



Computer Vision, Speech Communication & Signal Processing Group,  
Intelligent Robotics and Automation Laboratory  
National Technical University of Athens, Greece (NTUA)

# Introduction to Tropical Geometry and its Applications to Machine Learning

**Petros Maragos**

---

slides: <https://robotics.ntua.gr/ivmsp2022-tutorial/>

Tutorial at IEEE IVMSp-2022, Nafplio, Greece, 26 June 2022

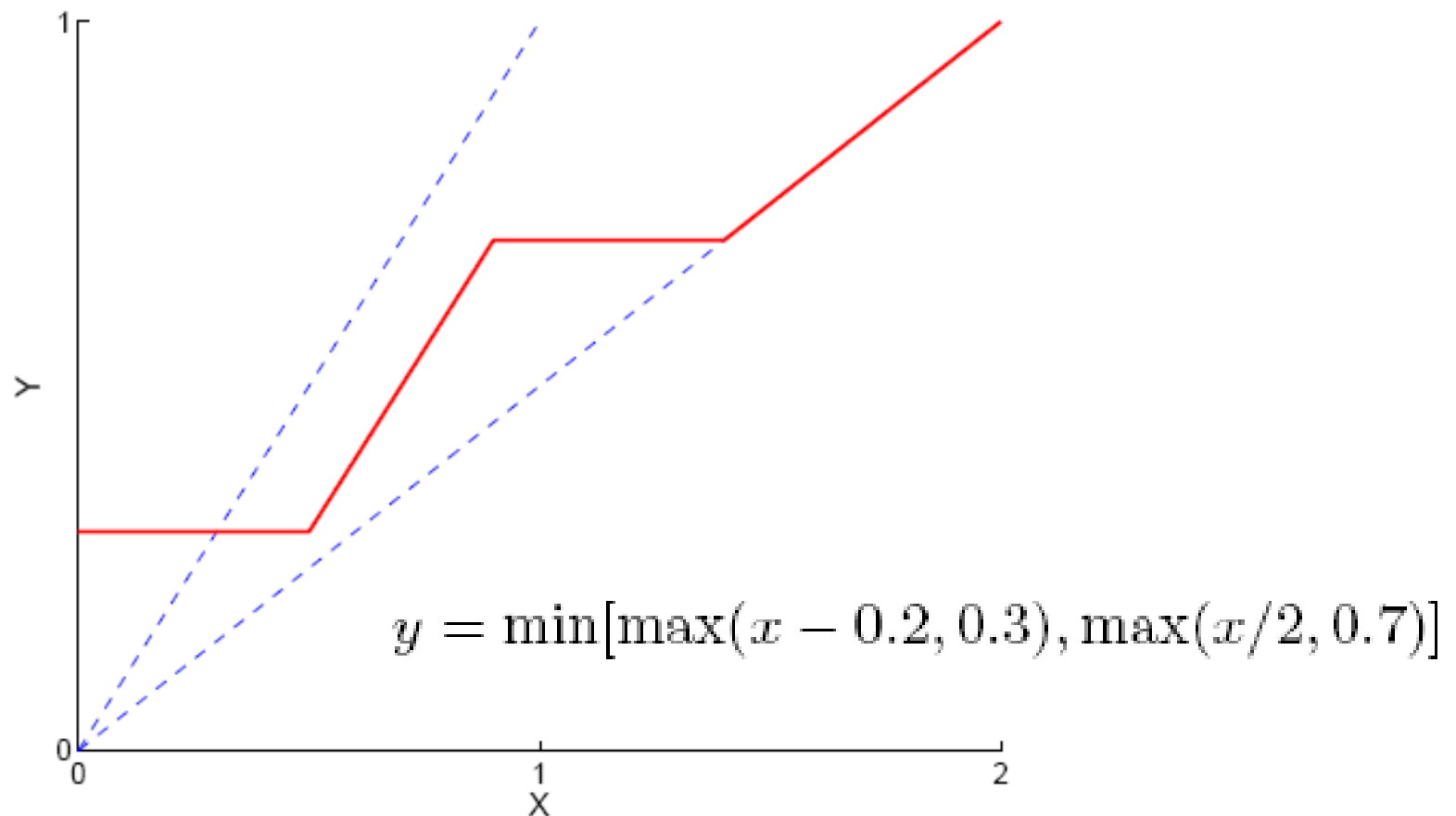
The research work is supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the “2nd Call for H.F.R.I. Research Projects to support Faculty Members & Researchers” (Project Number:2656).



# What does TROPICAL mean?

- The adjective “**tropical**” was coined by French mathematicians Dominique Perrin and Jean-Eric Pin, to honor their Brazilian colleague Imre Simon, a pioneer of min-plus algebra as applied to finite automata in computer science.
- Tropical (**Τροπικός** in Greek) comes from the greek word «**Τροπή**» which means “turning” or “changing the way/direction”.

Polygonal lines



# Outline (and some introductory references)

## 1. Elements of Tropical Geometry

“a marriage between algebraic geometry and polyhedral geometry” [Maclagan & Sturmfels 2015]

- **Tropical semirings:** Max-plus & Min-plus Arithmetic
- **Tropical Polynomials**
- **Geometrical objects:** tropical curves/surfaces, halfspaces, Newton polytopes
- **Max-plus Matrix Algebra:** “linear algebra of DP & Combinatorics” [O.R., Graphs: Cuninghame-Green 1979], [DES, Nonlinear Control: Baccelli et al. 2001, Butkovic 2010], Optimization [Gaubert et al, Max-plus group], Mathematical Morphology & Image Analysis, Idempotent Mathematics [Maslov, Litvinov, et al]

## 2. Applications to Neural Networks:

- **Tropical Geometry of NNs with Piecewise-Linear (PWL) Activations**
- **Advances in Morphological Networks: Training and Pruning**
- **NN Minimization via Tropical Polynomial Division and Zonotopes**

## 3. Optimization and Tropical Regression:

- **Optimal solutions of max-plus matrix equations**
- **Tropical Regression: fitting tropical polynomials to data**



# Tropical Semirings

## Scalar Arithmetic Rings

Integer/Real Addition-Multiplication Ring:  $(\mathbb{R}, +, \times)$ ,  $(\mathbb{Z}, +, \times)$

## Tropical Semirings

$$\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}, \quad \mathbb{R}_{\min} = \mathbb{R} \cup \{+\infty\}$$

$$\vee = \max, \quad \wedge = \min$$

**Max-plus semiring:**  $(\mathbb{R}_{\max}, \vee, +)$

**Min-plus semiring:**  $(\mathbb{R}_{\min}, \wedge, +)$

Correspondences between linear and  $(\max, +)$  arithmetic

Linear arithmetic	$(\max, +)$ arithmetic
$+$	$\max$
$\times$	$+$
$0$	$-\infty$
$1$	$0$
$x^{-1} = 1/x$	$x^{-1} = -x$

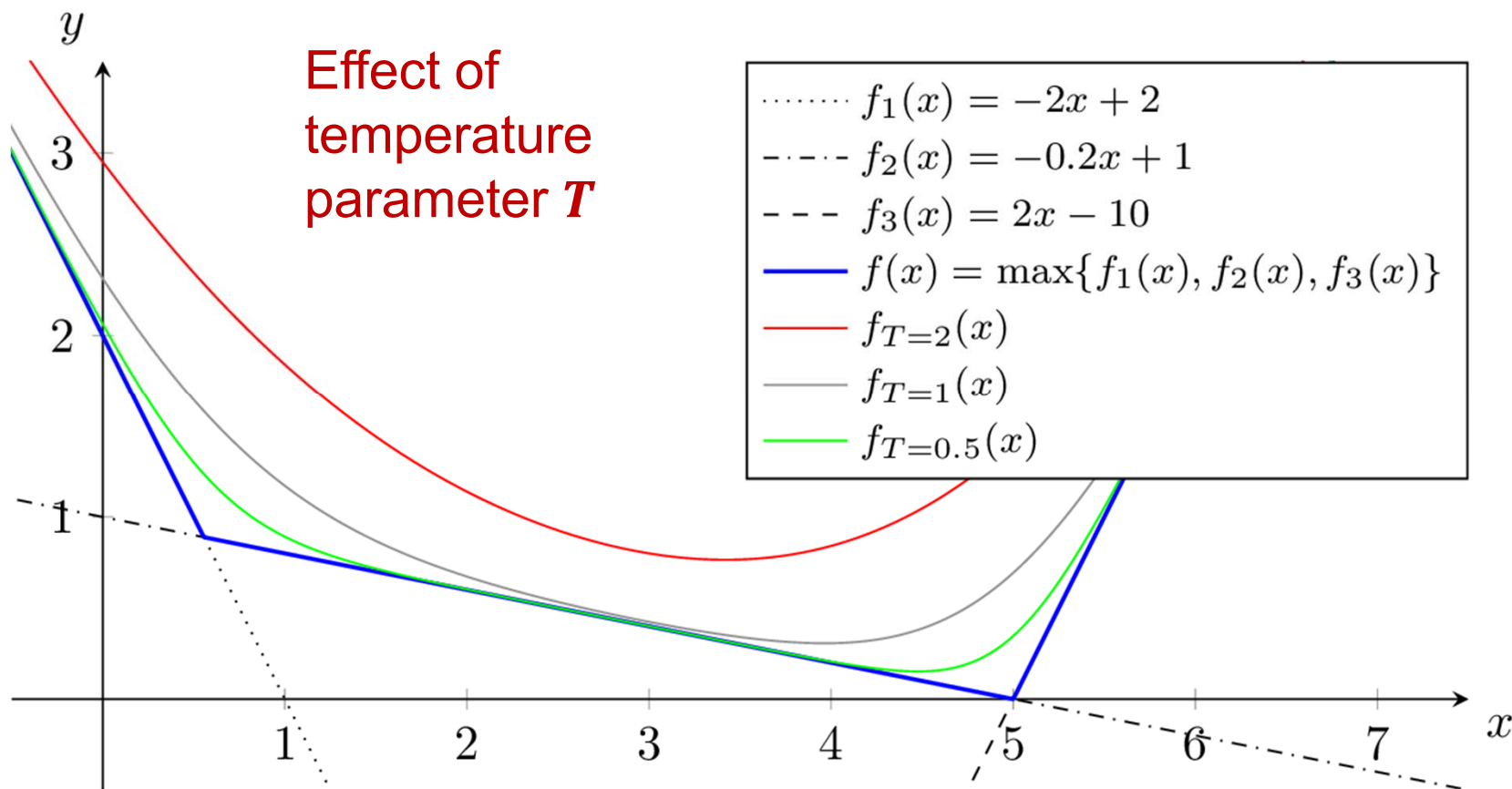
# Maslov Dequantization $\rightarrow$ Log - Sum - Exp approximation

## Log-Sum-Exp (LSE) approximation

(Maslov "Dequantization" in idempotent mathematics [Maslov 1987, Litvinov 2007])

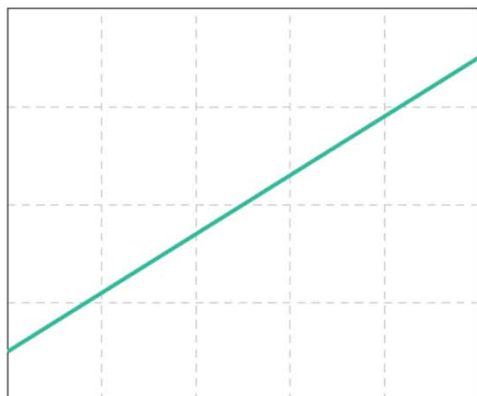
$$\lim_{T \downarrow 0} T \cdot \log(e^{a/T} + e^{b/T}) = \max(a, b)$$

$$\lim_{T \downarrow 0} (-T) \log(e^{-a/T} + e^{-b/T}) = \min(a, b)$$

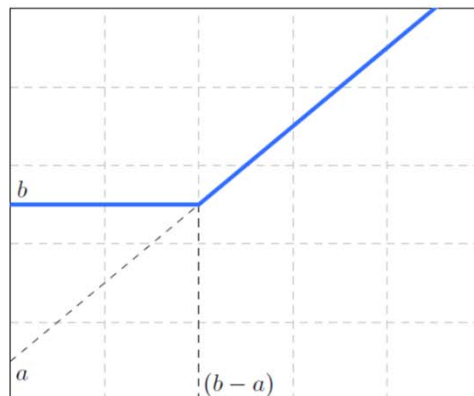


# Graphs of Max-plus Tropical 1D Polynomials

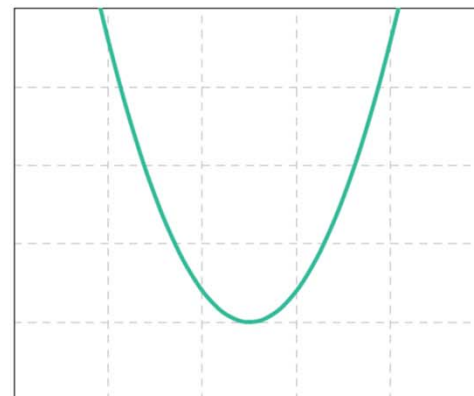
$$y_{\text{t-line}} = \max(a + x, b), \quad y_{\text{t-parab}} = \max(a + 2x, b + x, c)$$



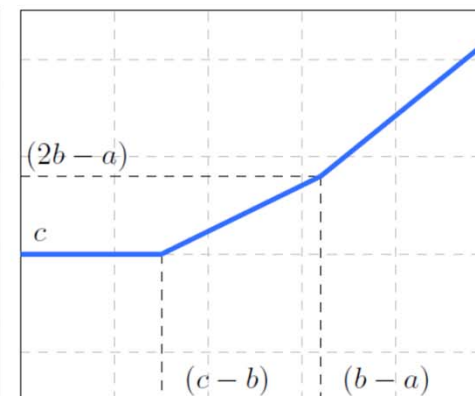
(a) Euclidean line



(b) Tropical line

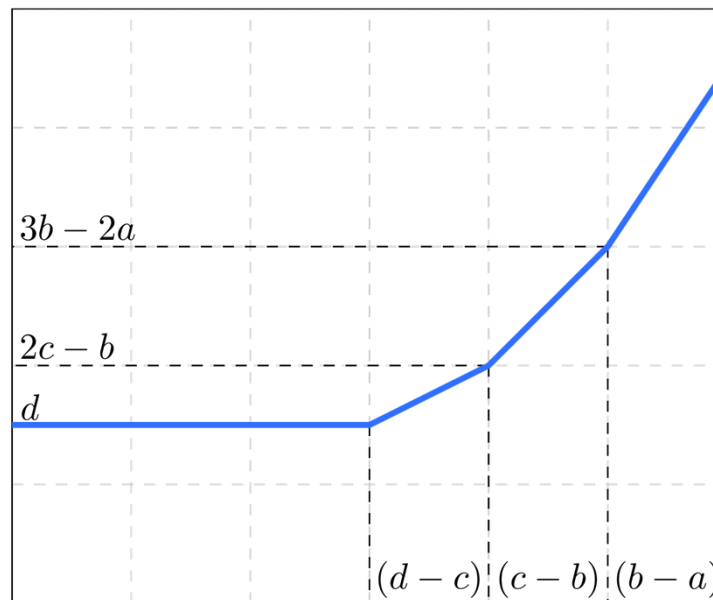
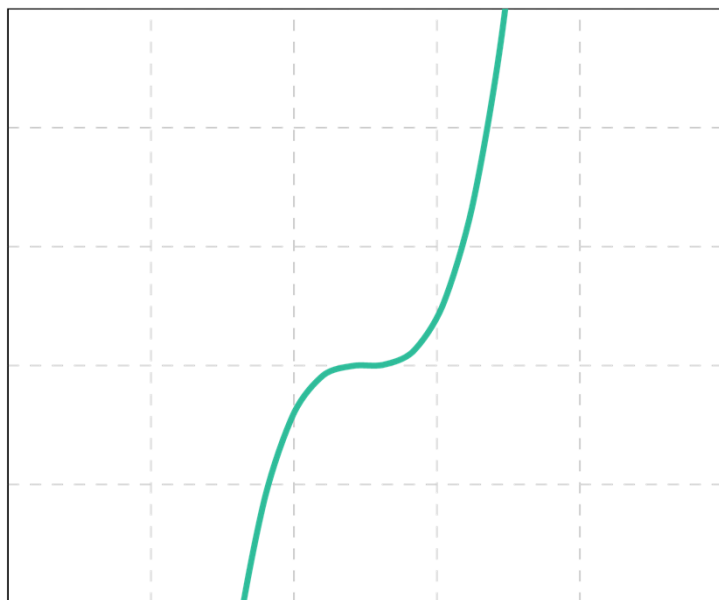


(c) Euclid parabola



(d) Tropic parabola

## Cubic polynomial

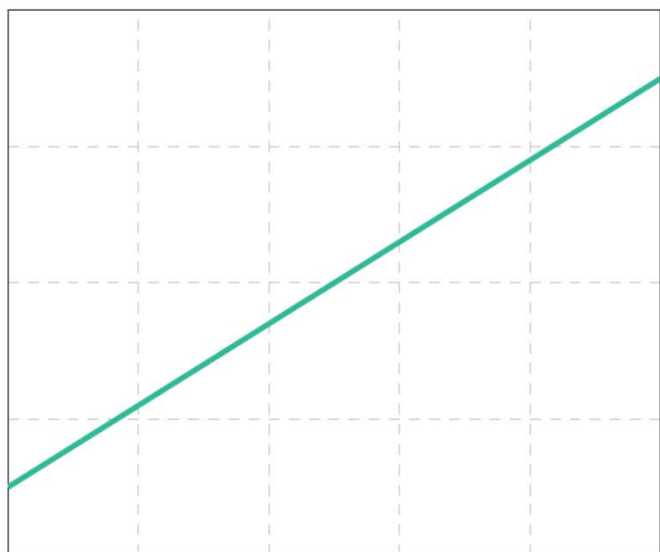


# Max-plus and Min-Plus Tropical 1D Polynomials

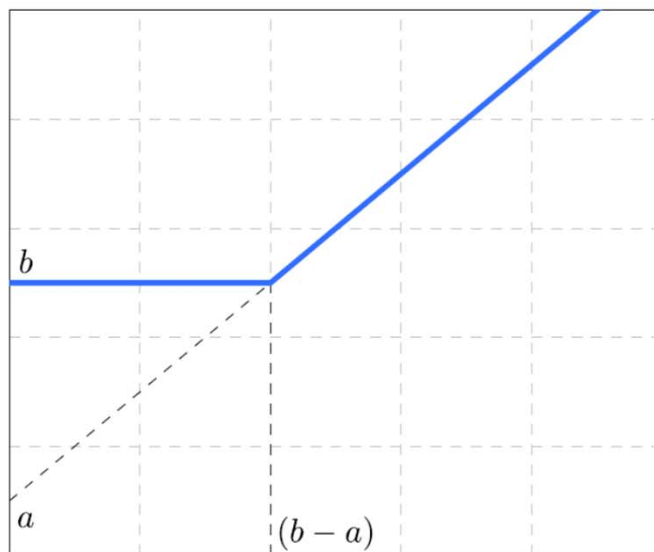
Euclidean

Max-plus

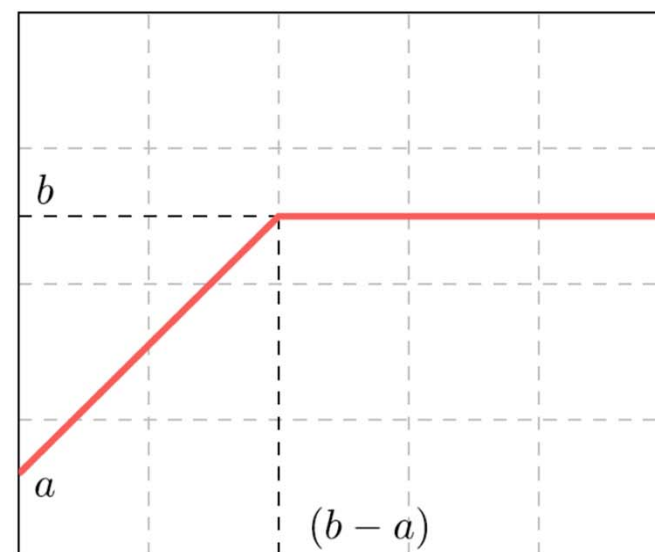
Min-plus



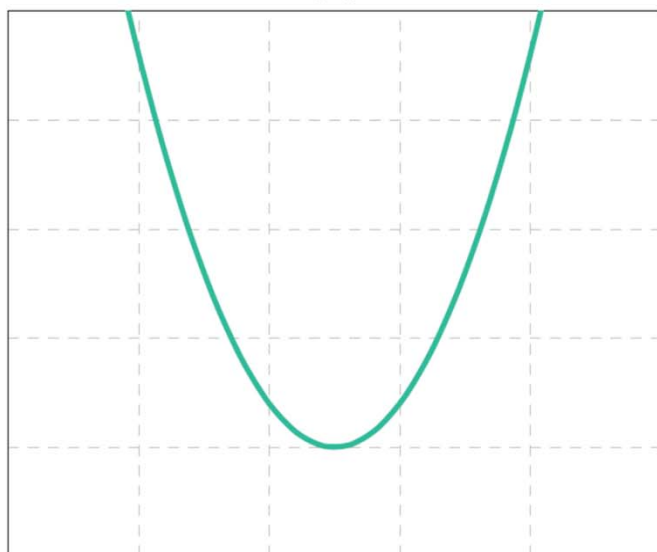
(a)



