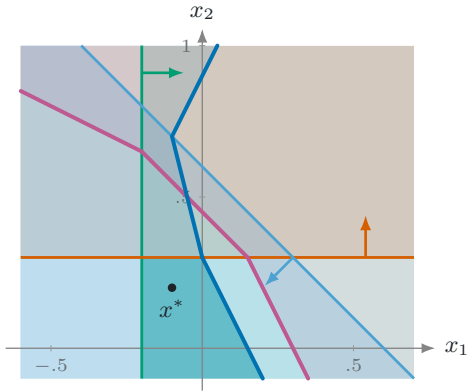


Second layer: Bent Hyperplanes



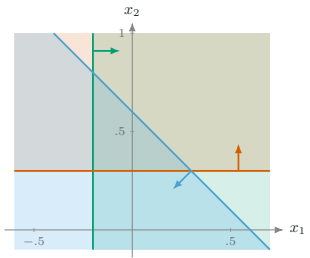
$$k_1 = \mathrm{ReLU}(h_1 + h_2 - h_3 - 0.2)$$

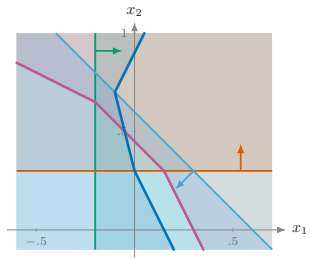
$$k_2 = \mathrm{ReLU}(-h_1 + 0.5h_2 + h_3 - 0.1)$$



On this cell: $f(x) = 1.6x_1 + 0.8x_2 - 0.24$.

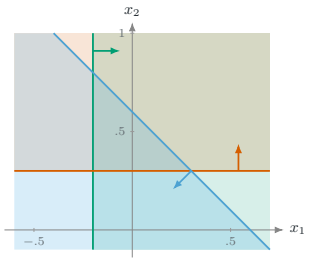
Linear regions as a measure of expressivity

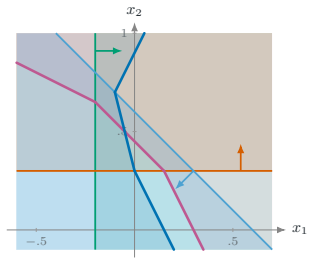




Measure of expressive power:
How many linear regions can a given ReLU network architecture produce?

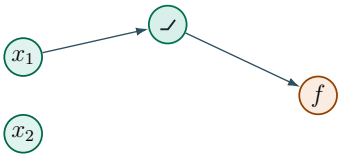
Linear regions as a measure of expressivity

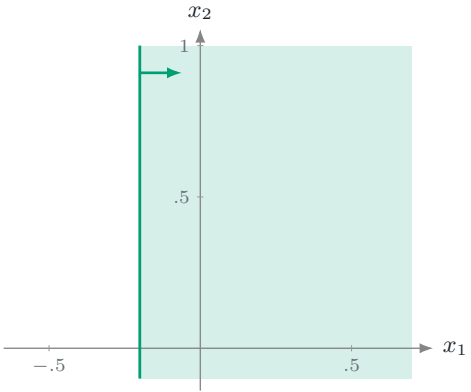




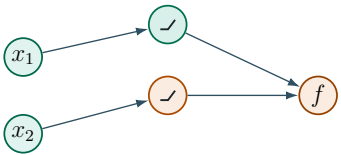
Measure of expressive power:
How many linear regions can a given ReLU network architecture produce?

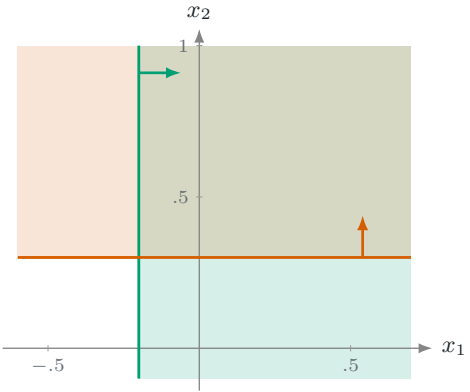
Bounding the number of linear regions: One hidden layer



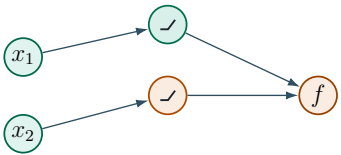


Bounding the number of linear regions: One hidden layer

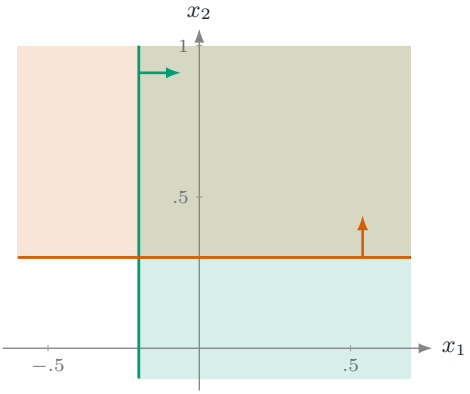




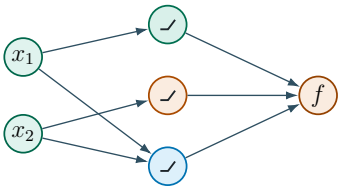
Bounding the number of linear regions: One hidden layer



$$\#\text{cells} \leq 2^n$$

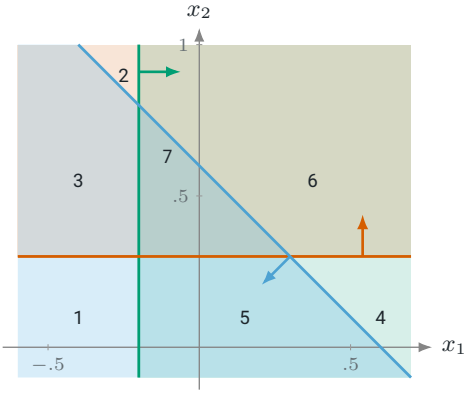


Bounding the number of linear regions: One hidden layer

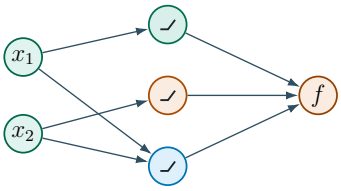


$$\#\text{cells} \leq 2^n$$

3 hyperplanes, 7 cells

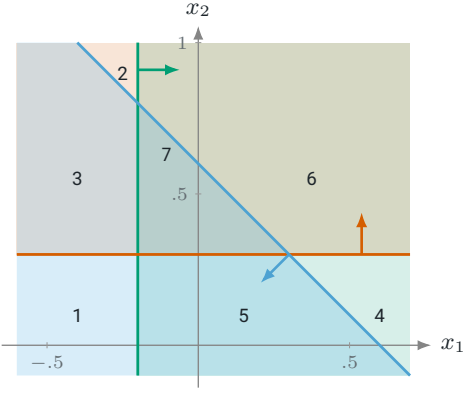


Bounding the number of linear regions: One hidden layer



$$\#\text{cells} \leq 2^n$$

3 hyperplanes, 7 cells

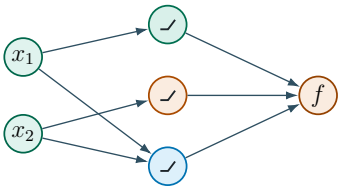


Zaslavsky's theorem

$$\#\text{cells} \leq \sum_{j=0}^{d} \binom{n}{j}$$

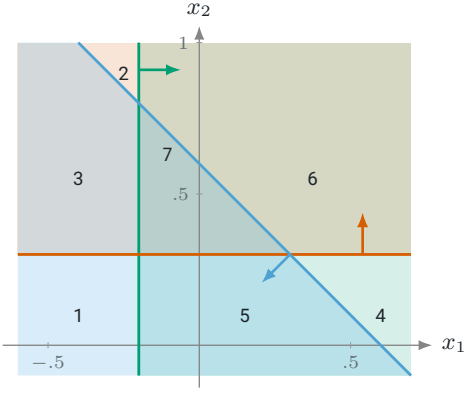
Zaslavsky (1975). *Facing up to Arrangements: Face-Count Formulas for Partitions of Space by Hyperplanes.*

Bounding the number of linear regions: One hidden layer



$$\#\text{cells} \leq 2^n$$

3 hyperplanes, 7 cells



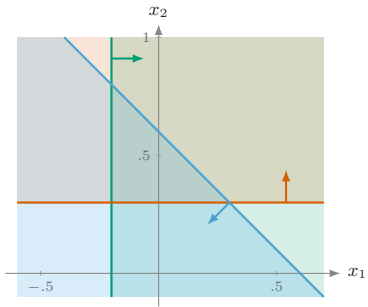
Zaslavsky's theorem

$$\#\text{cells} \leq \sum_{j=0}^{d} \binom{n}{j} \in O(n^d)$$

Zaslavsky (1975). *Facing up to Arrangements: Face-Count Formulas for Partitions of Space by Hyperplanes.*

Bounding the number of linear regions

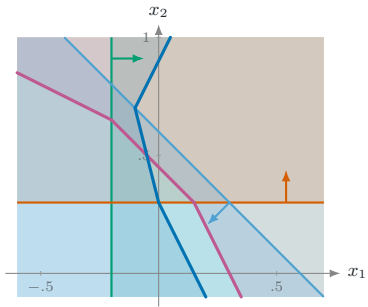
widths $n_i \leq n$ fixed input dimension d ℓ hidden layers (fixed)



1 layer $O(n^d)$

Bounding the number of linear regions

widths $n_i \leq n$ fixed input dimension d ℓ hidden layers (fixed)

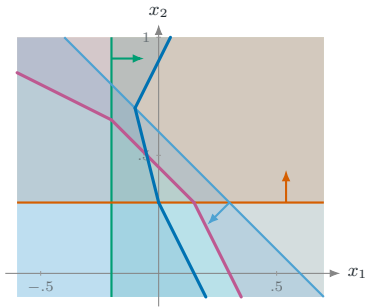


1 layer $O(n^d)$

2 layers $O(n^{2d})$

Bounding the number of linear regions

widths $n_i \leq n$ fixed input dimension d ℓ hidden layers (fixed)

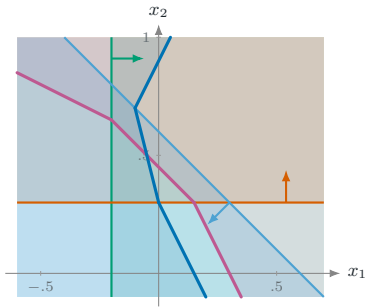


1 layer	$O(n^d)$
2 layers	$O(n^{2d})$
ℓ layers	$O(n^{\ell d})$

Raghu, Poole, Kleinberg, Ganguli and Dickstein (2017). *On the Expressive Power of Deep Neural Networks*.

Bounding the number of linear regions

widths $n_i \leq n$ fixed input dimension d ℓ hidden layers (fixed)

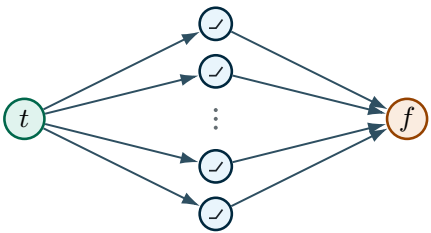


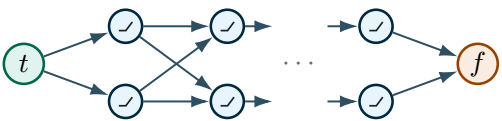
1 layer	$O(n^d)$
2 layers	$O(n^{2d})$
ℓ layers	$O(n^{\ell d})$

$$\#\text{regions} \leq \prod_{i=1}^{\ell} \left(\sum_{j=0}^{d} \binom{n_i}{j} \right)$$

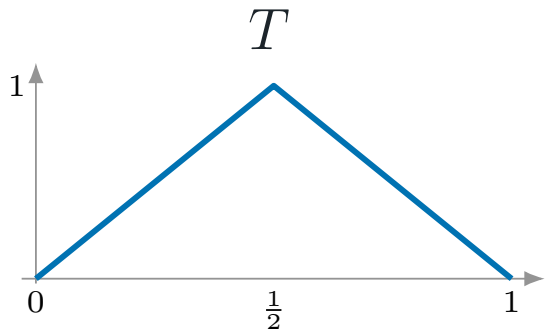
Raghu, Poole, Kleinberg, Ganguli and Dickstein (2017). *On the Expressive Power of Deep Neural Networks*.

Are deeper architectures more expressive than shallow ones?





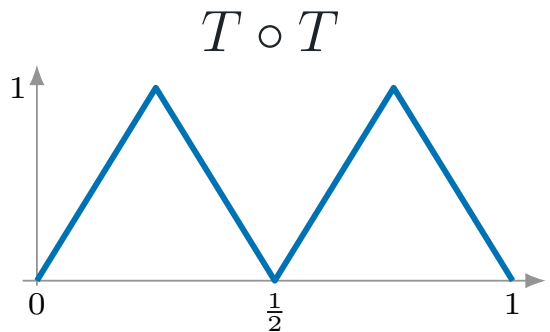
Deep vs shallow networks



$$T(t) = 2\,\mathrm{ReLU}(t) - 4\,\mathrm{ReLU}(t - \tfrac{1}{2}), \qquad t \in [0, 1]$$

Telgarsky (2016). *Benefits of Depth in Neural Networks*.

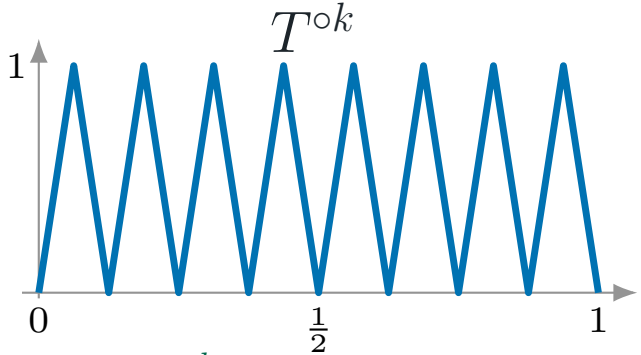
Deep vs shallow networks



$$T(t) = 2\,\mathrm{ReLU}(t) - 4\,\mathrm{ReLU}(t - \tfrac{1}{2}), \qquad t \in [0, 1]$$

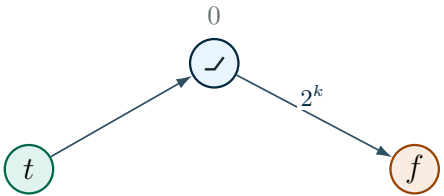
Telgarsky (2016). *Benefits of Depth in Neural Networks*.

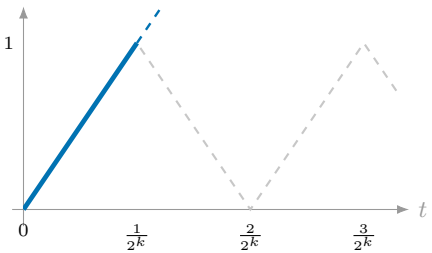
Deep vs shallow networks



Telgarsky (2016). *Benefits of Depth in Neural Networks*.

Deep vs shallow networks

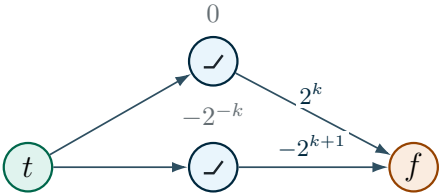


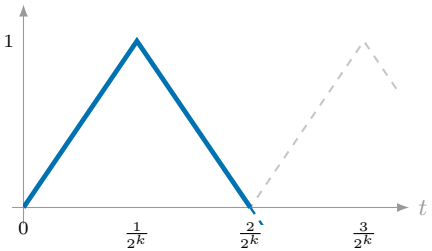


First three segments of $T^{\circ k}$

Telgarsky (2016). *Benefits of Depth in Neural Networks*.