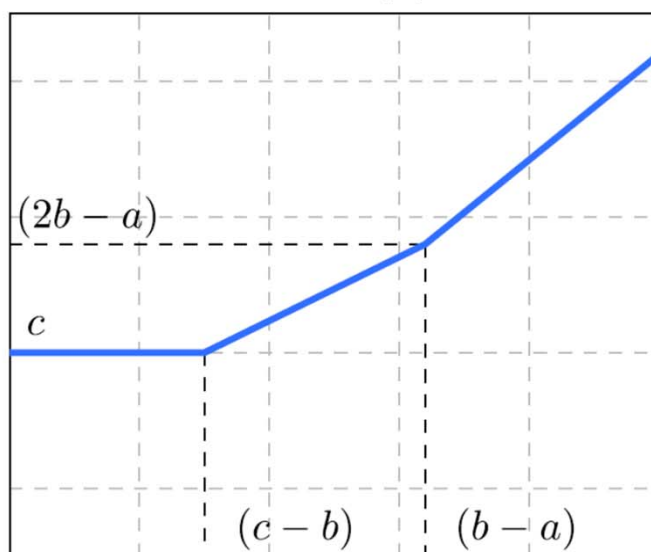
(b)



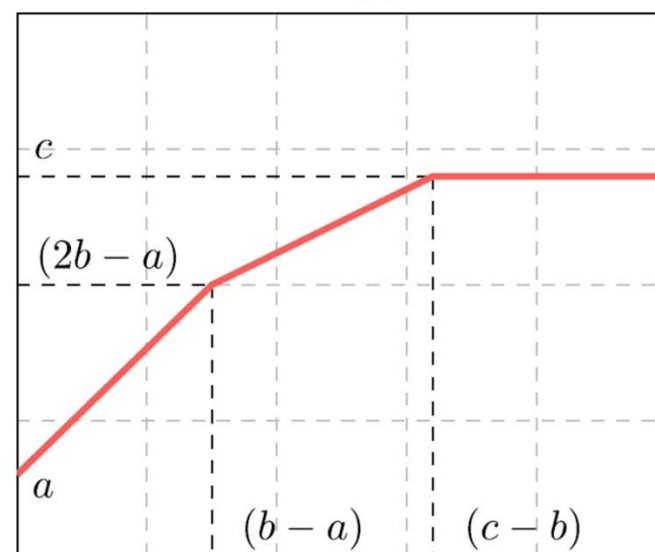
(c)



(d)



(e)

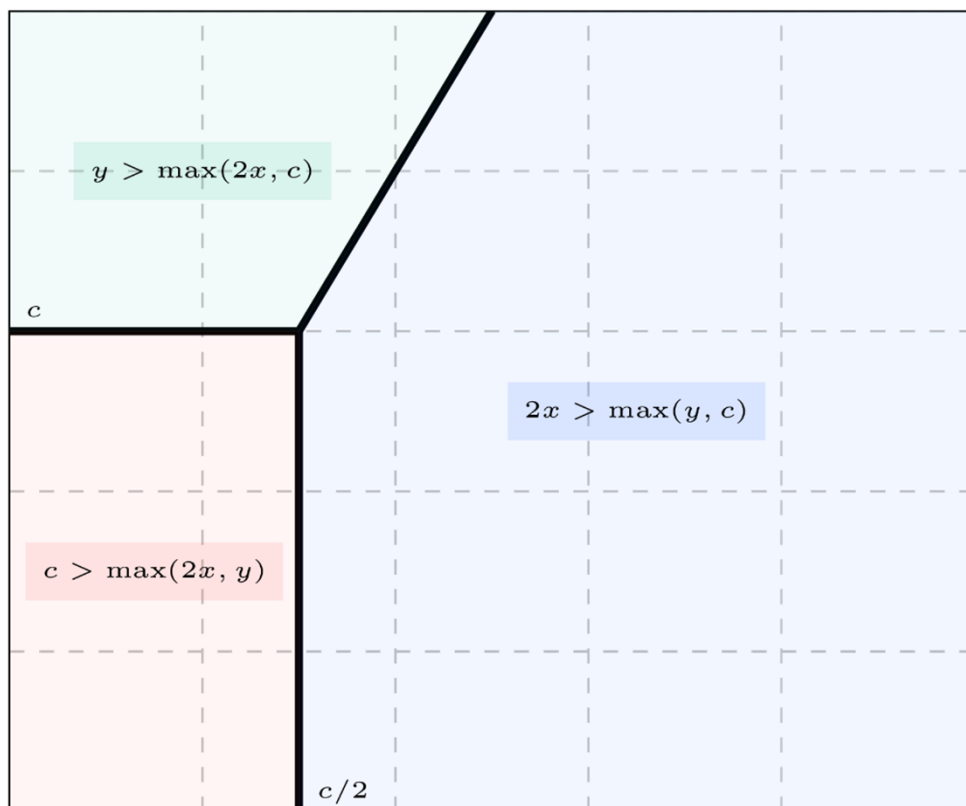


(f)

# Tropical Curve of Max/Min-Polynomials

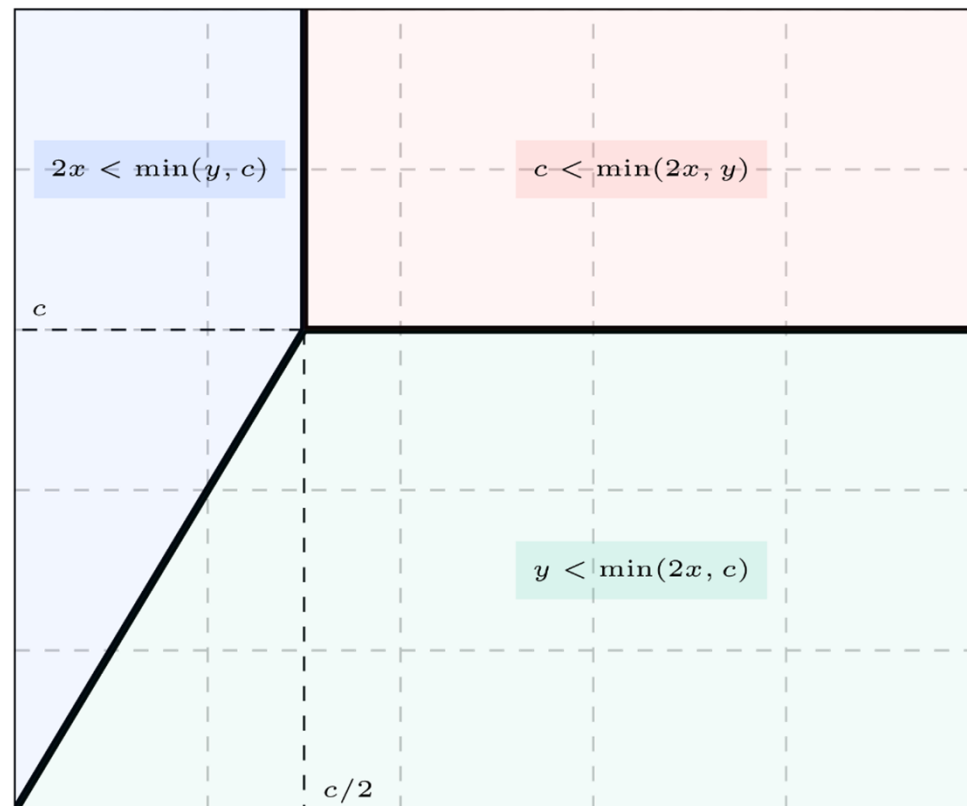
Tropical curve of  $p(x,y) =$

“Zero locus” of a max/min polynomial is the set of points where the max/min is attained by more than one of the “monomial” terms of the polynomial.



Tropical curve of the max-polynomial

$$p(x,y) = \max(2x, y, c)$$



Tropical curve of the min-polynomial

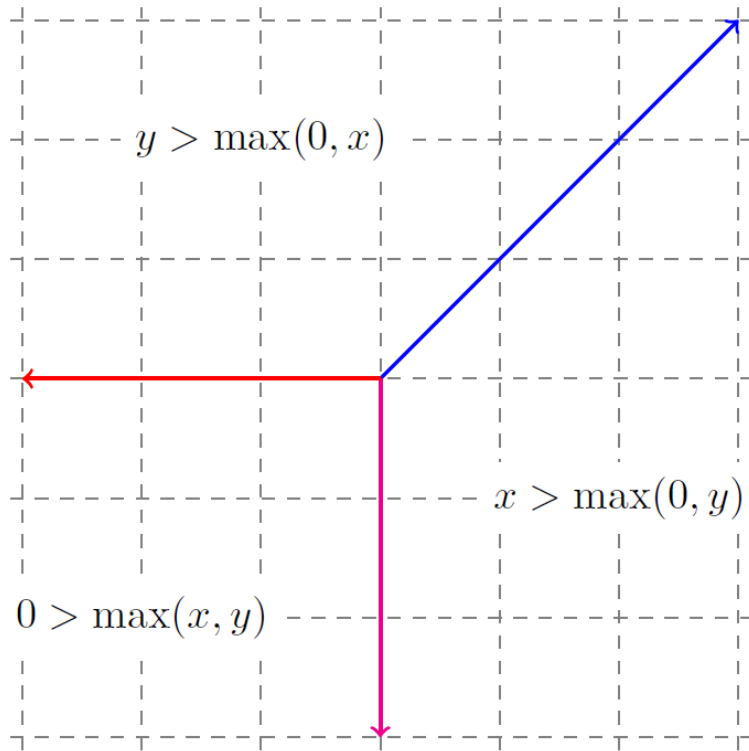
$$p'(x,y) = \min(2x, y, c)$$

# Tropical Curve vs Newton Polytope

Max polynomial

$$p(\mathbf{x}) = \max_{i \in \{1, 2, \dots, k\}} \{c_{i1}x_1 + c_{i2}x_2 + \dots + c_{in}x_n\} = \bigvee_{i=1}^k c_i^T \mathbf{x}$$

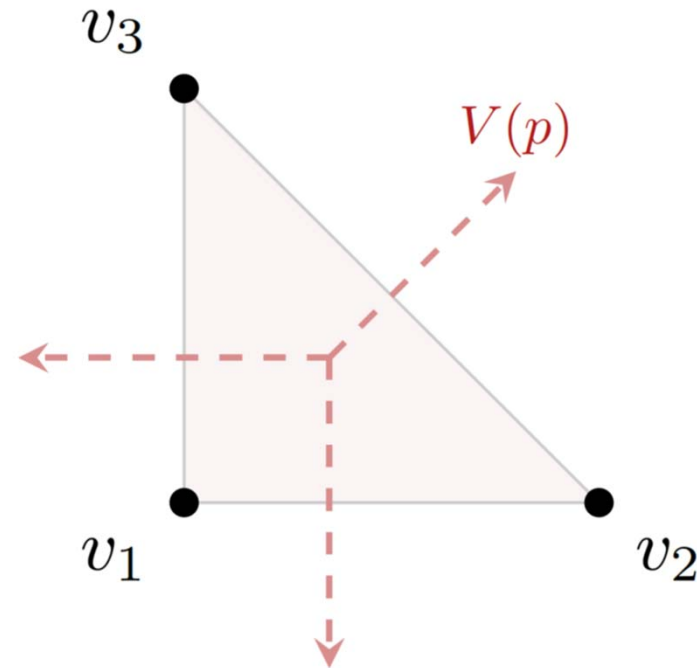
“Zero locus” of a max polynomial is the set of points where the max is attained by more than two polynomial terms.



**Tropical curve**  $V(p)$  of  
 $p(x,y) = \max(x,y,0)$

**Newton polytope**  $N(p)$  of max polynomial  $p$  is the convex hull of its coefficients' vectors.

$$\mathcal{N}(p) = \text{conv} \{v_1, v_2, v_3\}$$



**Duality** between Newton polytope  $N(p)$  and tropical curve  $V(p)$

# Graph and Trop Curve of a tropical “Conic” polynomial

## Tropical Polynomial of degree 2 in two variables

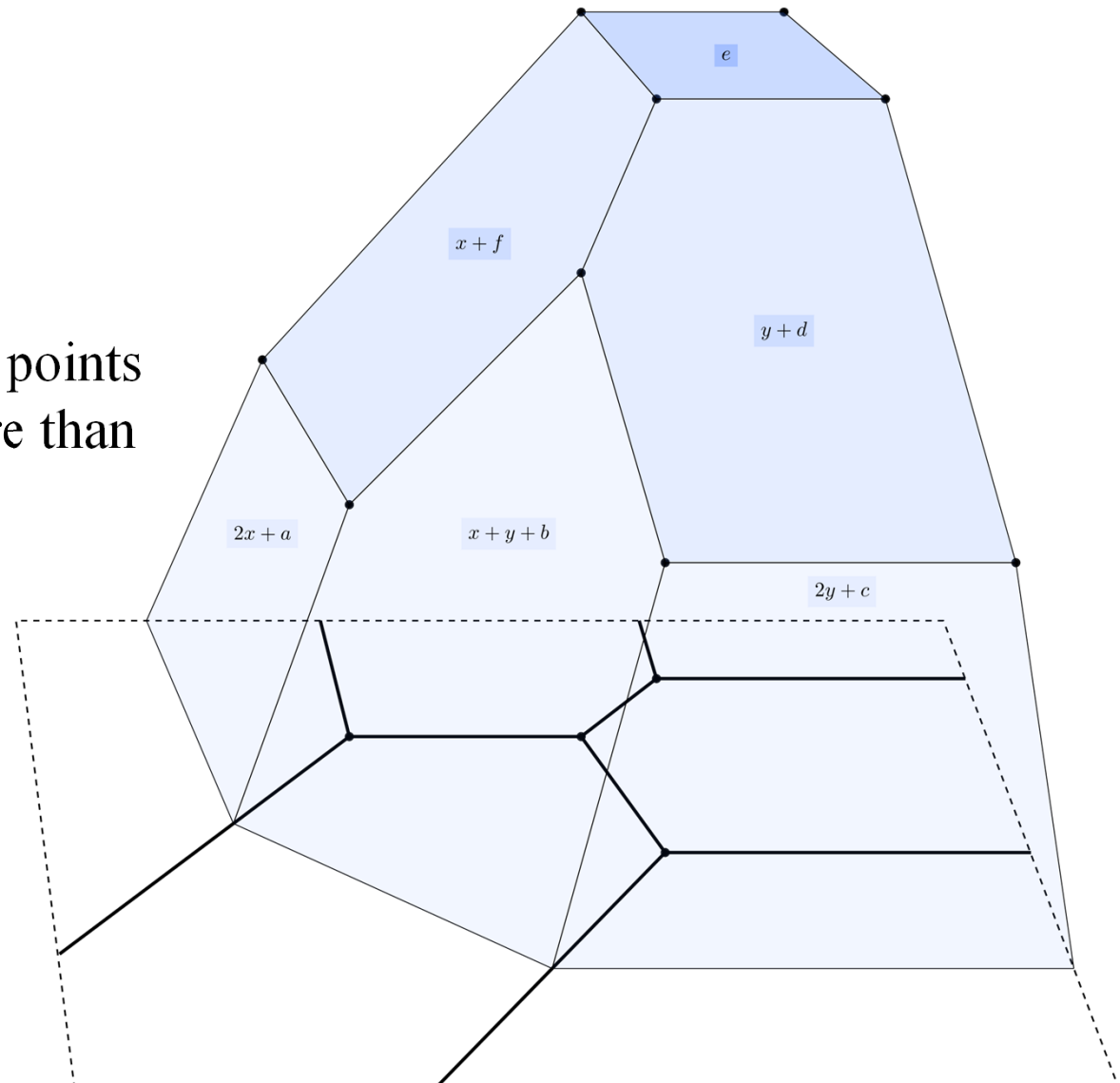
classical: " $ax^2 + bxy + cy^2 + dy + e + fx$ "

tropical:  $p(x, y) = \min(a + 2x, b + x + y, c + 2y, d + y, e, f + x)$

**Graph of  $p(x, y)$**

and

**its Tropical Curve** = set of  $(x, y)$  points where the min is attained by more than one terms.



# Obtain Tropical Polynomials via Dequantization

Classic polynomial:  $f(\mathbf{u}) = \sum_{k=1}^K c_k u_1^{a_{k1}} u_2^{a_{k2}} \cdots u_n^{a_{kn}}$ ,  $\mathbf{u} = (u_1, u_2, \dots, u_n)$

Posynomial if  $c_k > 0$ ,  $\mathbf{a}_k = (a_{k1}, \dots, a_{kn}) \in \mathbb{R}^n$ ,  $\mathbf{u} > \mathbf{0}$ ;

Log-Sum-Exp (Viro's "logarithmic paper" [Viro 2001]):

$\mathbf{x} = \log(\mathbf{u})$ ,  $b_k = \log(c_k)$

$$\lim_{T \downarrow 0} T \cdot \log f(e^{\mathbf{x}/T}) = \lim_{T \downarrow 0} T \cdot \log \sum_{k=1}^K \exp(\langle \mathbf{a}_k, \mathbf{x} / T \rangle + b_k / T) \rightarrow$$

**Tropical (max-plus) Polynomial = Piecewise-Linear Function**

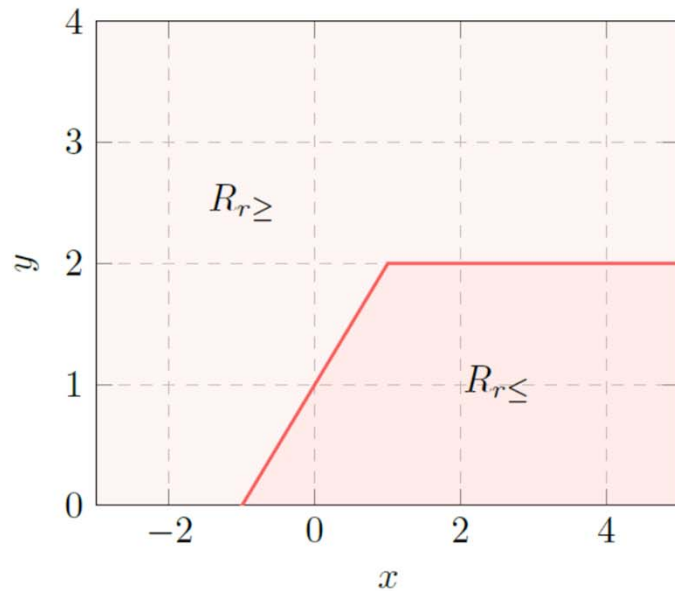
$$p(\mathbf{x}) = \text{MAX}_{k=1}^K \{ \langle \mathbf{a}_k, \mathbf{x} \rangle + b_k \} = \text{MAX}_{k=1}^K \{ a_{k1} x_1 + \cdots + a_{kn} x_n + b_k \}$$