Deep vs shallow networks

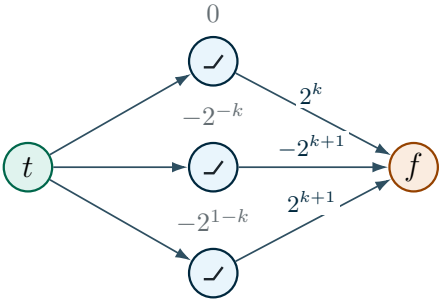


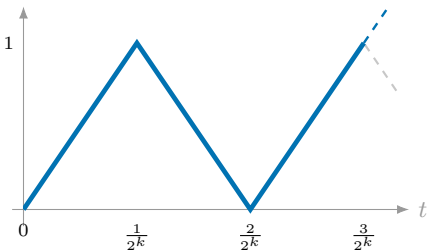


First three segments of $T^{\circ k}$

Telgarsky (2016). *Benefits of Depth in Neural Networks*.

Deep vs shallow networks



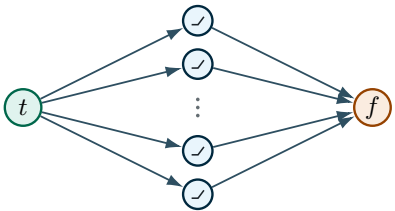


First three segments of $T^{\circ k}$

Telgarsky (2016). *Benefits of Depth in Neural Networks*.

Deep vs shallow networks

one hidden layer

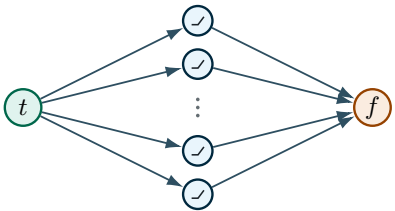


at least 2^k neurons

Telgarsky (2016). *Benefits of Depth in Neural Networks*.

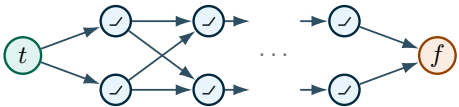
Deep vs shallow networks

one hidden layer



at least 2^k neurons

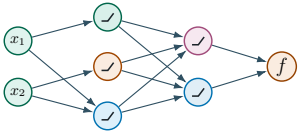
k hidden layers, width 2



$2k$ neurons

Telgarsky (2016). *Benefits of Depth in Neural Networks*.

Topic of this talk



Expressivity

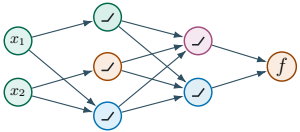
Which functions can a given ReLU network architecture compute?

Verification

How efficiently can we verify properties of a given ReLU network?

- Number of linear regions
- Depth vs width tradeoff

Topic of this talk



Expressivity

How many hidden layers are required to compute a given CPWL function $f : \mathbb{R}^d \to \mathbb{R}$?

Verification

How efficiently can we verify properties of a given ReLU network?

- Number of linear regions
- Depth vs width tradeoff

How many hidden layers are required?

- Wang and Sun showed for any CPWL function $f : \mathbb{R}^d \to \mathbb{R}$

$$f(x) = \sum_{i=1}^{m} \sigma_i \max_{d+1}\left(a_1^{(i)}(x), \ldots, a_{d+1}^{(i)}(x)\right)$$

for some affine linear functions $a_j^{(i)}$ and coefficients $\sigma_i \in \{\pm 1\}$.

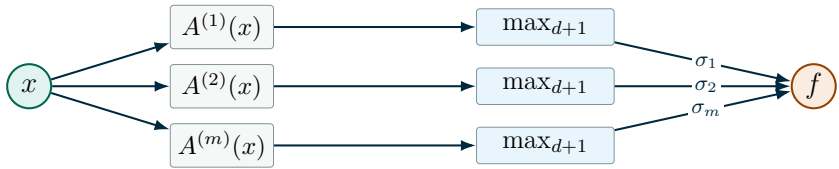
Wang and Sun (2005). *Generalization of Hinging Hyperplanes*.

How many hidden layers are required?

- Wang and Sun showed for any CPWL function $f : \mathbb{R}^d \to \mathbb{R}$

$$f(x) = \sum_{i=1}^{m} \sigma_i \max_{d+1}\left(a_1^{(i)}(x), \ldots, a_{d+1}^{(i)}(x)\right)$$

for some affine linear functions $a_j^{(i)}$ and coefficients $\sigma_i \in \{\pm 1\}$.

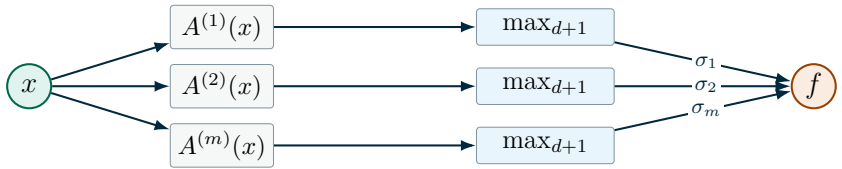


How many hidden layers are required?

- Wang and Sun showed for any CPWL function $f : \mathbb{R}^d \to \mathbb{R}$

$$f(x) = \sum_{i=1}^{m} \sigma_i \max_{d+1}\left(a_1^{(i)}(x), \ldots, a_{d+1}^{(i)}(x)\right)$$

for some affine linear functions $a_j^{(i)}$ and coefficients $\sigma_i \in \{\pm 1\}$.



$$\text{depth}(\max_{d+1}) \leq \ell \quad \Longrightarrow \quad \text{depth}(f) \leq \ell$$

Wang and Sun (2005). *Generalization of Hinging Hyperplanes*.

How many hidden layers are required for $\max_n$?

$$\max_n(x) = \max\{x_1, \ldots, x_n\}$$

Arora, Basu, Mianjy and Mukherjee (2018). *Understanding Deep Neural Networks with Rectified Linear Units*.
Hertrich, Basu, Di Summa and Skutella (2021). *Towards Lower Bounds on the Depth of ReLU Neural Networks*.

How many hidden layers are required for $\max_n$?

$$\max_n(x) = \max\{x_1, \ldots, x_n\}$$

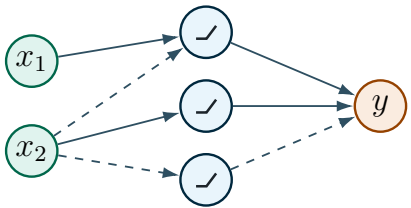
$$\max\{x_1, x_2\} = \max\{0, x_1 - x_2\} + x_2$$

Arora, Basu, Mianjy and Mukherjee (2018). *Understanding Deep Neural Networks with Rectified Linear Units*.
Hertrich, Basu, Di Summa and Skutella (2021). *Towards Lower Bounds on the Depth of ReLU Neural Networks*.

How many hidden layers are required for $\max_n$?

$$\max_n(x) = \max\{x_1, \ldots, x_n\}$$

$$\max\{x_1, x_2\} = \max\{0, x_1 - x_2\} + \max\{0, x_2\} - \max\{0, -x_2\}$$

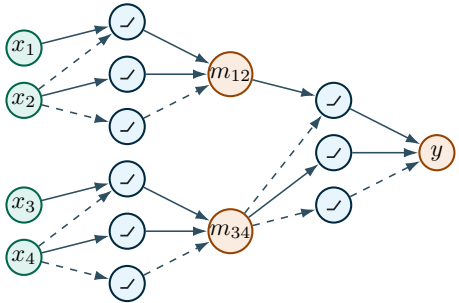


Arora, Basu, Mianjy and Mukherjee (2018). *Understanding Deep Neural Networks with Rectified Linear Units*.
Hertrich, Basu, Di Summa and Skutella (2021). *Towards Lower Bounds on the Depth of ReLU Neural Networks*.

How many hidden layers are required for $\max_n$?

$$\max_n(x) = \max\{x_1, \ldots, x_n\}$$

$$\max\{x_1, x_2\} = \max\{0, x_1 - x_2\} + \max\{0, x_2\} - \max\{0, -x_2\}$$

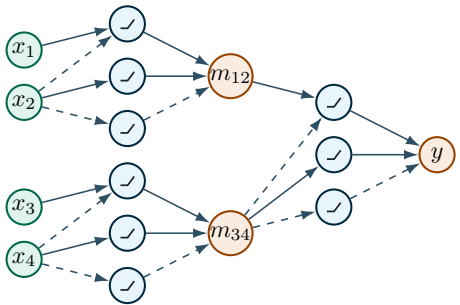


Arora, Basu, Mianjy and Mukherjee (2018). *Understanding Deep Neural Networks with Rectified Linear Units*.
Hertrich, Basu, Di Summa and Skutella (2021). *Towards Lower Bounds on the Depth of ReLU Neural Networks*.

How many hidden layers are required for $\max_n$?

$$\max_n(x) = \max\{x_1, \ldots, x_n\}$$

$$\max\{x_1, x_2\} = \max\{0, x_1 - x_2\} + \max\{0, x_2\} - \max\{0, -x_2\}$$



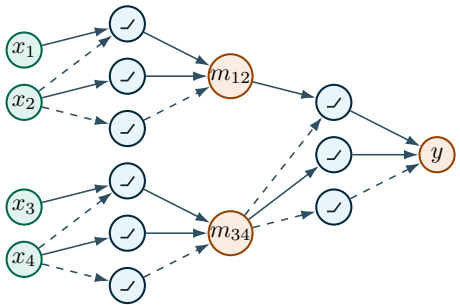
$$\text{depth}(\max_n) \leq \lceil \log_2 n \rceil$$

Arora, Basu, Mianjy and Mukherjee (2018). *Understanding Deep Neural Networks with Rectified Linear Units*.
Hertrich, Basu, Di Summa and Skutella (2021). *Towards Lower Bounds on the Depth of ReLU Neural Networks*.

How many hidden layers are required for $\max_n$?

$$\max_n(x) = \max\{x_1, \ldots, x_n\}$$

$$\max\{x_1, x_2\} = \max\{0, x_1 - x_2\} + \max\{0, x_2\} - \max\{0, -x_2\}$$



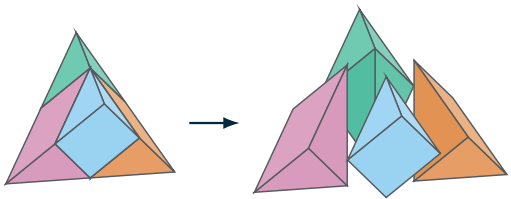
$$\operatorname{depth}(\max_n) \leq \lceil \log_2 n \rceil$$

Former conjecture

The $\log_2$ bound is optimal.

Arora, Basu, Mianjy and Mukherjee (2018). *Understanding Deep Neural Networks with Rectified Linear Units*.
Hertrich, Basu, Di Summa and Skutella (2021). *Towards Lower Bounds on the Depth of ReLU Neural Networks*.

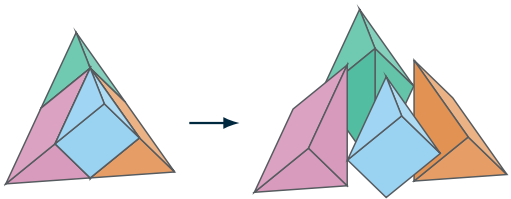
How many hidden layers are required for $\max_n$?



Subdivision of the 3-simplex
into simpler pieces

Bakaev, Brunck, Hertrich, Stade and Yehudayoff (2025). *Better Neural Network Expressivity: Subdividing the Simplex*.

How many hidden layers are required for $\max_n$?

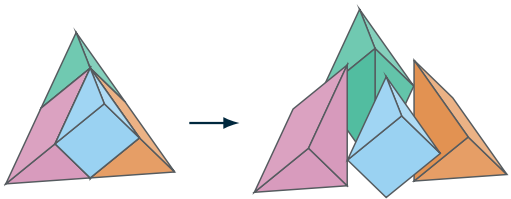


Subdivision of the 3-simplex
into simpler pieces

$\max_5$ in two hidden layers

Bakaev, Brunck, Hertrich, Stade and Yehudayoff (2025). *Better Neural Network Expressivity: Subdividing the Simplex*.

How many hidden layers are required for $\max_n$?



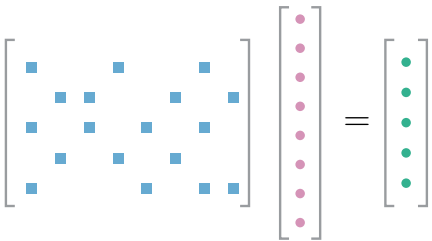
Subdivision of the 3-simplex into simpler pieces

$\max_5$ in two hidden layers

$$\operatorname{depth}(\max_n) \leq \lceil \log_3(n-2) \rceil + 1$$

Bakaev, Brunck, Hertrich, Stade and Yehudayoff (2025). *Better Neural Network Expressivity: Subdividing the Simplex*.

How many hidden layers are required for $\max_n$?

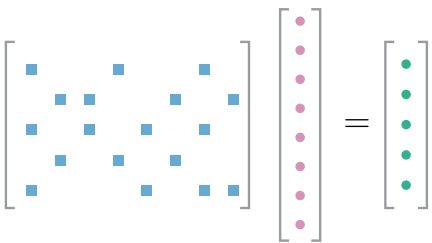


Recently:
$\max_{12}$ in two hidden layers

$$\text{depth}(\max_n) \leq \lceil \log_6(n/2) \rceil + 1$$

Rueß, Averkov, Brunck, Grillo, Hertrich, Loho, Stade, Stargalla, Sun and Winter (2026). *Shallower ReLU Network Representations via Exact Linear Algebra*.

How many hidden layers are required for $\max_n$?



Recently:
$\max_{12}$ in two hidden layers

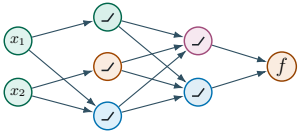
$$\operatorname{depth}(\max_n) \leq \lceil \log_6(n/2) \rceil + 1$$

New Conjecture

$\max_n$ can be computed with two hidden layers for all n.
$\implies$ Two hidden layers suffice for *any* CPWL function.

Rueß, Averkov, Brunck, Grillo, Hertrich, Loho, Stade, Stargalla, Sun and Winter (2026). *Shallower ReLU Network Representations via Exact Linear Algebra*.

Topic of this talk



Expressivity

Which functions can a given ReLU network architecture compute?

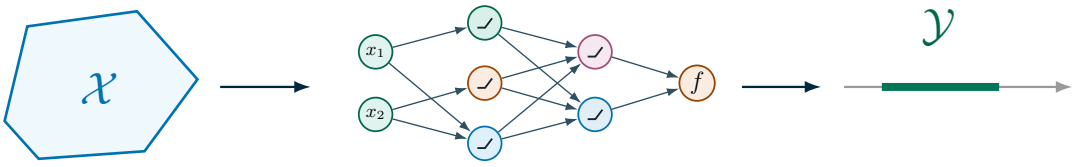
Verification

How efficiently can we verify properties of a given ReLU network?