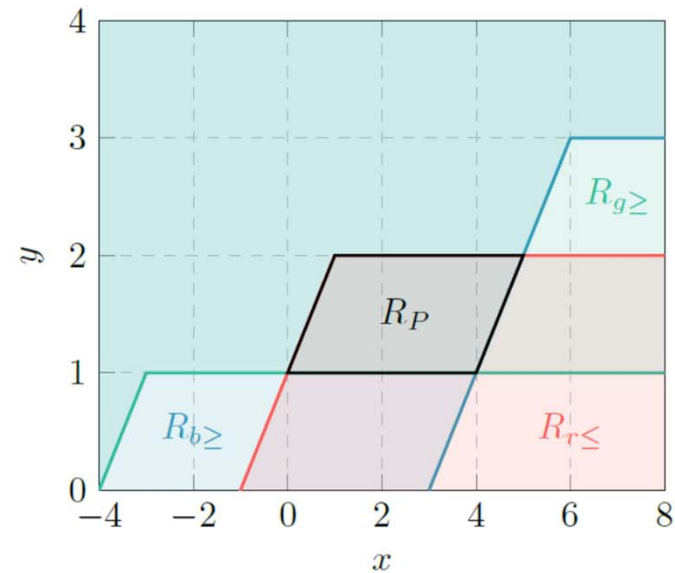
# Tropical Half-spaces and Polytopes in 2D

Tropical (affine) Half-space of  $\mathbb{R}_{\max}^n$  [ Gaubert & Katz 2011]

$$\mathcal{T}(\mathbf{a}, \mathbf{b}) \triangleq \left\{ \mathbf{x} \in \mathbb{R}_{\max}^n : \max(a_{n+1}, \bigvee_{i=1}^n a_i + x_i) \leq \max(b_{n+1}, \bigvee_{i=1}^n b_i + x_i) \right\}$$



(a) Single region



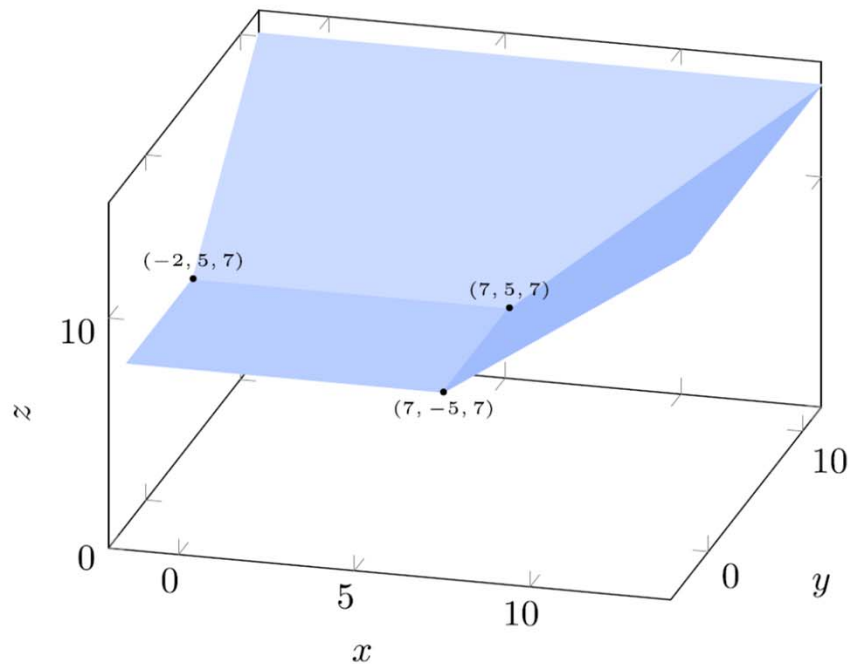
(b) Multiple regions

The region separating boundaries are tropical lines (or hyper-planes).

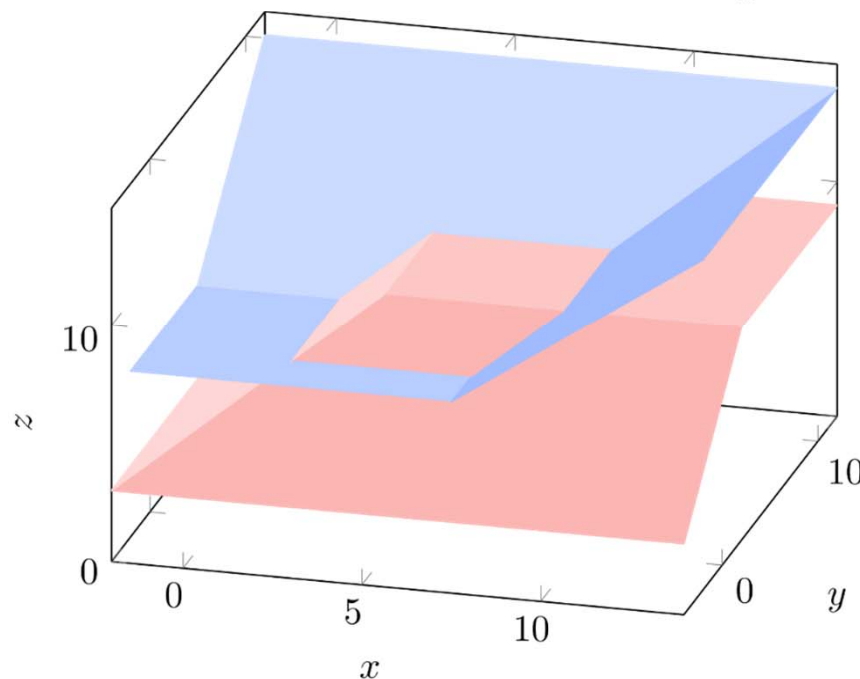
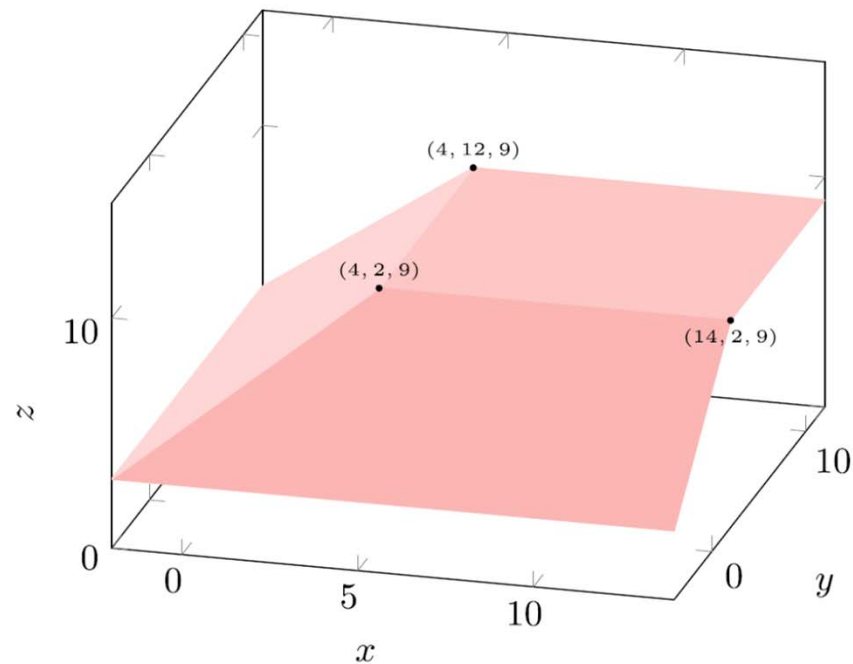
Tropical **Polyhedra** are formed from finite intersections of tropical half-spaces. **Polytopes** are compact polyhedra.

# Tropical Halfspaces and Polyhedra in 3D

$$f(x, y) = \max(x, 2 + y, 7)$$



$$g(x, y) = \min(5 + x, 7 + y, 9)$$



# (Extended) Newton Polytope

Let  $p(\mathbf{x}) = \max_{i=1,\dots,k} (\mathbf{a}_i^T \mathbf{x} + b_i)$  be a max-polynomial.

Definition ((Extended) Newton Polytope): We define as the **(Extended) Newton Polytope** of  $p$  the following:

$$\text{Newt}(p) = \text{conv}\{\mathbf{a}_i, i = 1, \dots, k\}$$

$$\text{ENewt}(p) = \text{conv}\{(\mathbf{a}_i, b_i), i = 1, \dots, k\}$$

where  $\text{conv}$  denotes the convex hull of the given set.

Theorem [Charisopoulos & Maragos, 2018; Zhang et al., 2018]:

Maxpolynomials with the same vertices in the upper hull of their Extended Newton Polytope correspond to the same function.

# Examples of (Ext) Newton Polytopes

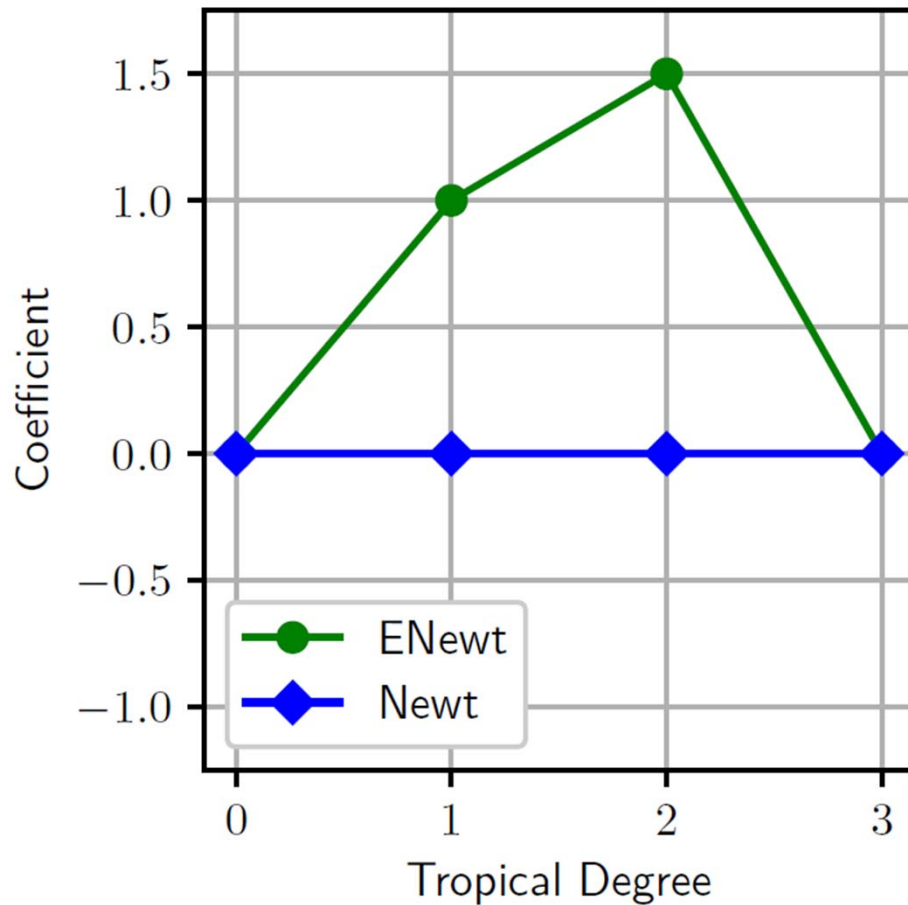


Figure: Polytopes of  $\max(3x, 2x + 1.5, x + 1, 0)$ .

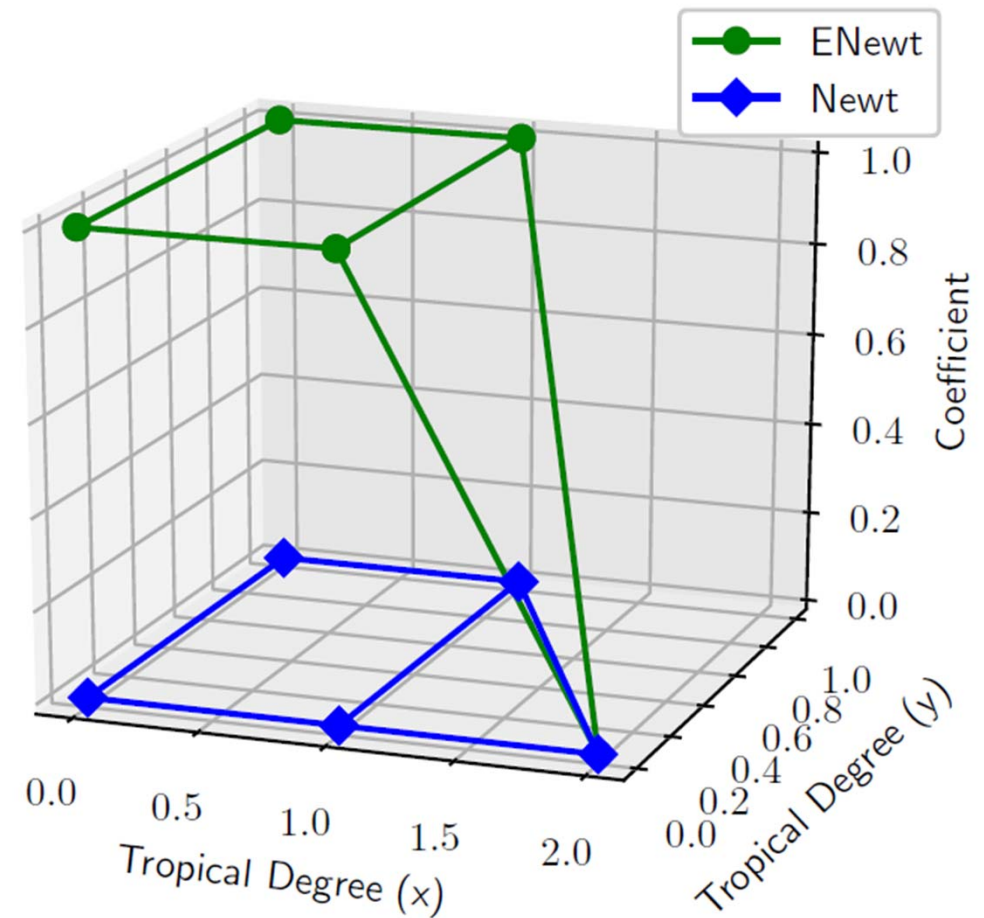
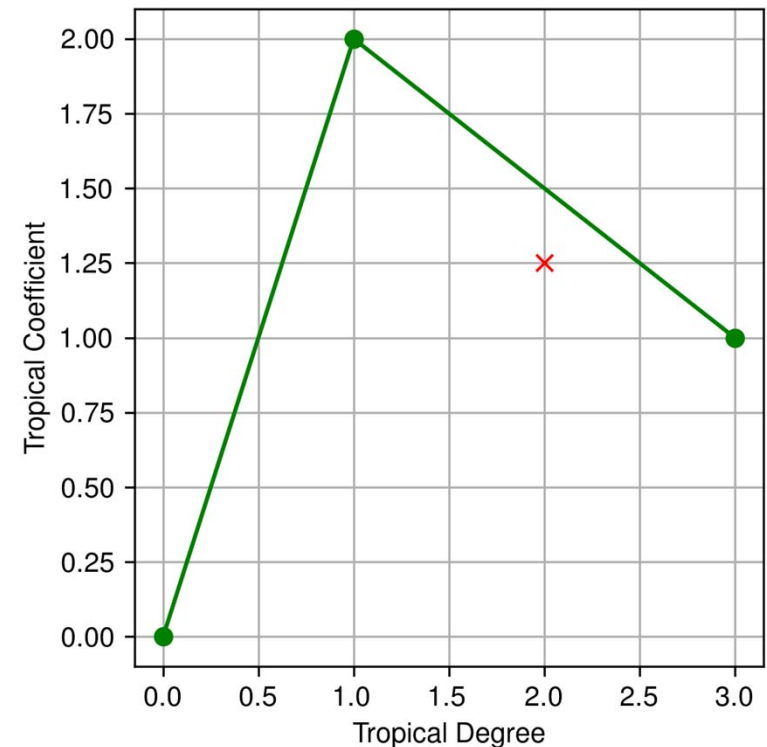


Figure: Polytopes of  $\max(2x, x + y + 1, x + 1, y + 1, 1)$ .

# Newton Polytope and Maxpolynomial Function

- “Upper” vertices of  $\text{ENewt}(p)$  define  $p(x)$  as a **function**.
- Geometrically:  
$$\max(3x + 1, 2x + 1.25, x + 2, 0)$$
$$= \max(3x + 1, x + 2, 0)$$
  
(**extra point** is not on the upper hull).

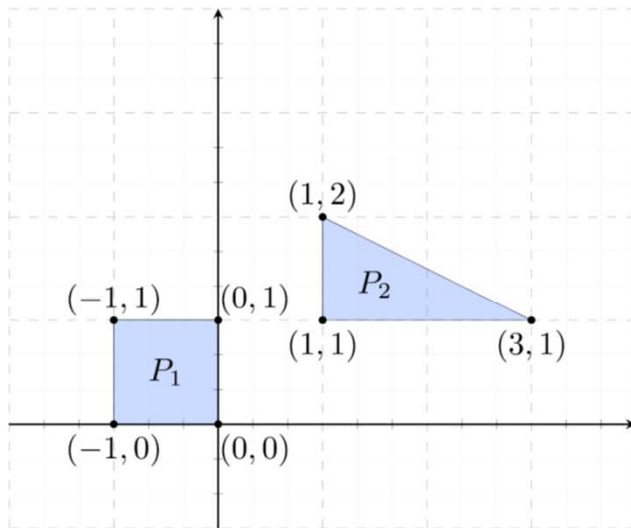


$$\text{ENewt}(p), p(x) = \max(3x + 1, x + 2, 0)$$

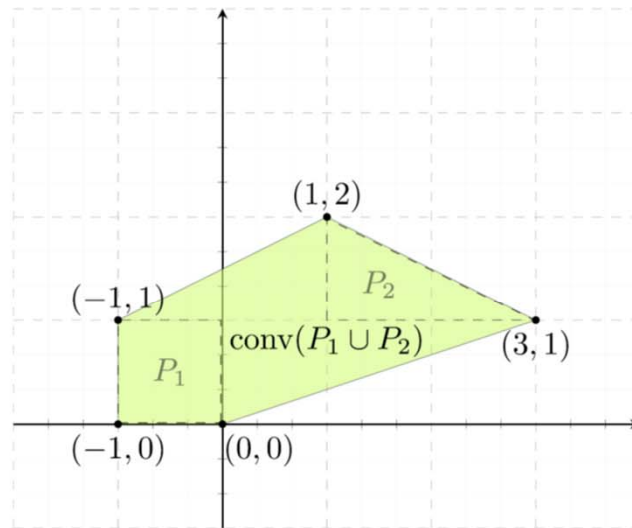
# Tropical Algebra of Max-plus Polynomials $\leftrightarrow$ Tropical Geometry of their Newton Polytopes

$$\text{Newt}(p_1 \vee p_2) = \text{conv}(\text{Newt}(p_1) \cup \text{Newt}(p_2))$$

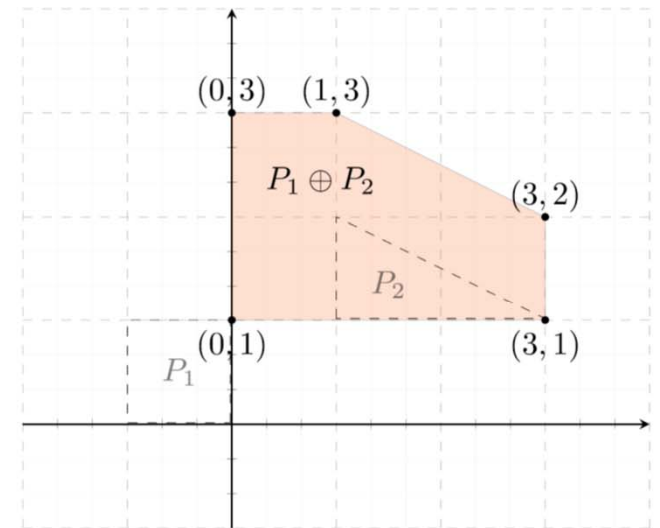
$$\text{Newt}(p_1 + p_2) = \text{Newt}(p_1) \oplus \text{Newt}(p_2)$$



(a)



(b)



(c)

Newton polytopes of (a) two max-polynomials

$$p_1(x,y) = \max(x+y, 3x+y, x+2y) \text{ and } p_2(x,y) = \max(0, -x, y, y-x),$$

(b) their  $\max(p_1, p_2)$ , and (c) their sum  $p_1 + p_2$

# Tropical Geometry of Neural Nets with Piecewise-Linear Activations

## References:

1. Charisopoulos, V., & Maragos, P. (2017, May). *Morphological perceptrons: geometry and training algorithms*, ISMM '17.
2. Charisopoulos, V., & Maragos, P. (2018). A Tropical Approach to Neural Networks with Piecewise Linear Activations. [arXiv:1805.08749](https://arxiv.org/abs/1805.08749).
3. Zhang, Liwen and Naitzat, Gregory and Lim, Lek-Heng. *Tropical geometry of deep neural networks*, Proc. ICML(35) 2018.