- Number of linear regions
- Depth vs width tradeoff
- Required depth

Neural network verification

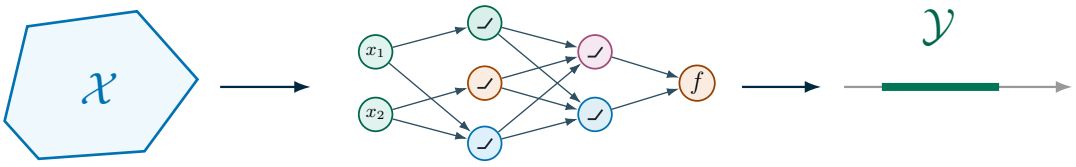
Given $f : \mathbb{R}^d \to \mathbb{R}^m$, $\mathcal{X} \subseteq \mathbb{R}^d$, $\mathcal{Y} \subseteq \mathbb{R}^m$



Froese, Grillo, Hertrich and Stargalla (2026). *Parameterized Hardness of Zonotope Containment and Neural Network Verification.*

Neural network verification

Given $f : \mathbb{R}^d \to \mathbb{R}^m$, $\mathcal{X} \subseteq \mathbb{R}^d$, $\mathcal{Y} \subseteq \mathbb{R}^m$

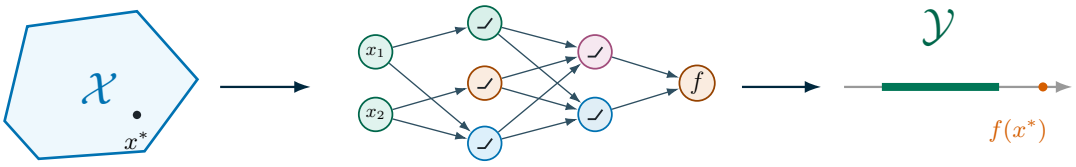


$$f(\mathcal{X}) \subseteq \mathcal{Y} \, ?$$

Froese, Grillo, Hertrich and Stargalla (2026). *Parameterized Hardness of Zonotope Containment and Neural Network Verification.*

Neural network verification

Given $f : \mathbb{R}^d \to \mathbb{R}^m$, $\quad \mathcal{X} \subseteq \mathbb{R}^d$, $\quad \mathcal{Y} \subseteq \mathbb{R}^m$



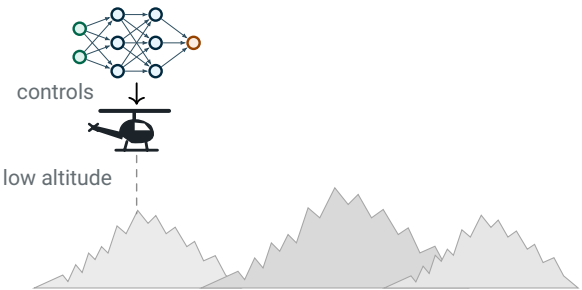
$$f(\mathcal{X}) \subseteq \mathcal{Y} \, ?$$

Counterexample: $x^* \in \mathcal{X}$: $\quad f(x^*) \notin \mathcal{Y}$

Neural network verification

$u : \mathbb{R}^d \to \mathbb{R}$ controls altitude

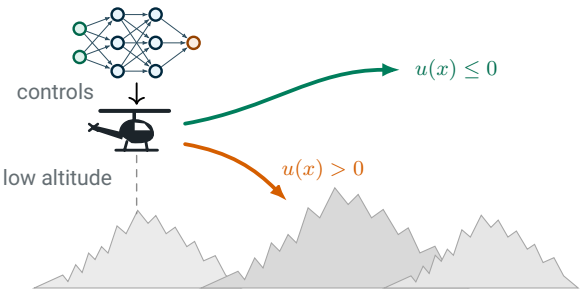
$u(x) > 0 \Rightarrow$ descent, $\quad u(x) \leq 0 \Rightarrow$ ascent



Neural network verification

$u : \mathbb{R}^d \to \mathbb{R}$ controls altitude

$u(x) > 0 \Rightarrow$ descent, $\quad u(x) \leq 0 \Rightarrow$ ascent



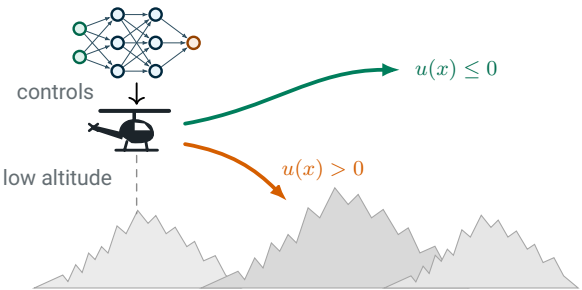
$\mathcal{X}$: low-altitude sensor inputs

$\mathcal{Y} = (-\infty, 0]$: no descent

Neural network verification

$u : \mathbb{R}^d \to \mathbb{R}$ controls altitude

$u(x) > 0 \Rightarrow$ descent, $\quad u(x) \leq 0 \Rightarrow$ ascent



$\mathcal{X}$: low-altitude sensor inputs

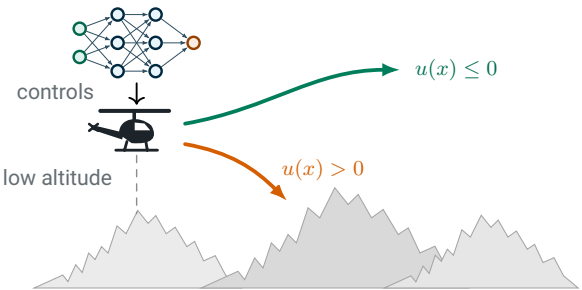
$\mathcal{Y} = (-\infty, 0]$: no descent

$$u(\mathcal{X}) \subseteq (-\infty, 0]$$

Neural network verification

$u : \mathbb{R}^d \to \mathbb{R}$ controls altitude

$u(x) > 0 \Rightarrow$ descent, $\quad u(x) \leq 0 \Rightarrow$ ascent



$\mathcal{X}$: low-altitude sensor inputs

$\mathcal{Y} = (-\infty, 0]$: no descent

$$\exists\, x \in \mathcal{X} :\ u(x) > 0\ ?$$

Special case of Verification: Positivity

Positivity

Does there exist $x \in \mathbb{R}^d$ with $f(x) > 0$?

Special case of Verification: Positivity

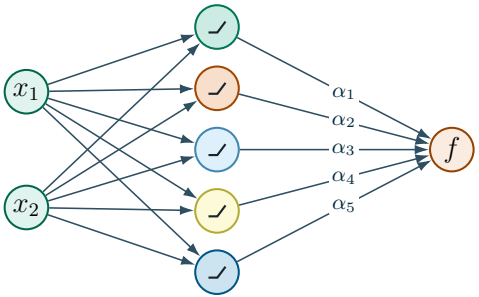
Positivity

Does there exist $x \in \mathbb{R}^d$ with $f(x) > 0$?

$$f(\mathbb{R}^d) \subseteq (-\infty, 0] \quad \Longleftrightarrow \quad \nexists\, x : \; f(x) > 0$$

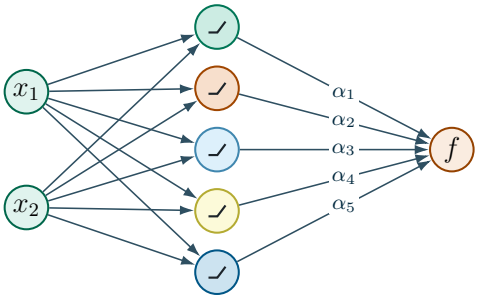
Positivity: Does the verification instance have a counterexample?