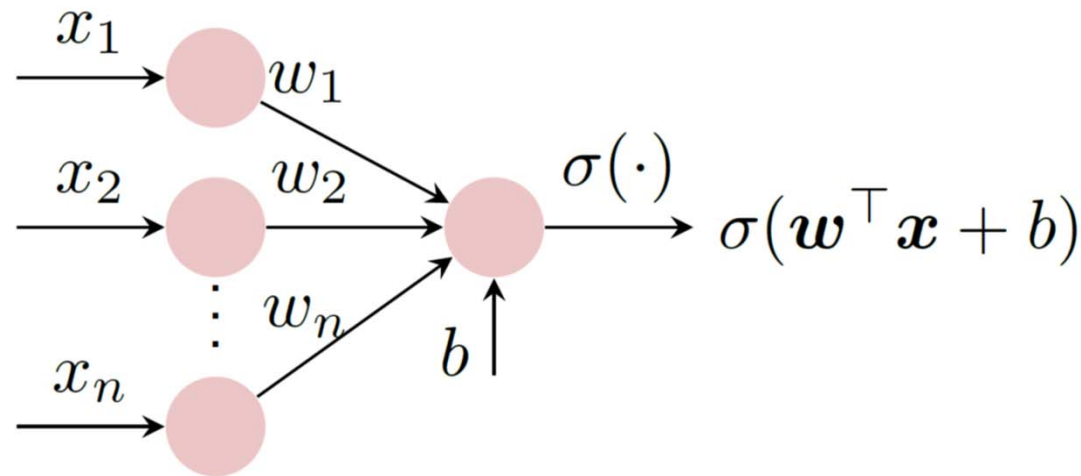


# NNs with PWL functions

---

Piecewise-linear functions used as *activation* functions  $\sigma$ :

1. **ReLU**:  $\max(0, v)$  or  $\max(\alpha v, v)$ ,  $\alpha \ll 1$  with  $v := \mathbf{w}^\top \mathbf{x} + b$
2. **Maxout**:  $\max_{k \in [K]} v_k$  with  $v_k := \mathbf{W}_k^\top \mathbf{x} + b_k$



**Linear regions:** maximally connected regions of input space on which the NN's output is linear [Montufar et al., 2014].



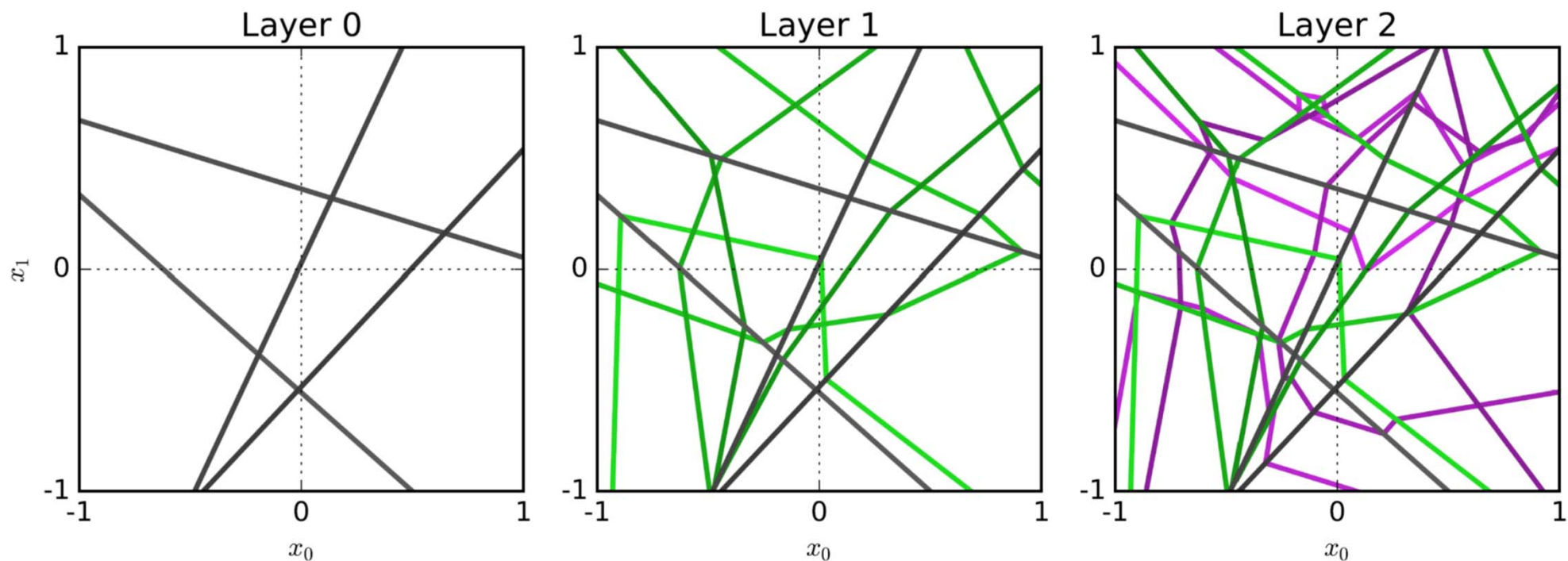


Figure: Input space is subdivided into convex polytopes, each of which is a “linear region” for the NN. Reproduced from [Raghu et al., 2016]

**Claim:** more linear regions  $\equiv$  more expressive power

# PWL functions and tropical geometry

---

Convex + PWL: ideal to study under lens of **tropical geometry**

Formally: *tropical semiring*  $(\mathbb{R} \cup \{-\infty\}, \vee, +)$

- binary “addition”  $x \vee y := \max(x, y)$ , “multiplication”  $x + y$
- operations on vectors  $\mathbf{x}, \mathbf{y}$ :

$$\mathbf{x} \vee \mathbf{y} := \begin{pmatrix} \max(x_1, y_1) \\ \vdots \\ \max(x_n, y_n) \end{pmatrix}, \quad \mathbf{x}^\top \boxplus \mathbf{y} := \bigvee_{i=1}^n x_i + y_i$$

Key object: tropical {poly, posy, sig}nomials

# Single neuron result

---

An application of the fundamental theorem of LP yields:

**Proposition** [Charisopoulos & Maragos, 2017]

The number of linear regions for a single maxout unit  $p(\mathbf{x}) = \max_{j \in [k]} \mathbf{w}_j^\top \mathbf{x} + b_i$  are equal to the number of vertices on the upper hull of  $\mathcal{N}(p)$

- subsumes **relu**
- all terms corresponding to interior vertices can be *removed* without affecting  $p(\mathbf{x})$  as a function.

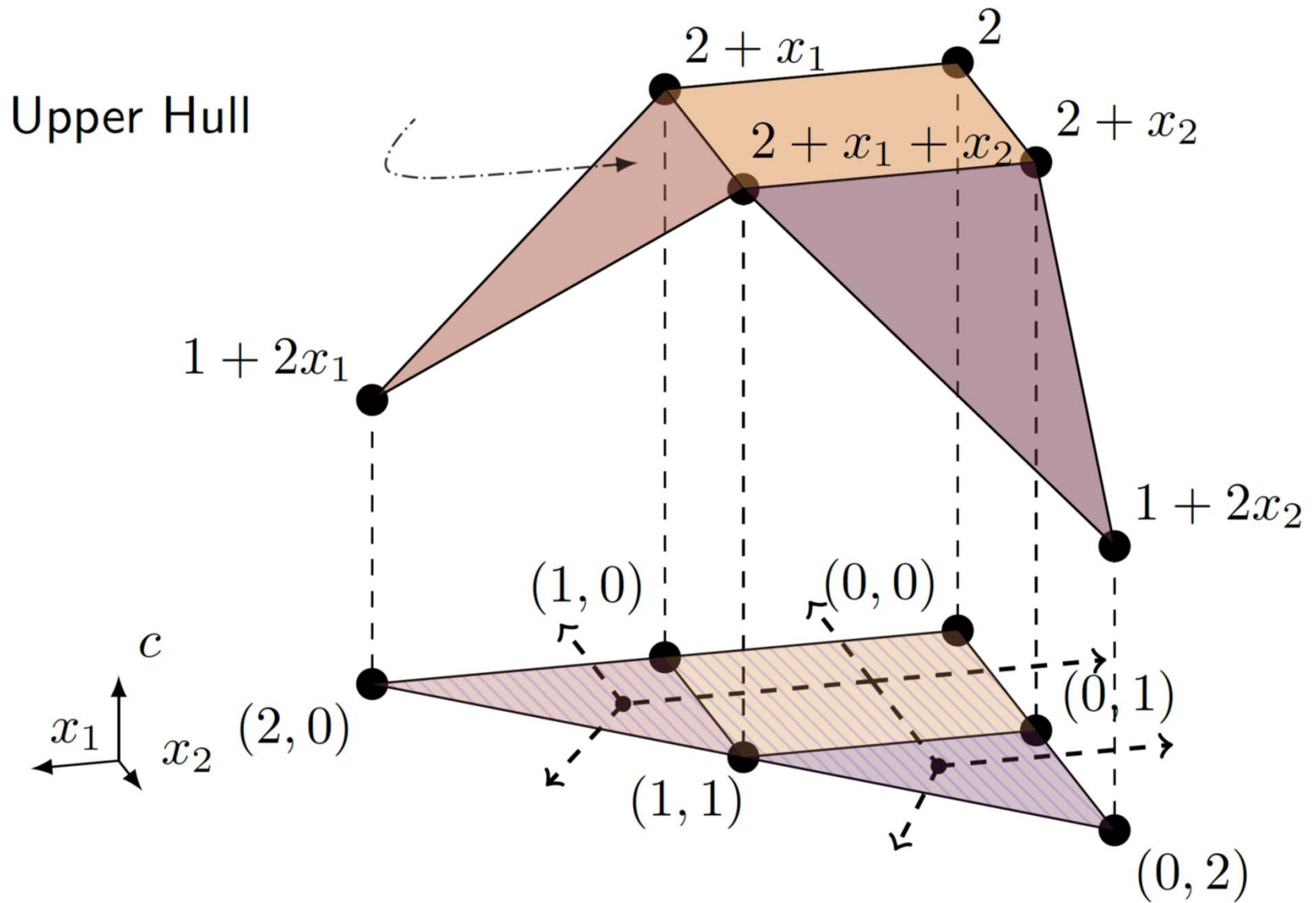


Figure: Upper Hull example for  
 $p(\mathbf{x}) = \max(1 + 2x_1, 2 + x_1, 2, 2 + x_2, 2 + x_1 + x_2)$

For a collection of tropical polynomials, suffices to work with Minkowski sums:

**Proposition** [Charisopoulos & Maragos, 2018] [Zhang et al., 2018]

The number of linear regions of a layer with  $n$  inputs and  $m$  neurons is upper bounded by the number of vertices in the upper convex hull of

$$\mathcal{N}(p_1) \oplus \cdots \oplus \mathcal{N}(p_m),$$

where  $\oplus$  denotes Minkowski sum.



# Main Result

---

Immediate application of a bound from [Gritzmann and Sturmfels, 1993] on faces of Minkowski sums gives

**Proposition** [Charisopoulos & Maragos, 2018]

The number of linear regions of  $n$  input,  $m$  output layer consisting of convex PWL activations of rank  $k$  is bounded above by

$$\min \left\{ k^m, 2 \sum_{j=0}^n \binom{m \frac{k(k-1)}{2}}{j} \right\}.$$

In case of ReLU, use symmetry of zonotopes to refine to

$$\min \left\{ 2^m, \sum_{j=0}^n \binom{m}{j} \right\}$$

# Counting in practice

---

**Goal:** given a network, count # of linear regions (exactly or approximately)

**Exact** counting using insight from Newton polytopes:

- ▷ vertex enumeration algorithm for Mink. sums [Fukuda, 2004]  $\Rightarrow$  requires solving  $\Omega(|\text{vert}(P)|)$  LPs.
- ▷ impractical unless problem is small

**MIP** representability of NNs [Serra et al., 2018]:

- ▷ Assumes bounded range of input space
- ▷ Requires enumerating solutions of MILPs

**Geometric Algorithm:** Randomized method for Sampling the Extreme Points of the Upper Hull of a Polytope [Charisopoulos & Maragos 2019, arXiv:1805.08749v2], [Maragos, Charisopoulos & Theodosis, Proc. IEEE 2021]

# Geometry & Algebra of NNs with PWL Activations

Theorem (Wang 2004): A continuous piecewise linear function is equal to the difference of two max-polynomials.

Theorem (Charisopoulos & Maragos 2018): The essential terms of a tropical polynomial are in bijection 1 – 1 with the vertices on the upper convex hull of its extended Newton polytope.

Theorem (Zhang et al. 2018): A neural network with ReLU-type activations can be represented as the difference of two max-polynomials, i.e. with a tropical rational function.



[Calafiore et al., 2019] use the Maslov dequantization to design universal approximators for convex (+loglog-convex) data

$$f \text{ convex} \Rightarrow f \simeq f_{\text{PWL}} \Leftrightarrow f \simeq f_T,$$

where  $f_{\text{PWL}} \leq f_T \leq T \log K + f_{\text{PWL}}$  and are given by

$$\left\{ \begin{array}{l} f_{\text{PWL}} := \max_{k \in [K]} \langle \mathbf{a}_k, \mathbf{x} \rangle + b_k, \\ f_T := T \log \left( \sum_{k=1}^K \exp \{b_k + \langle \mathbf{a}_k, \mathbf{x} \rangle\}^{1/T} \right) \end{array} \right.$$

In particular, fixing  $\varepsilon > 0$  and compact  $\mathcal{C}$ , a small enough  $T$  will satisfy

$$\sup_{\mathbf{x} \in \mathcal{C}} |f_T(\mathbf{x}) - f(\mathbf{x})| \leq \varepsilon.$$

# Morphological Networks: Geometry, Training, and Pruning

## References:

- V. Charisopoulos and P. Maragos, “[Morphological Perceptrons: Geometry and Training Algorithms](#)”, Proc. ISMM 2017, LNCS 10225, Springer.
- N. Dimitriadis and P. Maragos, “[Advances in Morphological Neural Networks: Training, Pruning and Enforcing Shape Constraints](#)”, Proc. ICASSP, 2021.

# Motivation

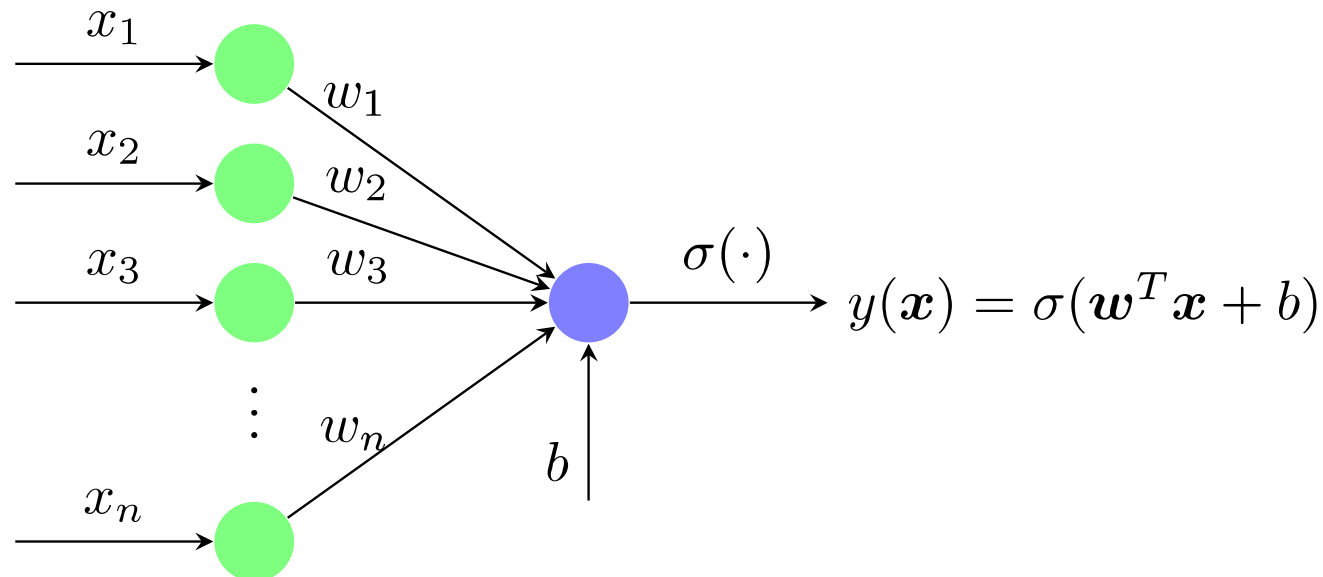
- Explosion of ML research in the last decade (now models with near-human or even human performance)
- Recent advances indicate shift towards nonlinearity, but...
- ...the “multiply-accumulate” (= linear) activations of the perceptron are still ubiquitous

## Our Questions:

- Are dot products and convolutions the only biologically plausible models of neuronal computation?
- Can we use results and tools from “nonlinear” mathematics to reason about complexity and dimension of learning models in current literature?

# Rosenblatt's perceptron

- Introduced in 1943, still prevalent neural model
- Activation:  $\phi(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$
- Nonlinearity at the output (e.g logistic sigmoid, ReLU):  
$$y(\mathbf{x}) = \sigma(\phi(\mathbf{x}))$$
- Multiply-accumulate architecture  $\rightarrow$  archetypal building block of all architectures (e.g. fully-connected, convolutional etc.)



# Max- plus Matrix Algebra

- vector/matrix ‘**addition**’ = pointwise max

$$\begin{aligned}\mathbf{x} \vee \mathbf{y} &= [x_1 \vee y_1, \dots, x_n \vee y_n]^T \\ \mathbf{A} \vee \mathbf{B} &= [a_{ij} \vee b_{ij}]\end{aligned}$$

- vector/matrix ‘**dual addition**’ = pointwise min

$$\begin{aligned}\mathbf{x} \wedge \mathbf{y} &= [x_1 \wedge y_1, \dots, x_n \wedge y_n]^T \\ \mathbf{A} \wedge \mathbf{B} &= [a_{ij} \wedge b_{ij}]\end{aligned}$$

- vector/matrix ‘**multiplication by scalar**’

$$\begin{aligned}c + \mathbf{x} &= [c + x_1, \dots, c + x_n]^T \\ c + \mathbf{A} &= [c + a_{ij}]\end{aligned}$$

- (max, +) ‘**matrix multiplication**’

$$[\mathbf{A} \boxplus \mathbf{B}]_{ij} = \bigvee_{k=1}^n a_{ik} + b_{kj}$$

- (min, +) ‘**matrix dual multiplication**’

$$[\mathbf{A} \boxplus' \mathbf{B}]_{ij} = \bigwedge_{k=1}^n a_{ik} + b_{kj}$$

# Morphological Operators on Lattices

( $\leq$  = partial ordering,  $\vee$  = supremum,  $\wedge$  = infimum)

---

- $\psi$  is **increasing** iff  $f \leq g \Rightarrow \psi(f) \leq \psi(g)$ .
- $\delta$  is **dilation** iff  $\delta(\vee_i f_i) = \vee_i \delta(f_i)$ .
- $\varepsilon$  is **erosion** iff  $\varepsilon(\wedge_i f_i) = \wedge_i \varepsilon(f_i)$ .
- $\alpha$  is **opening** iff increasing and antiextensive ( $\alpha(f) \leq f$ ),  
and idempotent ( $\alpha = \alpha^2$ ).
- $\beta$  is **closing** iff increasing and extensive ( $\beta(f) \geq f$ ),  
and idempotent ( $\beta = \beta^2$ ).
- $(\varepsilon, \delta)$  is **adjunction** iff  $\delta(g) \leq f \Leftrightarrow g \leq \varepsilon(f)$ .

(Galois connection)  
Residuation pair  
("Tropical Adjoints")

Then:  $\varepsilon$  is erosion,  $\delta$  is dilation,

$\delta\varepsilon$  is opening (projection),  $\varepsilon\delta$  is closing (projection).

# Solve Max-plus Equations

- **Problems:**

(1) Exact problem: Solve  $\delta_A(\mathbf{x}) = \mathbf{A} \boxplus \mathbf{x} = \mathbf{b}$ ,  $\mathbf{A} \in \overline{\mathbb{R}}^{m \times n}$ ,  $\mathbf{b} \in \overline{\mathbb{R}}^m$

(2) Approximate Constrained: Min  $\|\mathbf{A} \boxplus \mathbf{x} - \mathbf{b}\|_{p=1 \dots \infty}$  s.t.  $\mathbf{A} \boxplus \mathbf{x} \leq \mathbf{b}$

- **Theorem:** (a) The **greatest (sub)solution** of (1) and unique solution of (2) is

$$\hat{\mathbf{x}} = \varepsilon_A(\mathbf{b}) = \mathbf{A}^* \boxplus' \mathbf{b} = [\bigwedge_i b_i - a_{ij}], \quad \mathbf{A}^* \triangleq -\mathbf{A}^T$$

and yields the **Greatest Lower Estimate (GLE)** of data  $\mathbf{b}$ :

$$\delta_A(\varepsilon_A(\mathbf{b})) = \mathbf{A} \boxplus (\mathbf{A}^* \boxplus' \mathbf{b}) \leq \mathbf{b}$$

- (b) **Min Max Absolute Error (MMAE) unconstrained unique solution:**

$$\tilde{\mathbf{x}} = \hat{\mathbf{x}} + \mu, \quad \mu = \|\mathbf{A} \boxplus \hat{\mathbf{x}} - \mathbf{b}\|_{\infty} / 2$$

- **Geometry:** Operators  $\delta, \varepsilon$  are vector dilation and erosion, and the **GLE**  $\mathbf{b} \mapsto \delta\varepsilon(\mathbf{b})$  is an opening (**lattice projection**).

- **Complexity:**  $O(mn)$

# Morphological Perceptron

- Introduced in the 1990's. Instead of multiply-accumulate, computes a **dilation** (max-of-sums):

$$\tau(\mathbf{x}) = \mathbf{w}^T \boxplus \mathbf{x} \triangleq \bigvee_{i=1}^n w_i + x_i$$

or an **erosion**:

$$\tau'(\mathbf{x}) = \mathbf{w}^T \boxminus \mathbf{x} \triangleq \bigwedge_{i=1}^n w_i + x_i$$

- Ritter & Urcid (2003): argued about biological plausibility and proved that every compact region in  $n$ -dim Euclidean space can be approximated by morphological perceptrons to arbitrary accuracy.
- Related to a Maxout unit.



# Feasible Regions & Separability Condition for Max-plus Perceptron

Let  $\mathbf{X} \in \mathbb{R}_{\max}^{k \times n}$  be a matrix containing the patterns to be classified as its rows, let  $\mathbf{x}^{(k)}$  denote the  $k$ -th pattern (row) and let  $\mathcal{C}_1, \mathcal{C}_0$  be the two classes

**Max-plus perceptron**  $\tau(\mathbf{x}) = \mathbf{w}^T \boxplus \mathbf{x}$

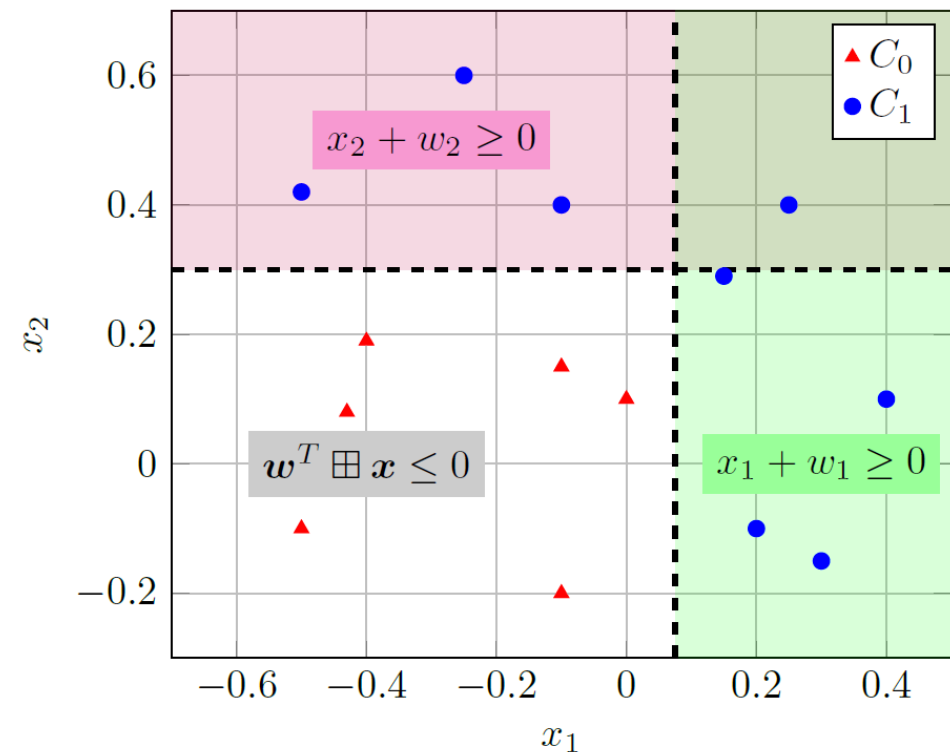
$$\tau(\mathbf{x}) = w_0 \vee (w_1 + x_1) \vee \cdots \vee (w_n + x_n) = w_0 \vee \left( \bigvee_{i=1}^n w_i + x_i \right)$$

**Feasible Region = Tropical Polyhedron**

$$\mathcal{T}(\mathbf{X}_{\text{pos}}, \mathbf{X}_{\text{neg}}) = \{ \mathbf{w} \in \mathbb{R}_{\max}^n : \mathbf{X}_{\text{pos}} \boxplus \mathbf{w} \geq 0, \mathbf{X}_{\text{neg}} \boxplus \mathbf{w} \leq 0 \}$$

**Separability Condition, equivalent to Nonempty Trop. Polyhedron**

$$\mathbf{X}_{\text{pos}} \boxplus (\mathbf{X}_{\text{neg}}^* \boxplus' \mathbf{0}) \geq \mathbf{0}$$



# Morphological Neural Nets (MNNs) and Training Approaches

- **Constructive Algorithms**

Dendrite Learning [Ritter & Urcid, 2003], Iterative Partitioning / Competitive Learning [Sussner & Esmi, 2011]: combine (max, +) and (min, +) classifiers, build “bounding boxes” around patterns

- "perfect" fit to data, no concept of outlier

- **Morphological Associative Memories**

Introduce a Hopfield-type network, computing (noniteratively) a morphological/fuzzy response (e.g. Sussner & Valle, 2006):

- **Gradient Descent Variants**

Min-max classifiers [Yang & Maragos, 1995], MRL nodes [Pessoa & Maragos, 2000], Dilation-Erosion Linear Perceptron [Araujo et al. 2012].

- **Recent Approaches:**

Convex-Concave Programming (CCP) for Max-plus Perceptron and DEP (Binary Classification) [Charisopoulos & Maragos 2017 ]

Reduced Dilation-Erosion Perceptron (r-DEP) trained via CCP for Binary Classification [Valle 2020]

Dense Morphological Networks [Mondal et al. 2019]

Deep Morphological Networks [Franchi et al. 2020]

r-DEP for Multiclass Classification via CCP, L1 Pruning on Dense MNNs [Dimitriadis & Maragos 2021]

# Our Approach for Training MP on Non-separable Data

Training a (max, +) perceptron can be stated as a difference-of-convex (DC) optimization problem. Solved iteratively (but global optimum not guaranteed) by the Convex-Concave Procedure (**CCP**) [Yuille & Rangarajan, 2003], implemented via DCCP [Shen et al. 2016]

Given a sequence of training data  $\{\mathbf{x}^k\}_{k=1}^K$  :

$$\begin{aligned} \text{Minimize } J(\mathbf{X}, \mathbf{w}, \boldsymbol{\nu}) &= \sum_{k=1}^K \nu_k \cdot \max(\xi_k, 0) \\ \text{s. t. } &\begin{cases} \bigvee_{i=1}^n w_i + x_i^{(k)} \leq \xi_k & \text{if } \mathbf{x}^{(k)} \in \mathcal{C}_0 & \text{Negative} \\ \bigvee_{i=1}^n w_i + x_i^{(k)} \geq -\xi_k & \text{if } \mathbf{x}^{(k)} \in \mathcal{C}_1 & \text{Positive} \end{cases} \end{aligned}$$

$\nu_k$  Some measure of "being outlier"

$\xi_k$  Positive only if misclassification occurs at  $k$ -th pattern

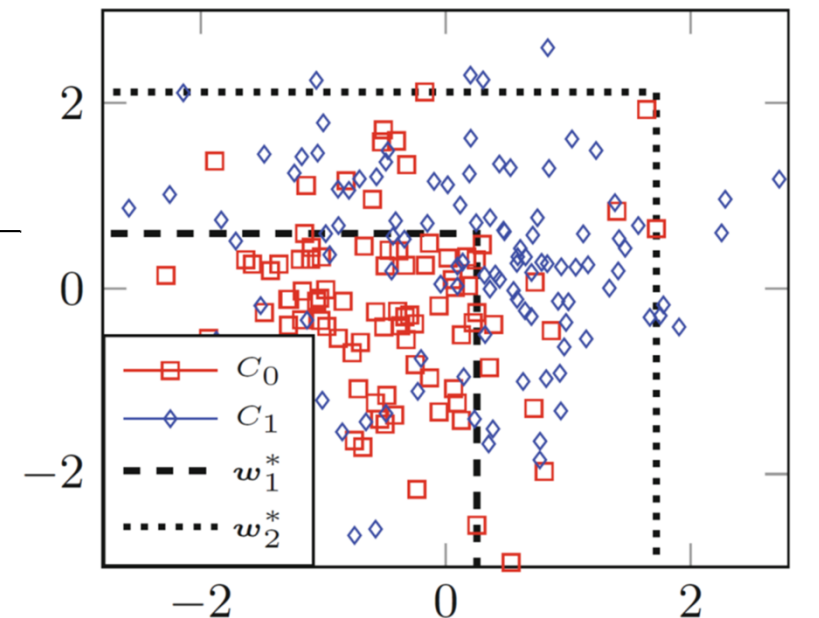
# Gradient Descent vs. CCP for Training (max,+ ) Perceptron

## Two Binary Classification Experiments.

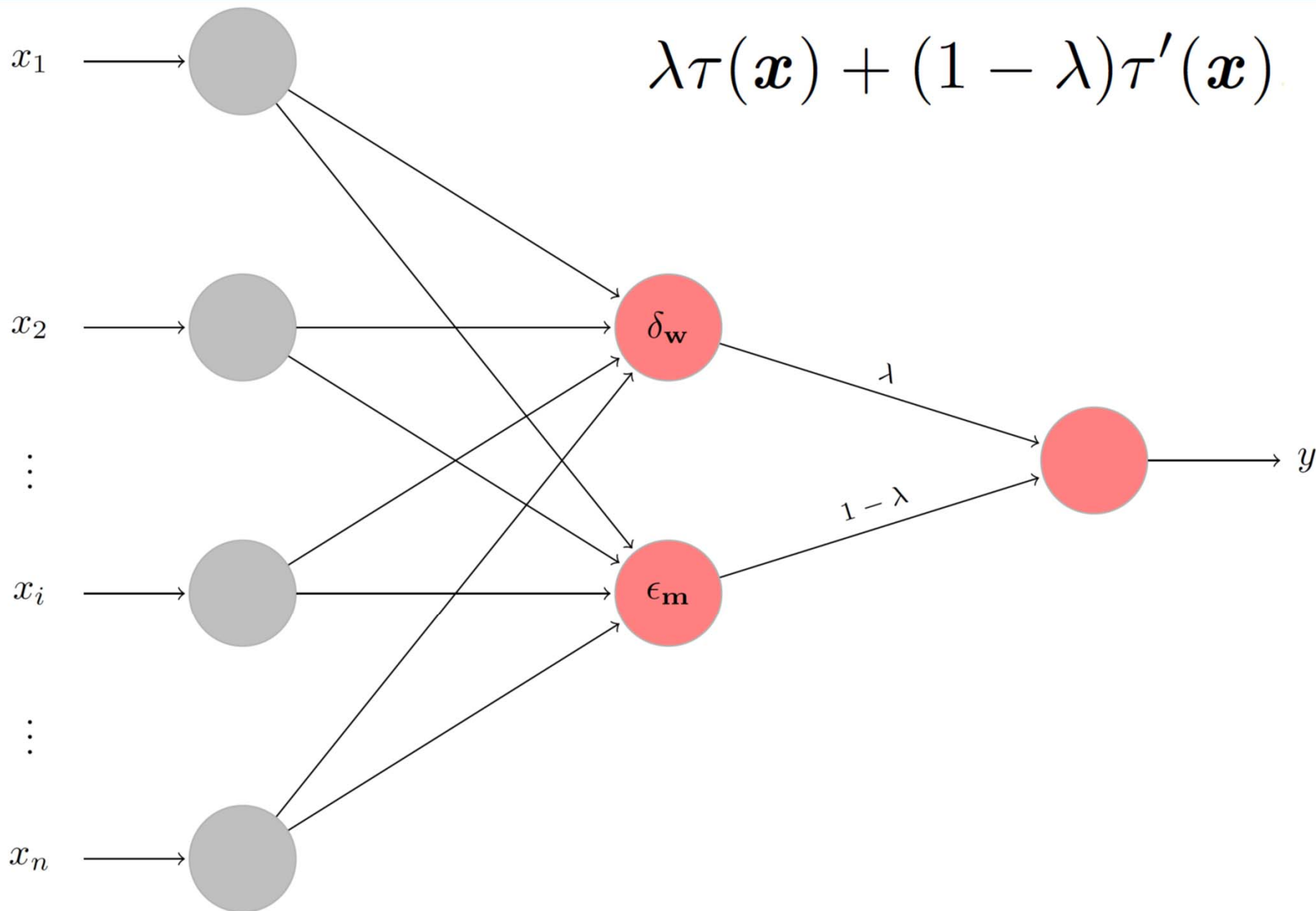
Gradient descent with fixed  $N = 100$  epochs vs. CCP using the DCCP toolkit for CvxPy (default parameters).

$\eta$	Ripleys		WDBC	
	SGD	WDCCP	SGD	WDCCP
0.01	$0.838 \pm 0.011$	<b>0.902</b> $\pm 0.001$	$0.726 \pm 0.002$	<b>0.908</b> $\pm 0.001$
0.02	$0.739 \pm 0.012$		$0.763 \pm 0.006$	
0.03	$0.827 \pm 0.008$		$0.726 \pm 0.004$	
0.04	$0.834 \pm 0.008$		$0.751 \pm 0.007$	
0.05	$0.800 \pm 0.009$		$0.783 \pm 0.012$	
0.06	$0.785 \pm 0.008$		$0.768 \pm 0.01$	
0.07	$0.776 \pm 0.009$		$0.729 \pm 0.009$	
0.08	$0.769 \pm 0.01$		$0.732 \pm 0.01$	
0.09	$0.799 \pm 0.009$		$0.730 \pm 0.015$	
0.1	$0.749 \pm 0.011$		$0.729 \pm 0.009$	

**CCP:** more robust results



# Dilation-Erosion Perceptron (DEP)



$$y = f(\mathbf{x}) = \lambda\delta_w(\mathbf{x}) + (1 - \lambda)\epsilon_m(\mathbf{x})$$

# Dilation-Erosion Perceptron Training

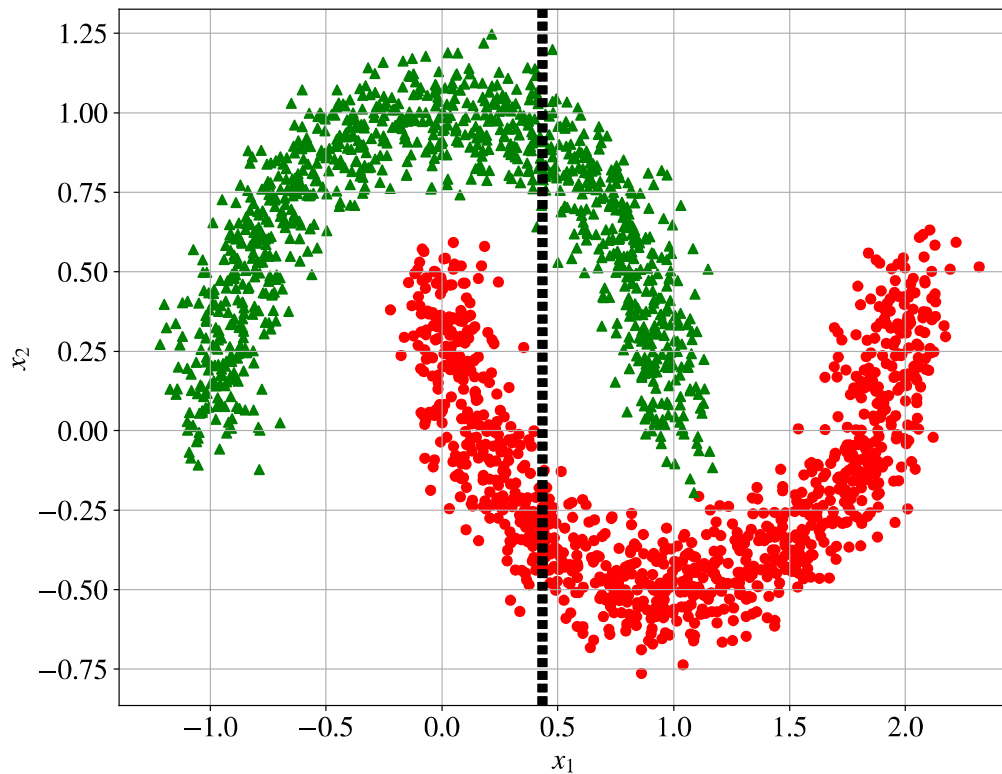
$$\begin{aligned}y = f(\mathbf{x}) &= \lambda\delta_w(\mathbf{x}) + (1 - \lambda)\epsilon_m(\mathbf{x}) = \lambda\delta_w(\mathbf{x}) - (1 - \lambda)[- \epsilon_m(\mathbf{x})] \\ &= \text{convex} - (-\text{concave}) \\ &= \text{convex} - \text{convex}\end{aligned}$$