ReLU networks $f : \mathbb{R}^d \to \mathbb{R}$ with one ReLU layer



$$f(x) = \sum_{i=1}^{n} \alpha_i \max\{0, w_i^\top x\}$$

ReLU networks $f : \mathbb{R}^d \to \mathbb{R}$ with one ReLU layer

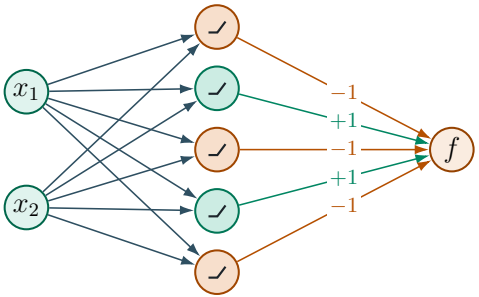


$$\alpha_i \max\{0, w_i^\top x\}$$

$$= \operatorname{sgn}(\alpha_i) \max\{0, (|\alpha_i| w_i)^\top x\}$$

$$f(x) = \sum_{i=1}^{n} \alpha_i \max\{0, w_i^\top x\}$$

ReLU networks $f : \mathbb{R}^d \to \mathbb{R}$ with one ReLU layer



$$\alpha_i \max\{0, w_i^\top x\}$$

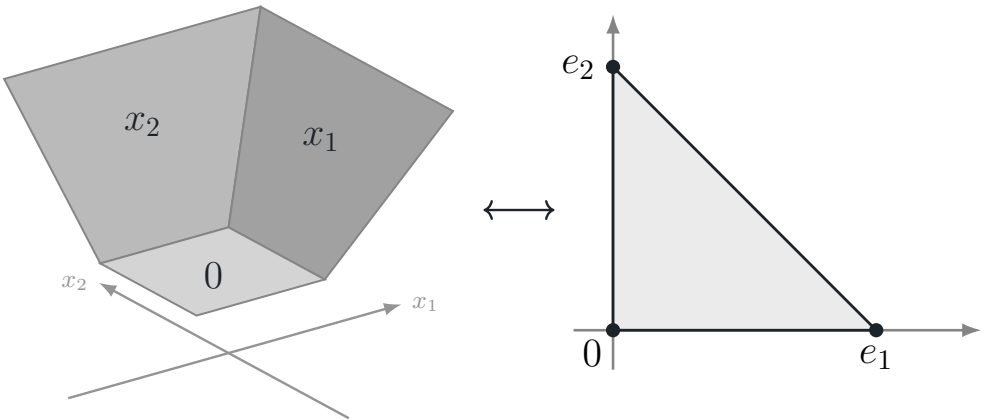
$$= \operatorname{sgn}(\alpha_i) \max\{0, (|\alpha_i| w_i)^\top x\}$$

$$\downarrow$$

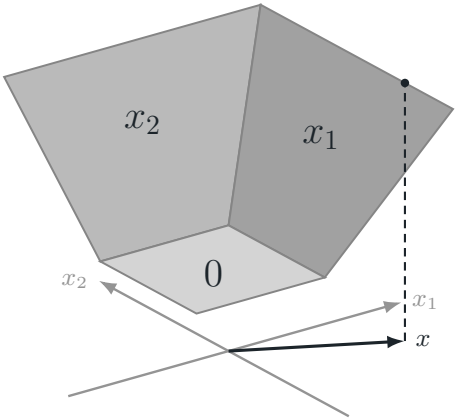
Wlog: $\alpha_i \in \{-1, +1\}$

$$f(x) = \sum_{i \in I^+} \max\{0, w_i^\top x\} - \sum_{i \in I^-} \max\{0, w_i^\top x\}$$

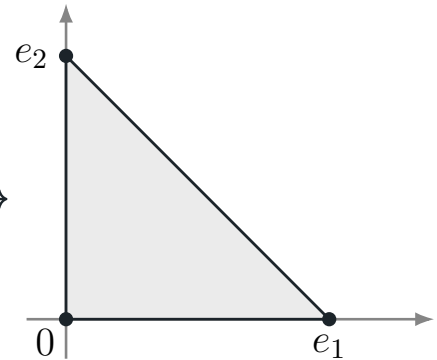
Duality



Support functions

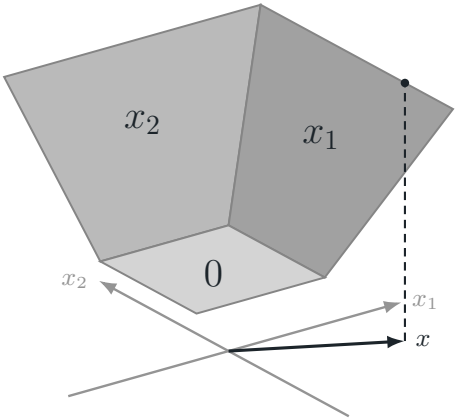


$\longleftrightarrow$

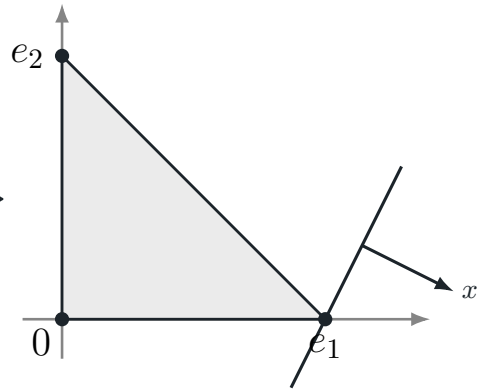


$$h_P(x) = \max_{z \in P} x^\top z \qquad \longleftrightarrow \qquad P$$

Support functions

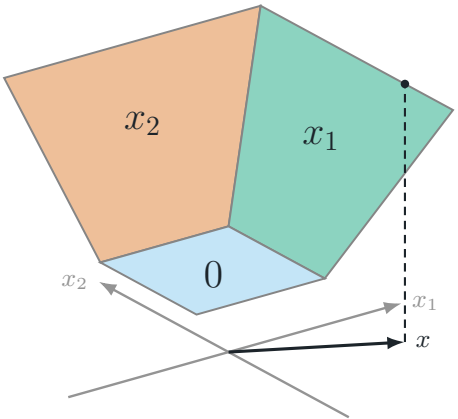


$\longleftrightarrow$

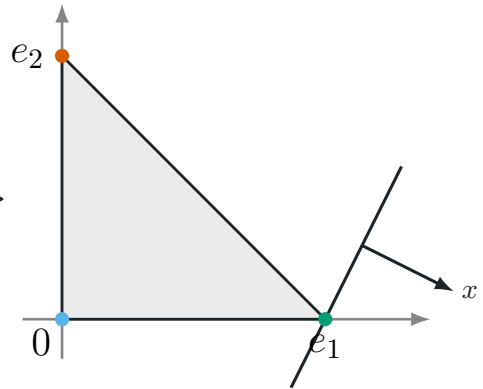


$$h_P(x) = \max_{z \in P} x^\top z \qquad \longleftrightarrow \qquad P$$

Support functions

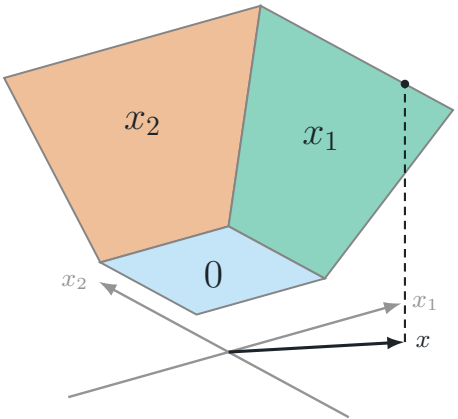


$\longleftrightarrow$

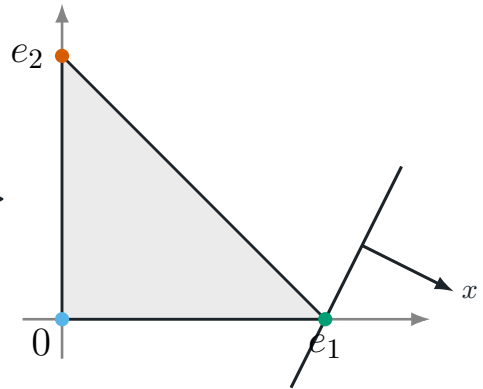


$$h_P(x) = \max_{z \in P} x^\top z \qquad \longleftrightarrow \qquad P$$

Support functions



$\longleftrightarrow$



$$h_P(x) = \max_{z \in P} x^\top z \qquad \longleftrightarrow \qquad P$$

Support functions

$$P \quad \longleftrightarrow \quad h_P(x) = \max_{z \in P} x^\top z$$

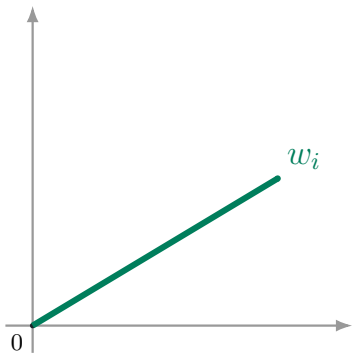
Support functions

$$P \quad \longleftrightarrow \quad h_P(x) = \max_{z \in P} x^\top z$$

$$\max\{0, w_i^\top x\}$$

Support functions

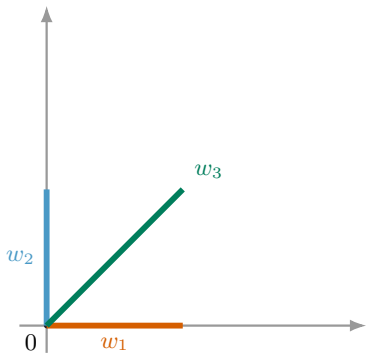
$$P \quad \longleftrightarrow \quad h_P(x) = \max_{z \in P} x^\top z$$