Training as Difference-of-Convex Optimization via Convex-Concave Programming

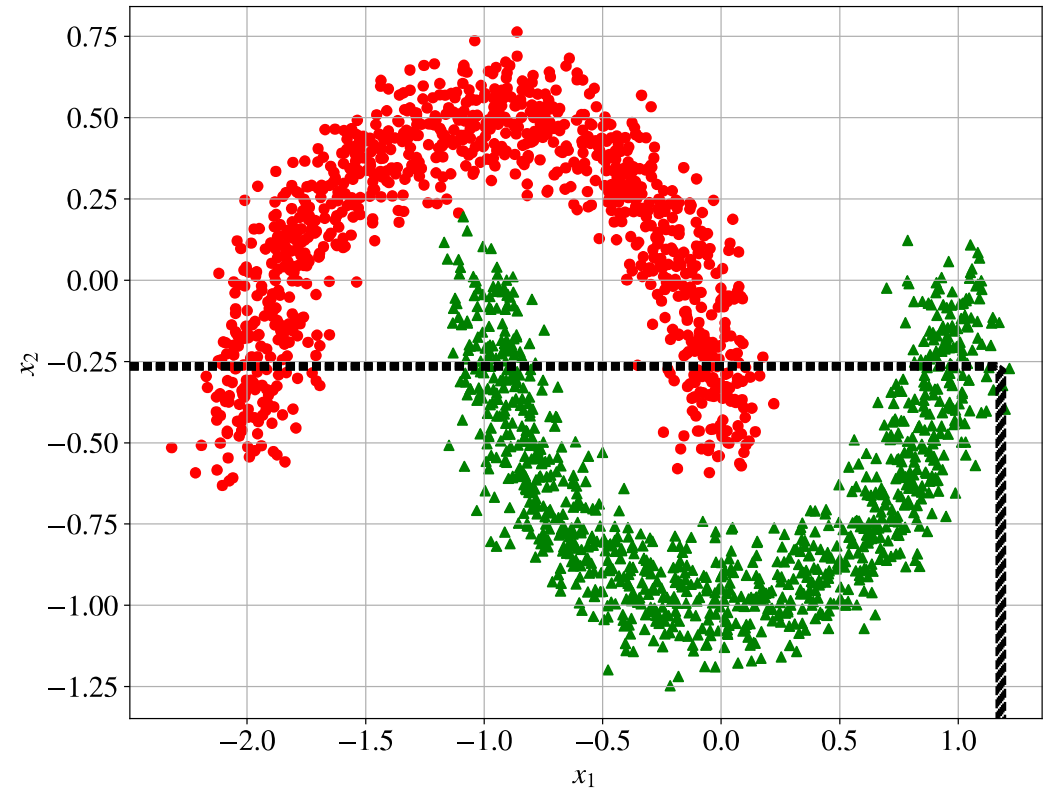
$$\begin{aligned}\text{minimize} \quad & \sum_{i=1}^N v_i \max\{0, \xi_i\} \\ \text{subject to} \quad & \lambda\delta_w(\mathbf{x}_i) + (1 - \lambda)\epsilon_m(\mathbf{x}_i) \geq -\xi_i \quad \forall \mathbf{x}_i \in \mathcal{P}, \\ & \lambda\delta_w(\mathbf{x}_i) + (1 - \lambda)\epsilon_m(\mathbf{x}_i) \leq +\xi_i \quad \forall \mathbf{x}_i \in \mathcal{N}\end{aligned}$$

# Effect of $\mathcal{N} \Leftrightarrow \mathcal{P}$ and Ordering Vector Data

## Double Moons example



Reversed labels



Correct labels

Reduced ordering [Valle 2020] for better ordering feature patterns:

Let  $V$  be a nonempty set,  $\mathcal{L}$  be a complete lattice and  $\rho: V \rightarrow \mathcal{L}$  be a surjective mapping.

A reduced ordering is defined as:  $x \leq_{\rho} y \Leftrightarrow \rho(x) \leq \rho(y) \forall x, y \in V$ .

Can be obtained via a supervised training on a set of positive and negative examples.

# Experiments: Multiclass r-DEP, CCP training

	MNIST	FashionMNIST
$n = 5$	<b>97.72 <math>\pm</math> 0.01</b>	<b>88.21 <math>\pm</math> 0.01</b>
$n = 10$	<b>97.72 <math>\pm</math> 0.01</b>	88.07 $\pm$ 0.01
$n = 15$	97.67 $\pm$ 0.01	88.11 $\pm$ 0.01
$n = 20$	97.64 $\pm$ 0.01	88.12 $\pm$ 0.01

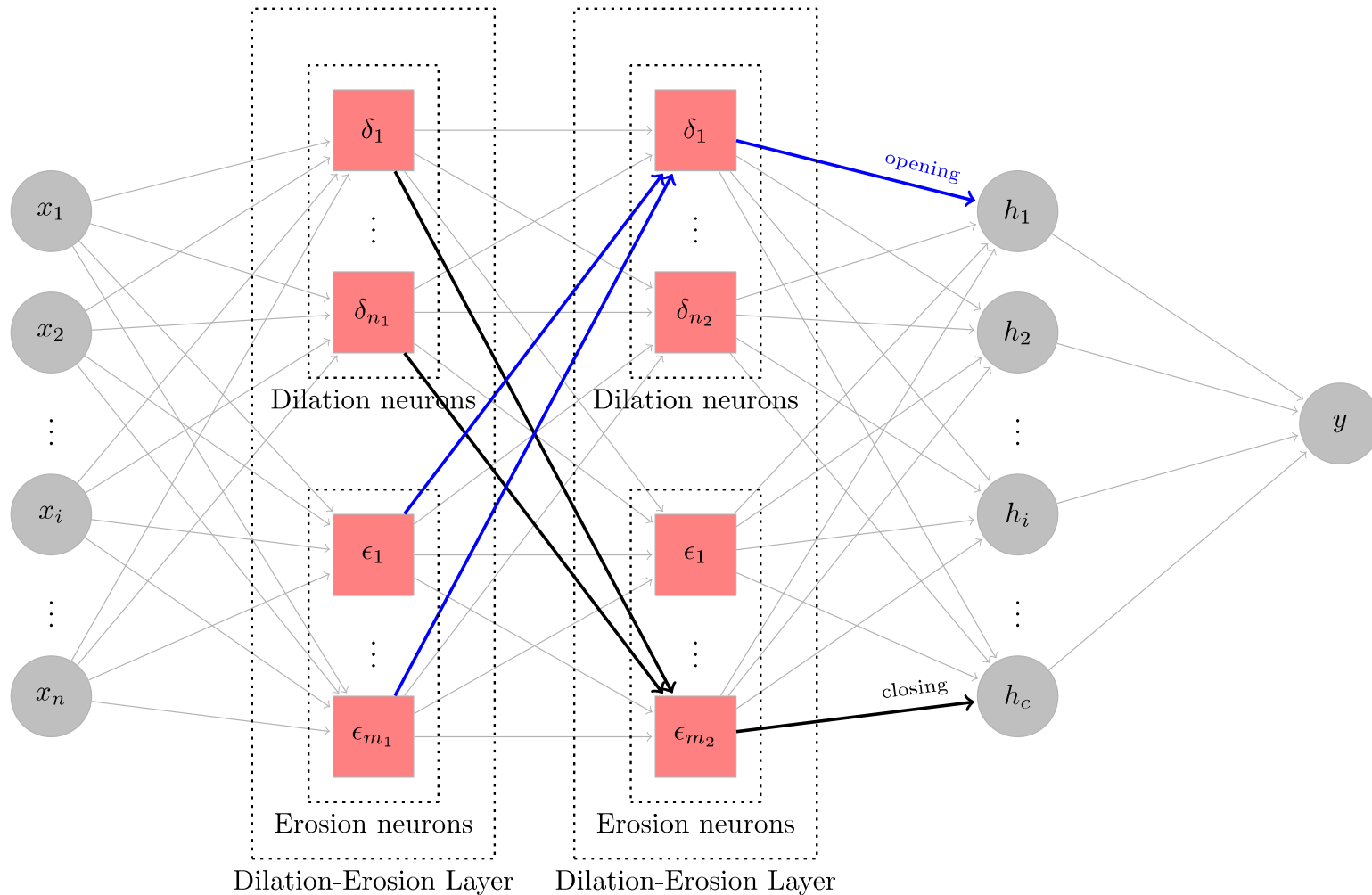
Table: Results of Bagging *multiclass* r-DEP with  $n$  RBF kernels.

- Performance similar to MLP-ReLU architectures trained via SGD
- CCP training is more robust

[Dimitriadis & Maragos 2021]



# Dense Morphological Networks



Dense Morphological Network with 2 hidden layers [similar to Mondal et al. 2019]

**Focus on Sparsity** [Dimitriadis & Maragos 2021]  $\rightarrow$  Apply  $\ell_1$  Pruning

# Experiments: Pruning Dense MNN vs MLP-ReLU

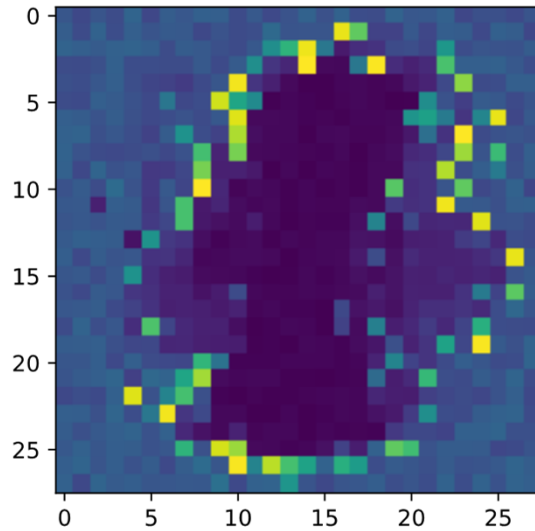
		Adaptive Momentum Estimation				Stochastic Gradient Descent			
		$\delta$	$\epsilon$	$(\delta, \epsilon)$	FF-ReLU	$\delta$	$\epsilon$	$(\delta, \epsilon)$	FF-ReLU
	$p$								
MNIST	100%	97.62	96.17	97.95	98.13	94.86	93.36	96.07	98.16
	75%	97.62	96.18	97.93	98.15	94.86	93.36	96.07	98.12
	50%	97.62	96.22	97.90	98.17	94.86	93.37	96.07	98.08
	25%	97.62	96.09	97.87	97.51	94.86	93.40	96.06	98.01
	10%	97.62	95.78	97.74	93.38	94.86	93.38	96.09	96.67
	7.5%	97.62	95.42	97.76	90.17	94.86	93.38	96.10	95.56
	5%	97.62	94.51	97.66	83.39	94.86	93.40	96.10	92.96
	2.5%	97.62	93.43	97.37	68.93	94.86	93.39	96.09	80.48
	1%	97.62	91.17	97.08	44.22	94.86	93.38	96.08	58.07
FashionMNIST	100%	86.31	86.82	88.32	88.82	82.06	85.23	86.21	87.79
	75%	86.30	86.81	88.30	88.88	82.00	85.23	86.21	87.75
	50%	86.22	86.80	88.33	88.18	82.05	85.25	86.20	87.19
	25%	85.95	86.85	88.31	82.15	81.90	85.26	86.28	84.35
	10%	85.58	86.27	88.05	65.89	81.67	85.27	86.23	73.22
	7.5%	85.47	86.15	87.99	57.93	81.63	85.27	86.21	63.95
	5%	85.37	85.81	87.76	49.12	81.52	85.24	86.22	47.73
	2.5%	84.91	85.47	87.56	42.48	81.14	85.26	86.22	38.84
	1%	81.14	84.86	86.85	28.13	80.68	85.27	86.18	35.46

**Table:** Accuracy of pruned networks on the MNIST and FashionMNIST datasets.

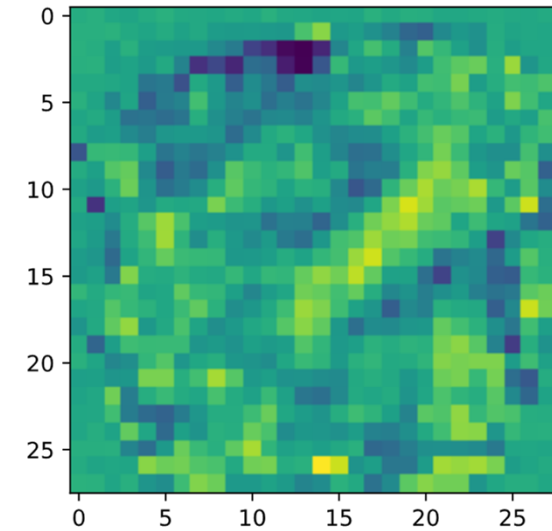
Models:  $\delta$   $\rightarrow$  only dilation neurons,  $\epsilon$   $\rightarrow$  only erosion,  $(\delta, \epsilon)$   $\rightarrow$  split equally, FF-ReLU  $\rightarrow$  FeedForward NN with ReLU.

shades of red showcase the degree of (severe) deterioration in accuracy green indicates the absence of performance loss

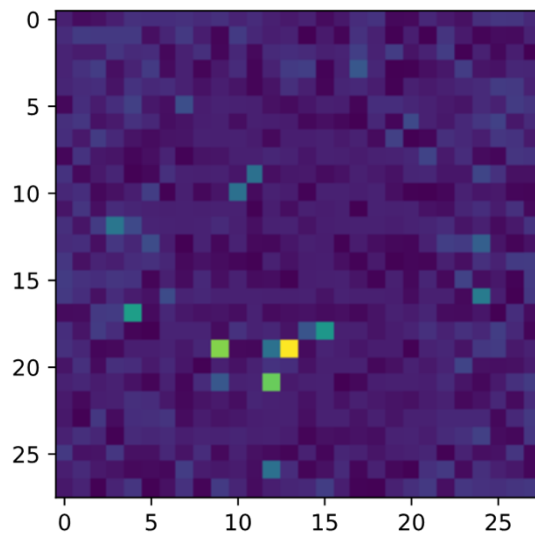
# Qualitative Perspectives on Sparsity



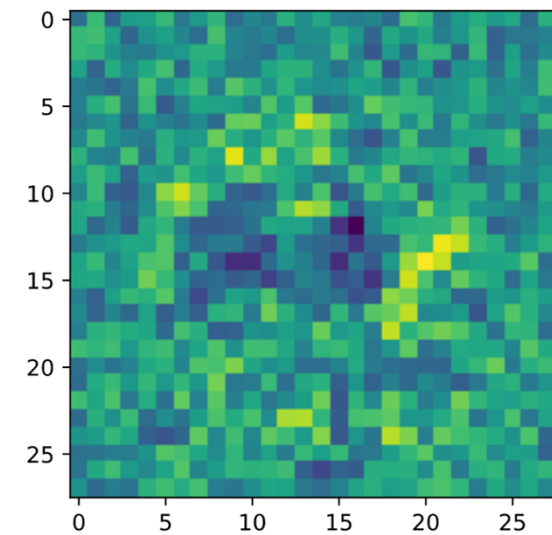
$(\delta, \epsilon) - Adam$



FF-ReLU - *Adam*



$(\delta, \epsilon) - SGD$



FF-ReLU - *SGD*

Examples of hidden layer activations for various NN models (MNIST dataset)

# Minimization of Neural Nets via Tropical Division and Newton Polytope Approximation

## References:

- G. Smyrnis, P. Maragos and G. Retsinas, “*MaxPolynomial Division With Application to Neural Network Simplification*”, Proc. ICASSP, 2020.
- G. Smyrnis and P. Maragos, “*Multiclass Neural Network Minimization Via Tropical Newton Polytope Approximation*”, Proc. ICML 2020.
- P. Misiakos, G. Smyrnis, G. Retsinas and P. Maragos, “*Neural Network Approximation based on Hausdorff distance of Tropical Zonotopes*”, Proc. ICLR 2022.

# Tropical Polynomials

**Tropical Semiring**  $(\mathbb{R}_{\max}, \vee, +)$

$$\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}$$


$$a \vee b = \max(a, b)$$

$$a + b = a + b$$

**Tropical Polynomials**

$$f(\mathbf{x}) = \max_{i \in [n]} \{ \mathbf{a}_i^T \mathbf{x} + b_i \}$$

*Real coefficients*



# Newton Polytopes

## Newton Polytopes

$$\text{Newt}(f) = \text{conv} \{ \mathbf{a}_i : i \in [n] \}$$

$$\text{ENewt}(f) = \text{conv} \{ (\mathbf{a}_i, b_i) : i \in [n] \}$$

## Polytope computation

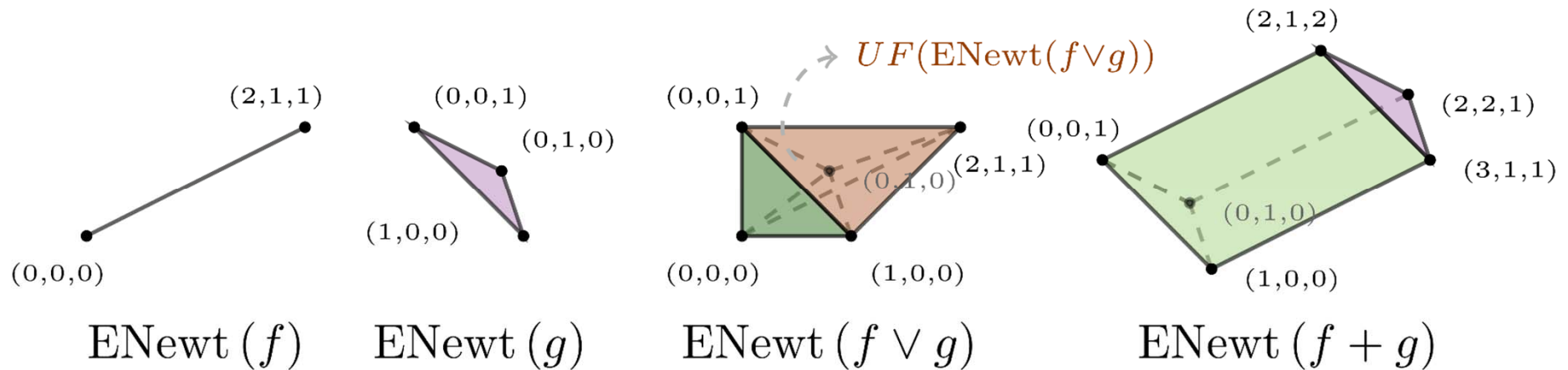
$$\text{ENewt}(f \vee g) = \text{conv} \{ \text{ENewt}(f) \cup \text{ENewt}(g) \}$$

$$\text{ENewt}(f + g) = \text{ENewt}(f) \oplus \text{ENewt}(g)$$

# Example: Polytope Computation

$$f(x, y) = \max(2x + y + 1, 0)$$

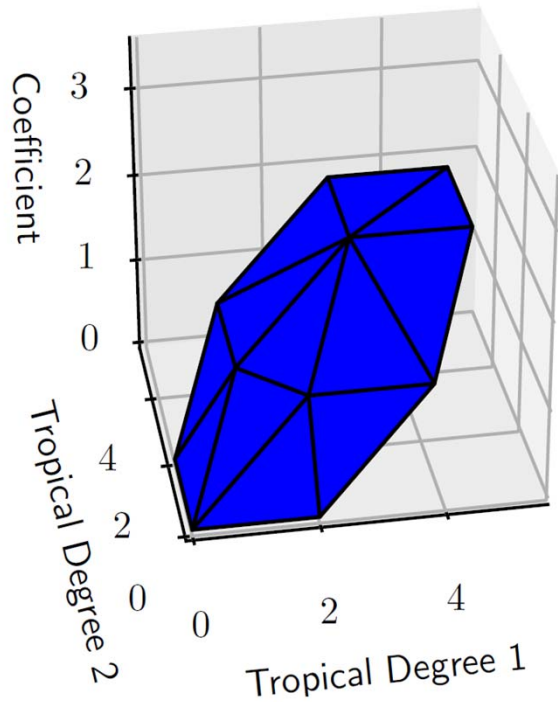
$$g(x, y) = \max(x, y, 1)$$



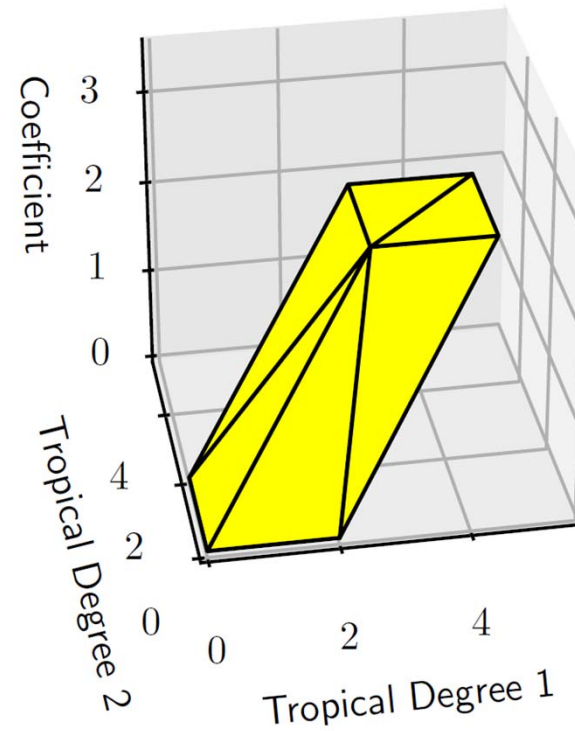
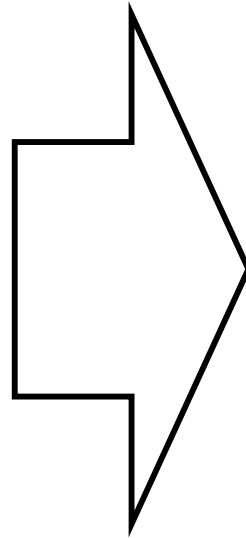
$$f \vee g = \max(2x + y + 1, 0, x, y, 1)$$

$$f + g = \max(x, y, 1, 3x + y + 1, 2x + 2y + 1, 2x + y + 2)$$

# General idea for Geometric NN Minimization



*Original Network Polytope*



*Approximate Network Polytope*



# Maxpolynomial Division

Problem: Assume we have two maxpolynomials  $p(\mathbf{x})$ ,  $d(\mathbf{x})$  (dividend and divisor). We want to find two maxpolynomials  $q(\mathbf{x})$ ,  $r(\mathbf{x})$  (quotient and remainder) such that:

$$p(\mathbf{x}) = \max(q(\mathbf{x}) + d(\mathbf{x}), r(\mathbf{x}))$$

**However!** The above is not always feasible (non-trivially).

**Approximate Division**: We relax the requirements, so that the polynomials we want to find satisfy:

$$p(\mathbf{x}) \geq \max(q(\mathbf{x}) + d(\mathbf{x}), r(\mathbf{x}))$$

We also require that  $q(\mathbf{x})$ ,  $r(\mathbf{x})$  satisfy the above maximally.

# Algorithm for Approximate Maxpolynomial Division

1. Let  $\mathcal{C}$  be the set of possible vectors  $\mathbf{c}$  by which we can h-shift  $\text{Newt}(d)$  (each of which corresponds to a linear term in  $q$ ).
2. We raise the shifted version of  $\text{ENewt}(d)$  as high as possible so that it still lies below  $\text{ENewt}(p)$ , and we mark the vertical shift as  $q_c$ .

3. We set the quotient equal to:

$$q(\mathbf{x}) = \max_{\mathbf{c} \in \mathcal{C}} (q_c + \mathbf{c}^T \mathbf{x})$$

and add all terms not covered by a h-shift  $\mathbf{c}$  to the remainder  $r(\mathbf{x})$ .

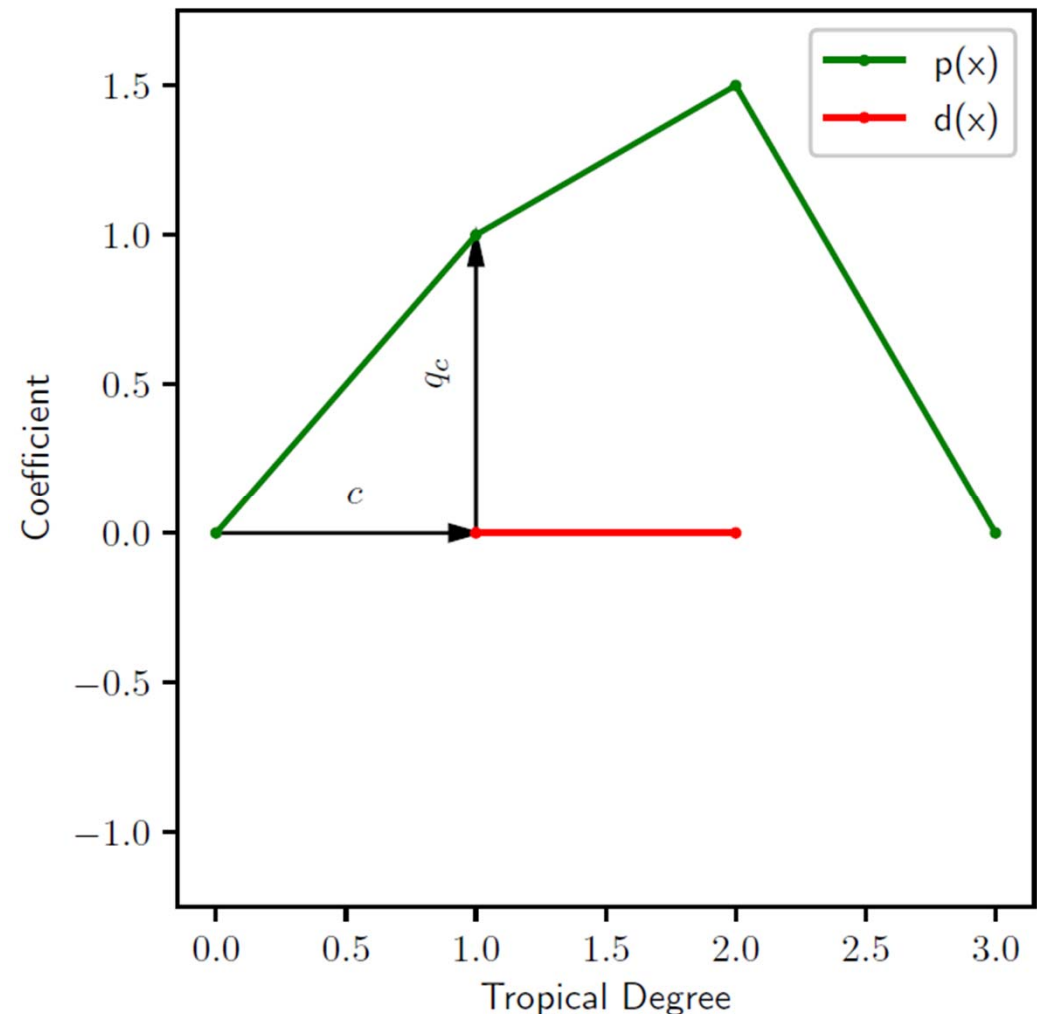


Figure: [Division Method](#)

Division of  $p(x) = \max(3x, 2x + 1.5, x + 1, 0)$   
by  $d(x) = \max(x, 0)$ .

# Division Example (1)

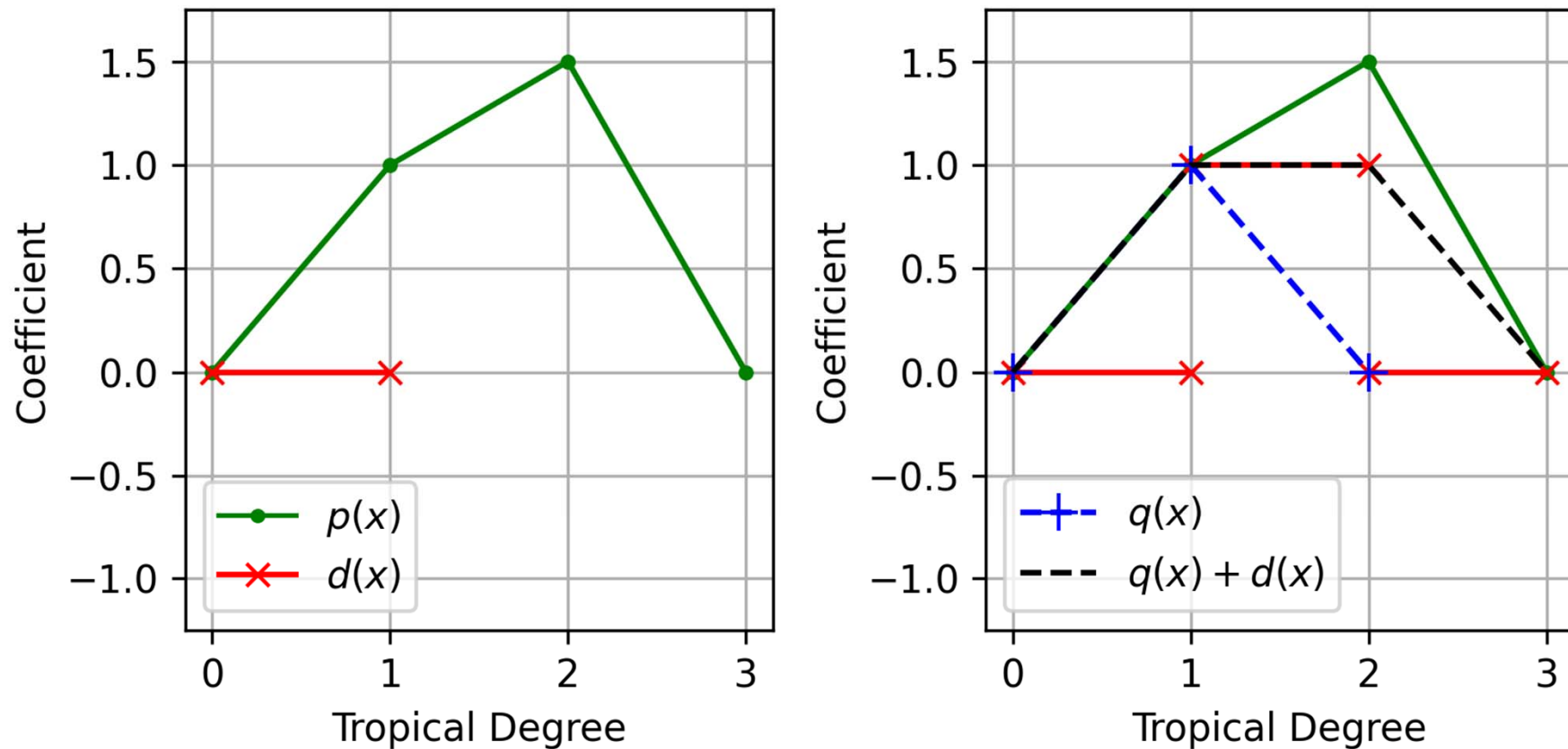


Figure: Division of  $p(x) = \max(3x, 2x + 1.5, x + 1, 0)$  by  $d(x) = \max(x, 0)$ .

Note: The Newton Polytope of the divisor is raised as much as possible, but it cannot match the polytope of the dividend exactly. Thus, only 3 out of the 4 vertices are perfectly matched.

# Division Example (2)

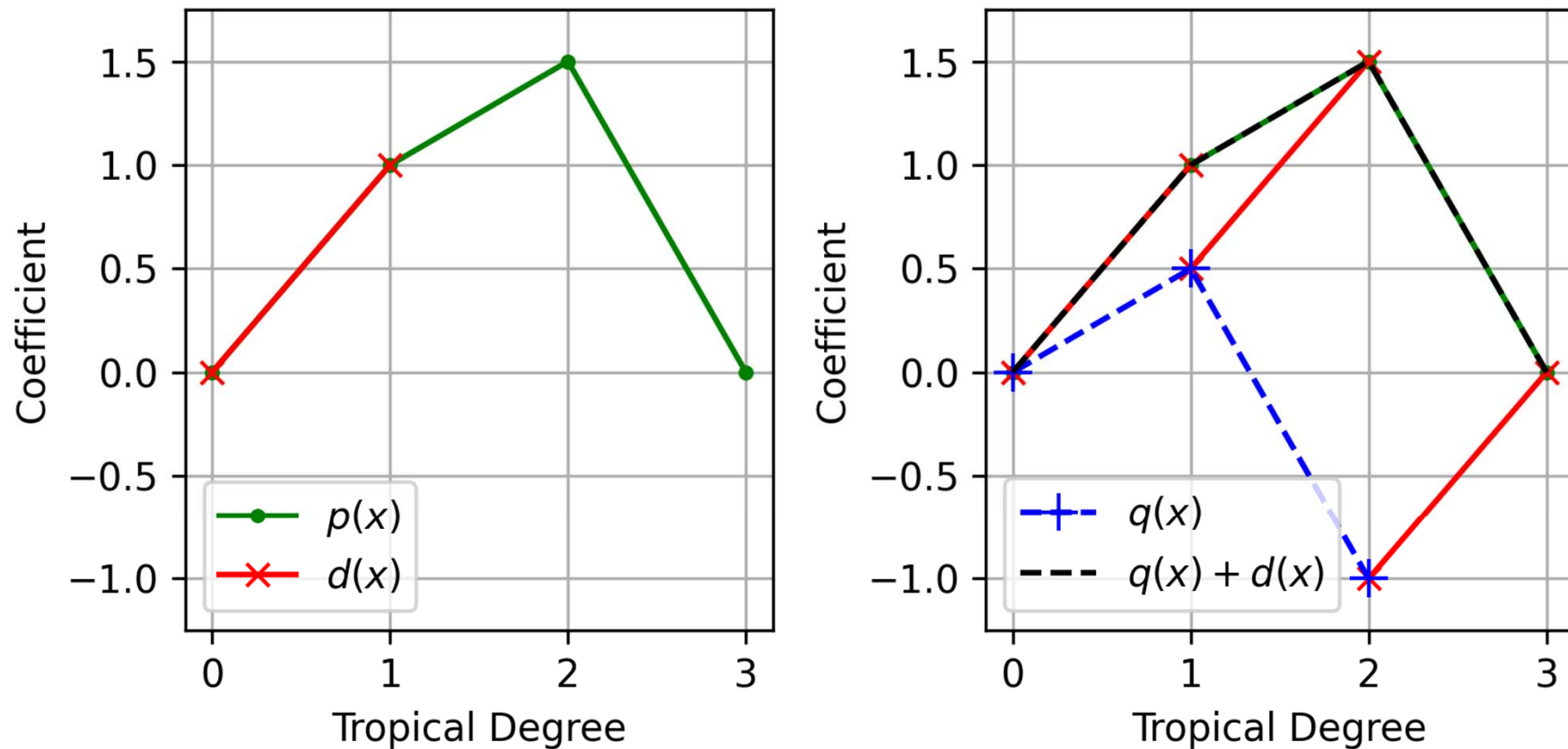


Figure: Division of  $p(x) = \max(3x, 2x + 1.5, x + 1, 0)$  by  $d(x) = \max(x + 1, 0)$ .

Note: In this case, the polytope of the divisor can match that of the dividend perfectly, so all vertices are covered.

# Application to Neural Network Minimization

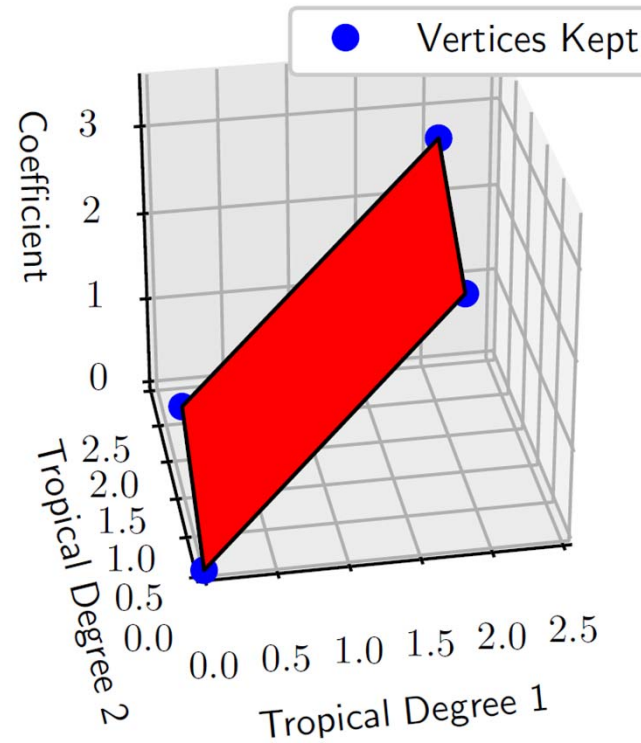
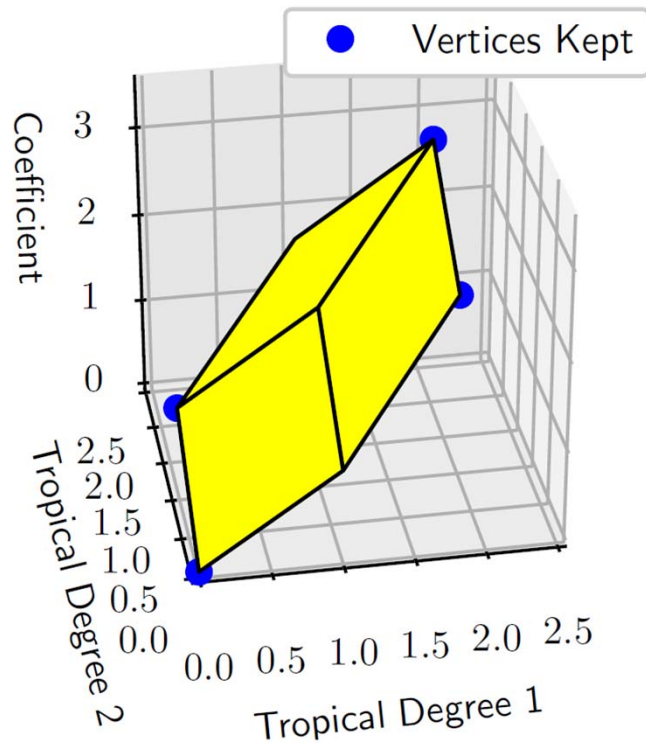
**General idea:** Our algorithm seeks to minimize the network by matching the most important vertices of the Newton Polytopes of its maxpolynomials.

## 2-layer 1-output NN:

The NNs considered are the difference of two maxpolynomials. For each of the two (+,-) maxpolynomials  $p(x)$  of the network, we first find a **divisor**  $d(x)$ . This is done by:

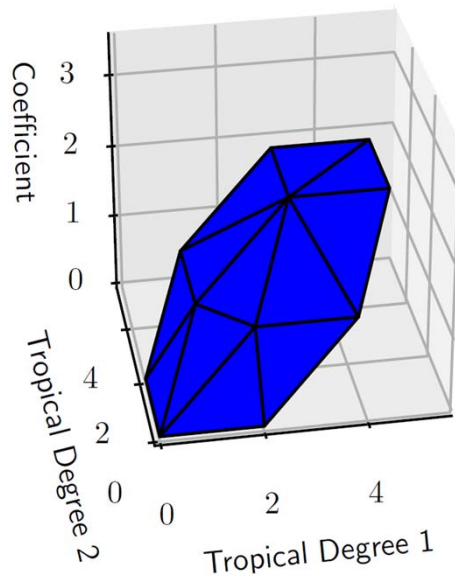
Finding the **most important vertices** of  $\text{ENewt}(p)$ , via the weights of the network (based on which combination of neurons is activated).