$$h_{[0,w_i]}(x) = \max\{0, w_i^\top x\}$$

Support functions

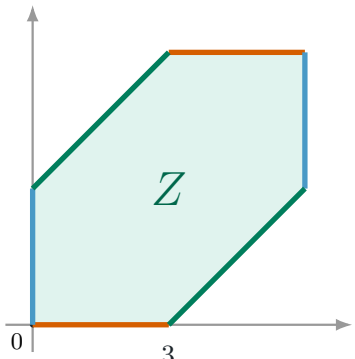
$$P \quad \longleftrightarrow \quad h_P(x) = \max_{z \in P} x^\top z$$



$$\sum_{i=1}^{3} \max\{0, w_i^\top x\}$$

Support functions

$$P \quad \longleftrightarrow \quad h_P(x) = \max_{z \in P} x^\top z$$

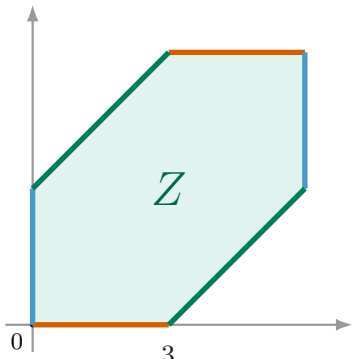


$$Z = \sum_{i=1}^{3} [0, w_i]$$

$$\sum_{i=1}^{3} \max\{0, w_i^\top x\}$$

Support functions

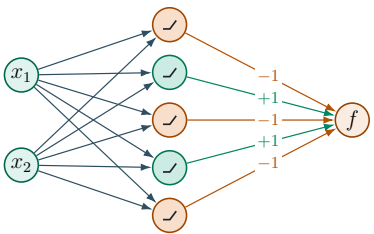
$$P \quad \longleftrightarrow \quad h_P(x) = \max_{z \in P} x^\top z$$

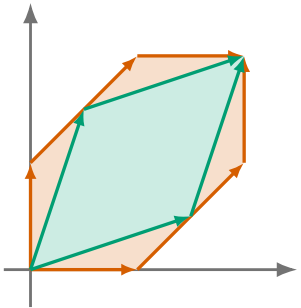


$$Z = \sum_{i=1}^{3} [0, w_i]$$

$$h_Z(x) = \sum_{i=1}^{3} \max\{0, w_i^\top x\}$$

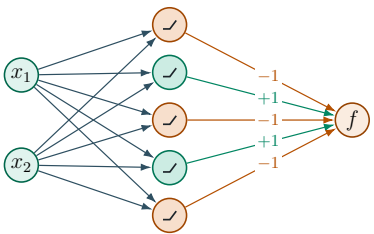
Positivity and Zonotope Containment

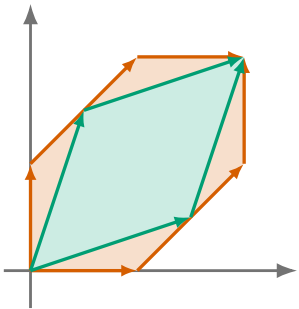




$$f(x) = \sum_{i \in I^+} \max\{0, w_i^\top x\} \quad - \quad \sum_{i \in I^-} \max\{0, w_i^\top x\}$$

Positivity and Zonotope Containment



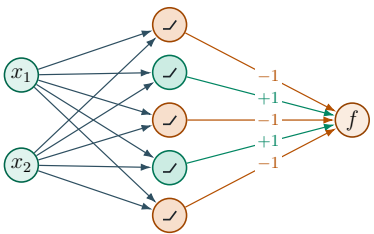


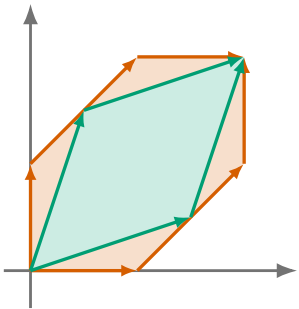
$$f(x) = \sum_{i \in I^+} \max\{0, w_i^\top x\} - \sum_{i \in I^-} \max\{0, w_i^\top x\}$$

$$\sum_{i \in I^+} [0, w_i] \qquad \sum_{i \in I^-} [0, w_i]$$

support function

Positivity and Zonotope Containment





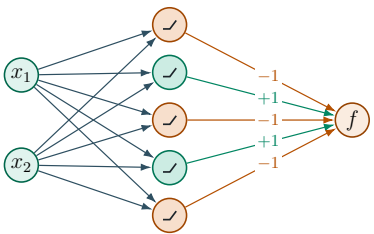
Positivity

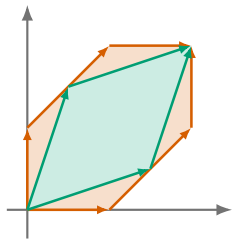
$$\exists x \in \mathbb{R}^d : \sum_{i \in I^+} \max\{0, w_i^\top x\} - \sum_{i \in I^-} \max\{0, w_i^\top x\} > 0$$

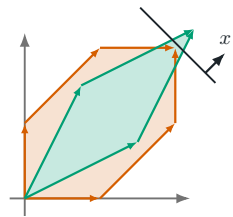
$$\sum_{i \in I^+} [0, w_i] \qquad \sum_{i \in I^-} [0, w_i]$$

support function

Positivity and Zonotope Containment







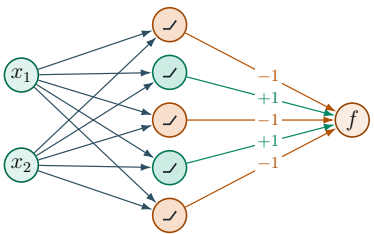
Positivity

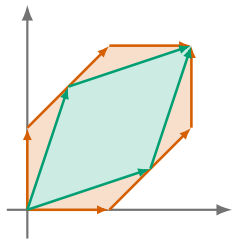
$$\exists x \in \mathbb{R}^d : \sum_{i \in I^+} \max\{0, w_i^\top x\} \; - \sum_{i \in I^-} \max\{0, w_i^\top x\} > 0$$

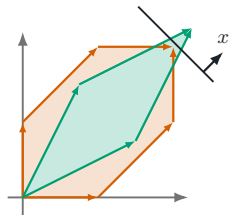
$$\sum_{i \in I^+} [0, w_i] \qquad \sum_{i \in I^-} [0, w_i]$$

support function

Positivity and Zonotope Containment







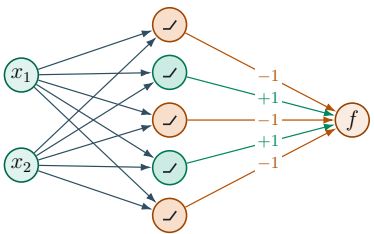
Positivity

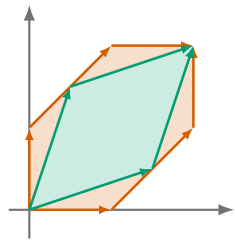
$$\exists x \in \mathbb{R}^d : \sum_{i \in I^+} \max\{0, w_i^\top x\} - \sum_{i \in I^-} \max\{0, w_i^\top x\} > 0$$

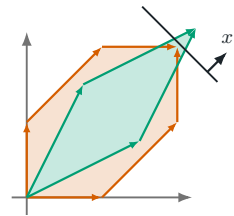
$$\iff \sum_{i \in I^+} [0, w_i] \not\subseteq \sum_{i \in I^-} [0, w_i]$$

support function

Positivity and Zonotope Containment







Positivity

$$\exists x \in \mathbb{R}^d : \sum_{i \in I^+} \max\{0, w_i^\top x\} - \sum_{i \in I^-} \max\{0, w_i^\top x\} > 0$$

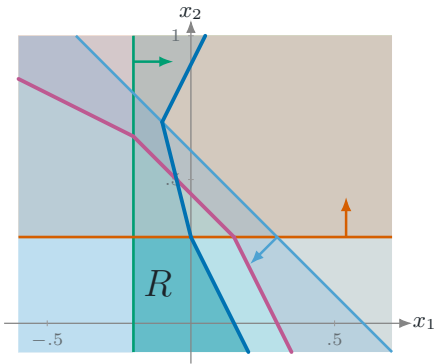
$$\iff$$

$$\sum_{i \in I^+} [0, w_i] \not\subseteq \sum_{i \in I^-} [0, w_i]$$

support function

Zonotope Non-Containment

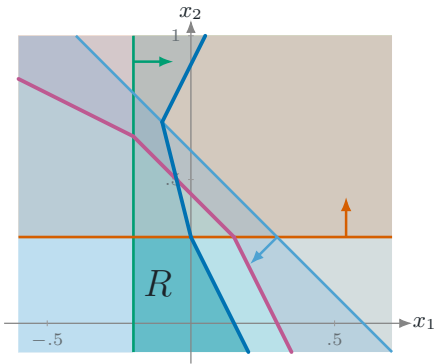
Brute-force solution for Positivity



Linear region R

$$f_R(x) = a_R^\top x + b_R$$

Brute-force solution for Positivity

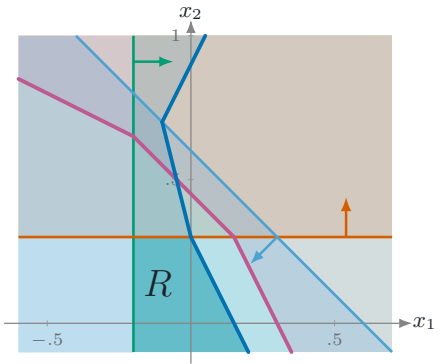


Linear region R

$$f_R(x) = a_R^\top x + b_R$$

$$\max_{x \in R} \left(a_R^\top x + b_R \right) > 0$$

Brute-force solution for Positivity



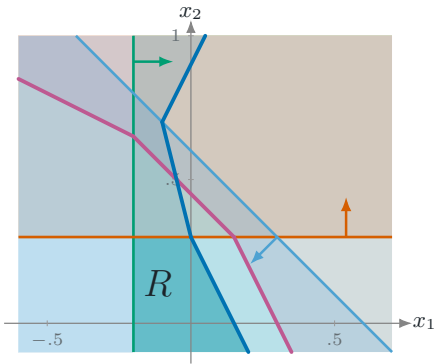
Linear region R

$$f_R(x) = a_R^\top x + b_R$$

$$\max_{x \in R} \left(a_R^\top x + b_R \right) > 0$$

$$\implies \exists x \in \mathbb{R}^d : \; f(x) > 0$$

Brute-force solution for Positivity

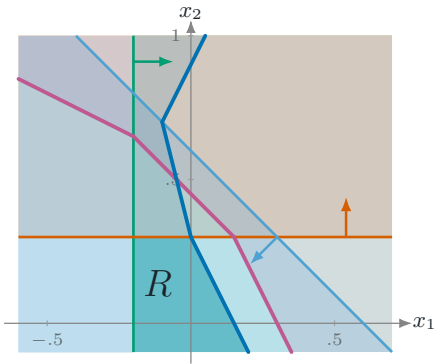


Brute-force algorithm

– Enumerate linear regions.

– Solve an LP for every linear region.

Brute-force solution for Positivity



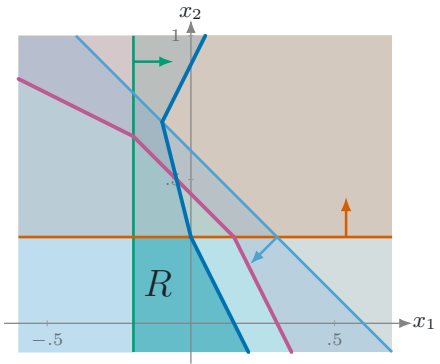
Brute-force algorithm

– Enumerate linear regions.

– Solve an LP for every linear region.

– $O(n^{\ell d})$ linear regions

Brute-force solution for Positivity



Brute-force algorithm

– Enumerate linear regions.

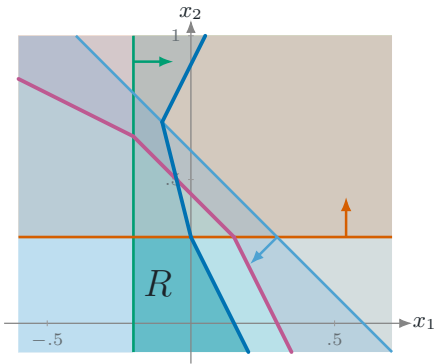
– Solve an LP for every linear region.

– $O(n^{\ell d})$ linear regions

$$\Longrightarrow\ n^{\ell d}\,\mathrm{poly}(N)\ \text{time}$$

N: encoding size (ℓ, d fixed)

Brute-force solution for Positivity



Brute-force algorithm

– Enumerate linear regions.

– Solve an LP for every linear region.

– $O(n^{\ell d})$ linear regions

$$\Longrightarrow\ n^{\ell d}\,\mathrm{poly}(N)\ \text{time}$$

N: encoding size (ℓ, d fixed)

Can we do better?

Can we do better than $n^{\ell d}\,\mathrm{poly}(N)$?

Theorem (Froese, Grillo, Hertrich and S)

Assuming the Exponential Time Hypothesis,
Positivity (and thus Verification) is not solvable in
$n^{o(\ell d)}\,\mathrm{poly}(N)$ time.

Can we do better than $n^{\ell d}\,\mathrm{poly}(N)$?

Theorem (Froese, Grillo, Hertrich and S)

Assuming the Exponential Time Hypothesis,
Positivity (and thus Verification) is not solvable in
$n^{o(\ell d)}\,\mathrm{poly}(N)$ time.

Enumeration is essentially optimal!

Takeaways



Expressivity

Understanding ReLU networks requires understanding their (polyhedral) geometry.

- Linear regions
- Depth vs width tradeoff
- Required depth

Verification

Polyhedral structure does not by itself allow for fast algorithms for certain problems.

- Positivity and zonotope containment
- Brute-force essentially optimal for verification

Takeaways



Expressivity

Understanding ReLU networks requires understanding their (polyhedral) geometry.

- Linear regions
- Depth vs width tradeoff
- Required depth

Verification

Polyhedral structure does not by itself allow for fast algorithms for certain problems.

- Positivity and zonotope containment
- Brute-force essentially optimal for verification

Which geometric restrictions make verification tractable?

References

Arora, Basu, Mianjy and Mukherjee (2018).
Understanding Deep Neural Networks with Rectified Linear Units.

Zaslavsky (1975).
Facing up to Arrangements: Face-Count Formulas for Partitions of Space by Hyperplanes.

Raghu, Poole, Kleinberg, Ganguli and Dickstein (2017).
On the Expressive Power of Deep Neural Networks.

Telgarsky (2016).
Benefits of Depth in Neural Networks.

Wang and Sun (2005).
Generalization of Hinging Hyperplanes.

Hertrich, Basu, Di Summa and Skutella (2021).
Towards Lower Bounds on the Depth of ReLU Neural Networks.

Bakaev, Brunck, Hertrich, Stade and Yehudayoff (2025).
Better Neural Network Expressivity: Subdividing the Simplex.

Rueß, Averkov, Brunck, Grillo, Hertrich, Loho, Stade, Stargalla, Sun and Winter (2026).
Shallower ReLU Network Representations via Exact Linear Algebra.

Froese, Grillo, Hertrich and Stargalla (2026).
Parameterized Hardness of Zonotope Containment and Neural Network Verification.