# Method for Single Output Neuron

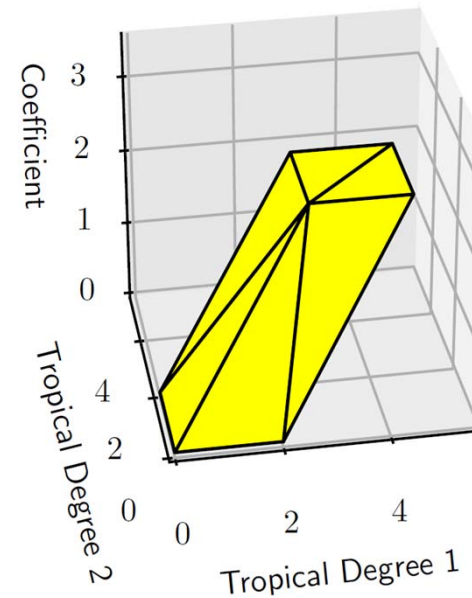
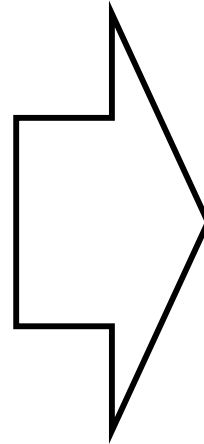


- Final polytope (right) is precisely under the original (left).
- The process is a “smoothing” of the original polytope.  
(From the 8 vertices of the original-yellow polytope we keep only the 4 blue which comprise the vertices of the final-red polytope.)

# Properties of Trop. Div. Approximation Method



*Original Network Polytope*



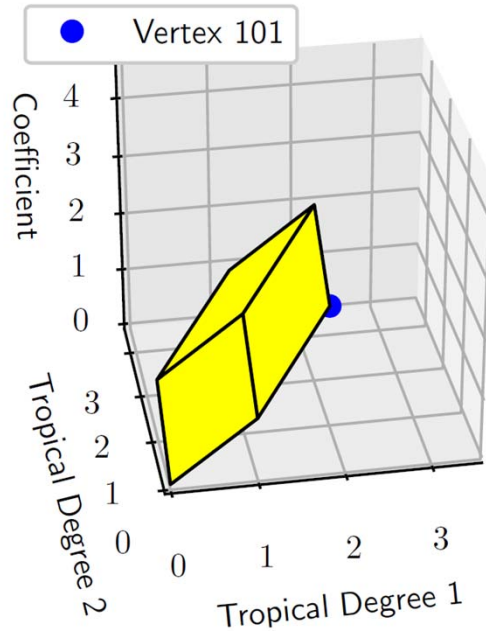
*Approximate Network Polytope*

1. Approximate polytope contains only vertices of the original.
2. The input samples activating the chosen vertices have the same output in the two networks.

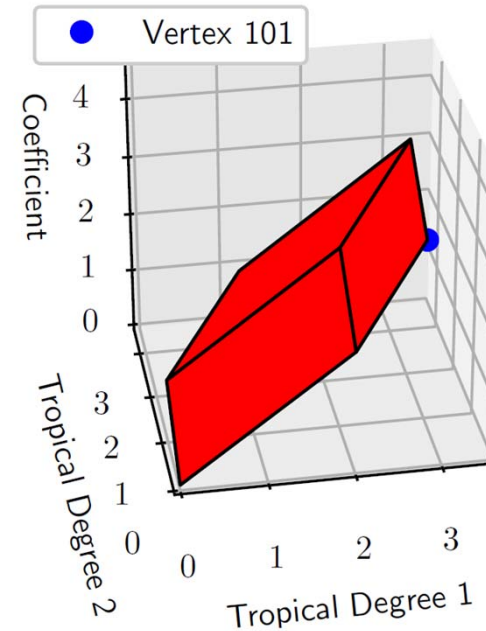
3. At least  $\frac{N}{\sum_{j=0}^d \binom{n}{j}} O(\log n')$  samples retain their output

( $N$  is # of samples,  $n$  and  $n'$  the # of neurons in hidden layer before and after the approximation). Note: this is not a tight bound.

# Extension with Multiple Output Neurons



Upper hull of polytope, *Neuron 1*

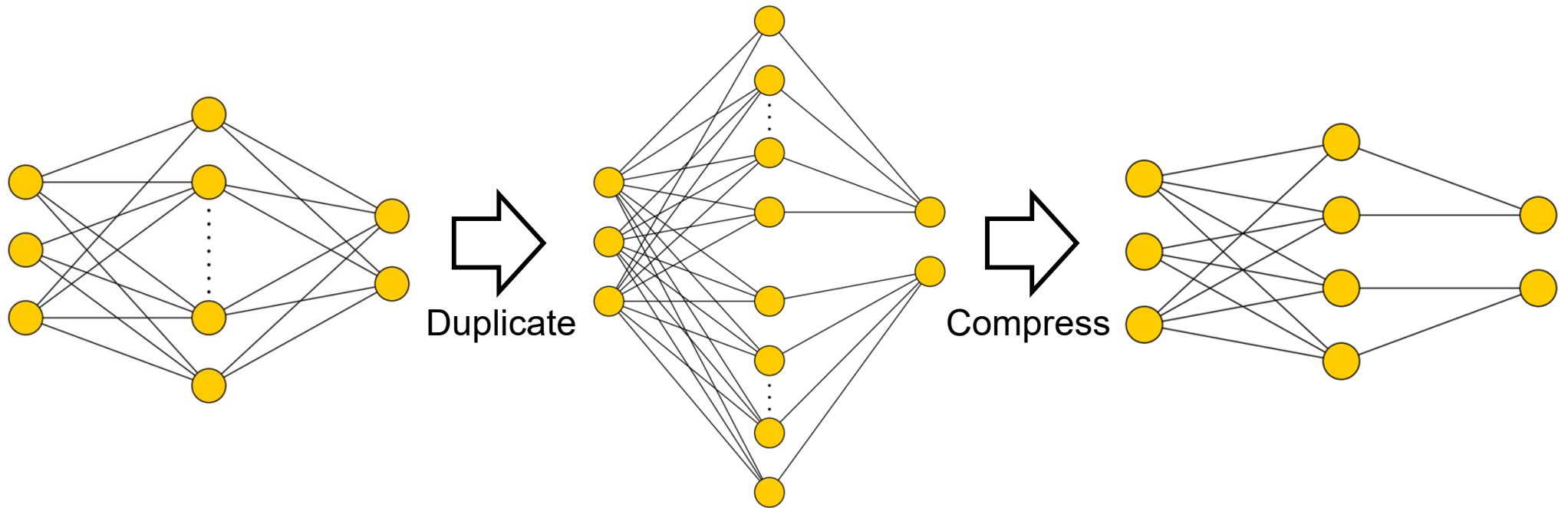


Upper hull of polytope, *Neuron 2*

- What we have: Multiple polytopes (one pair for each output neuron), interconnected (Minkowski sums of same hidden neurons but with different scaling weights).
- What we want: Simultaneous approximation of all polytopes.



# One-Vs-All Framework



# Experiments: Trop. Division NN Minimization

Neurons Kept	TropDiv Method, Avg. Accuracy	TropDiv Method, St. Deviation
Original	98.604	0.027
75%	<b>96.560</b>	<b>1.245</b>
50%	<b>96.392</b>	<b>1.177</b>
25%	<b>95.154</b>	<b>2.356</b>
10%	<b>93.748</b>	<b>2.572</b>
5%	92.928	2.589

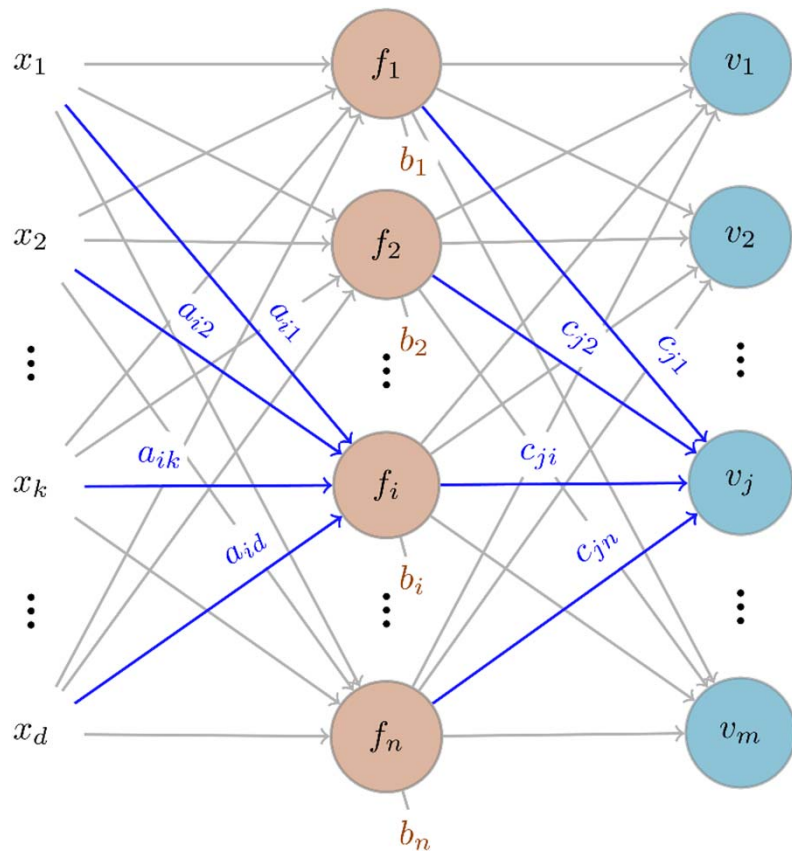
**MNIST**  
**Dataset**

Neurons Kept	TropDiv Method, Avg. Accuracy	TropDiv Method, St. Deviation
Original	88.658	0.538
75%	<b>83.556</b>	2.885
50%	<b>83.300</b>	2.799
25%	<b>82.224</b>	2.845
10%	<b>80.430</b>	3.267

**Fashion-**  
**MNIST**  
**Dataset**

[G. Smyrnis & P. Maragos, “*Multiclass Neural Net Minimization, Tropical Newton Polytope Approximation*”, ICML 2020]

# Neural Network Tropical Geometry



*1 hidden layer with ReLU activations*

$i$  – th hidden layer neuron

$$f_i(\mathbf{x}) = \max(\mathbf{a}_i^T \mathbf{x} + b_i, 0)$$

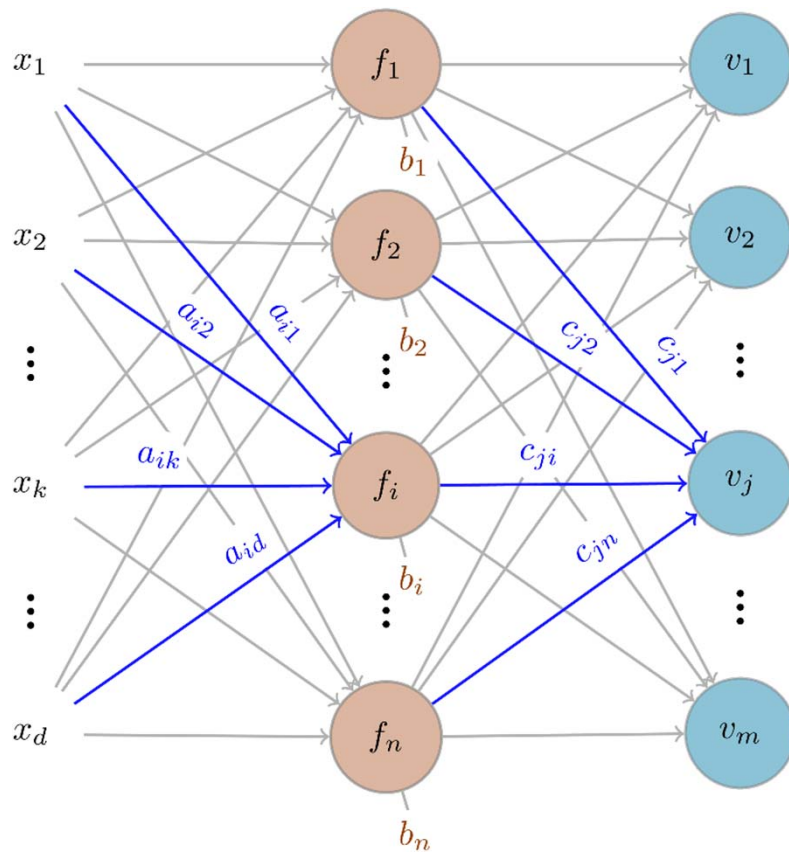
Tropical *polynomial*

$j$  – th output layer neuron

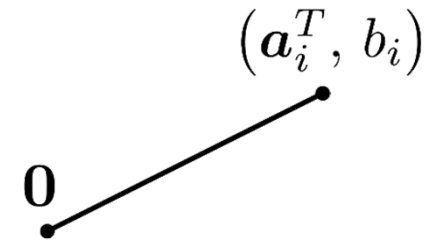
$$\begin{aligned} v_j(\mathbf{x}) &= \sum_{i=1}^n c_{ji} f_i(\mathbf{x}) \\ &= \sum_{c_{ji} > 0} |c_{ji}| f_i(\mathbf{x}) - \sum_{c_{ji} < 0} |c_{ji}| f_i(\mathbf{x}) \\ &= p_j(\mathbf{x}) - q_j(\mathbf{x}) \end{aligned}$$

Tropical *rational function*

# Neural Network Tropical Geometry



$$f_i(\mathbf{x}) = \max(\mathbf{a}_i^T \mathbf{x} + b_i, 0)$$



$\text{ENewt}(f_i)$  is a linear segment

$$\begin{aligned} v_j(\mathbf{x}) &= \sum_{c_{ji} > 0} |c_{ji}| f_i(\mathbf{x}) - \sum_{c_{ji} < 0} |c_{ji}| f_i(\mathbf{x}) \\ &= p_j(\mathbf{x}) - q_j(\mathbf{x}) \end{aligned}$$

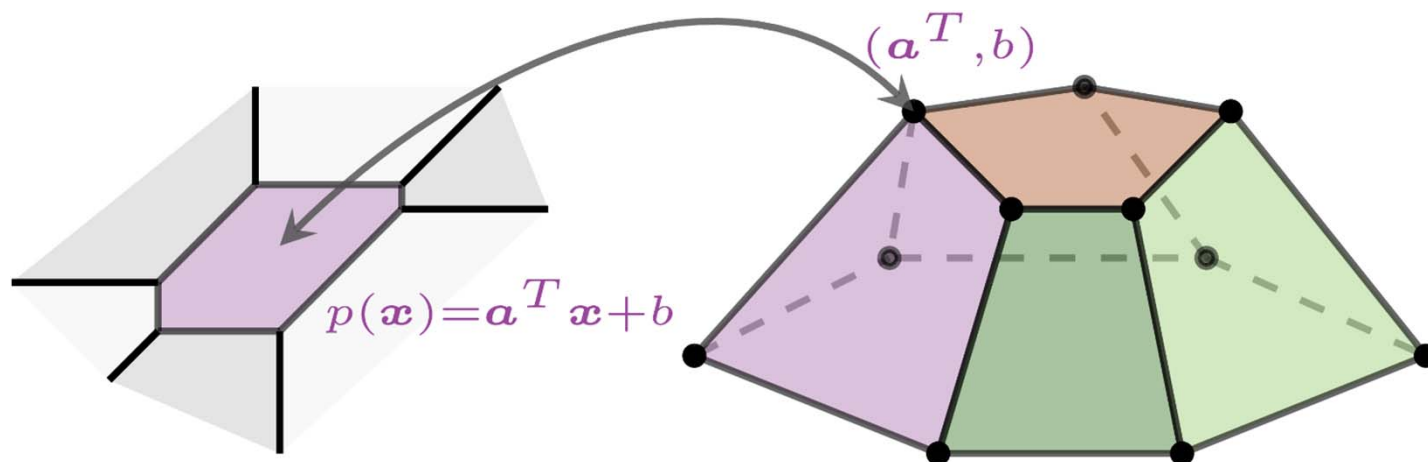
$$P_j = \text{ENewt}(p_j)$$

$$Q_j = \text{ENewt}(q_j)$$

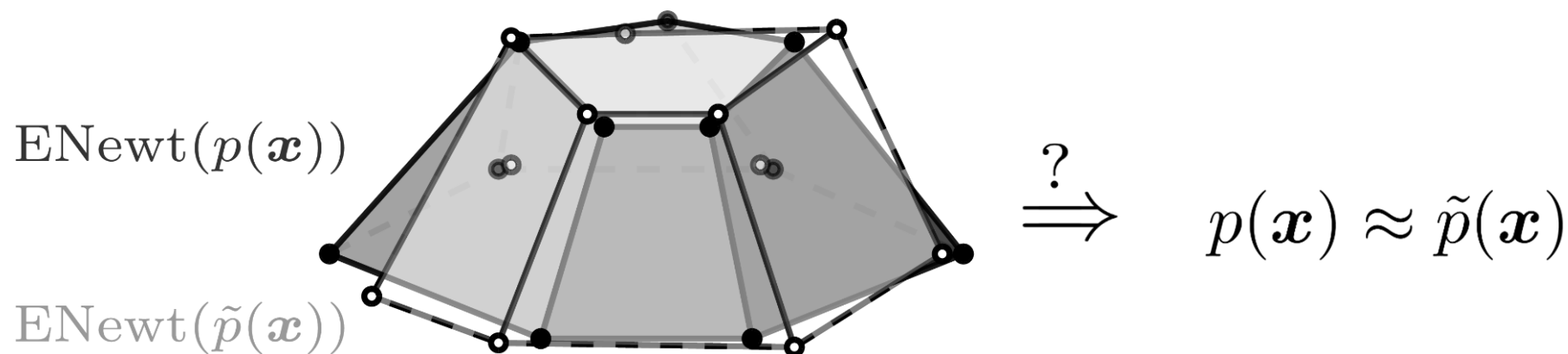
Positive and Negative  
zonotopes – or polytopes  
for deeper NNs

$c_{ji}(\mathbf{a}_i^T, b_i)$  Generators of the zonotopes

# Approximate Extended Newton Polytopes



linear regions  $\xleftrightarrow{1-1}$  vertices of the upper envelope of the extended Newton polytope



Approximate extended Newton polytopes

Approximate tropical polynomials

# Approximating Tropical Polynomials

**Proposition** Let  $p, \tilde{p} \in \mathbb{R}_{\max}[\mathbf{x}]$  and consider the polytopes  $P = \text{ENewt}(p)$ ,  $\tilde{P} = \text{ENewt}(\tilde{p})$ . Then,

$$\max_{x \in \mathcal{B}} |p(\mathbf{x}) - \tilde{p}(\mathbf{x})| \leq \rho \cdot \mathcal{H}(P, \tilde{P})$$

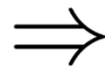
 *Hausdorff distance of polytopes*

# Neural Network Approximation Theorem

**Theorem:** Consider two neural networks  $v, \tilde{v}$  with output size  $m$  and  $P_j, Q_j, \tilde{P}_j, \tilde{Q}_j$  be the positive and negative extended Newton polytopes of  $v, \tilde{v}$  respectively. Then,

$$\max_{x \in \mathcal{B}} \|v(\mathbf{x}) - \tilde{v}(\mathbf{x})\|_1 \leq \rho \cdot \left( \sum_{j=1}^m \mathcal{H}(P_j, \tilde{P}_j) + \mathcal{H}(Q_j, \tilde{Q}_j) \right)$$

*Approximately* equal  
zonotopes

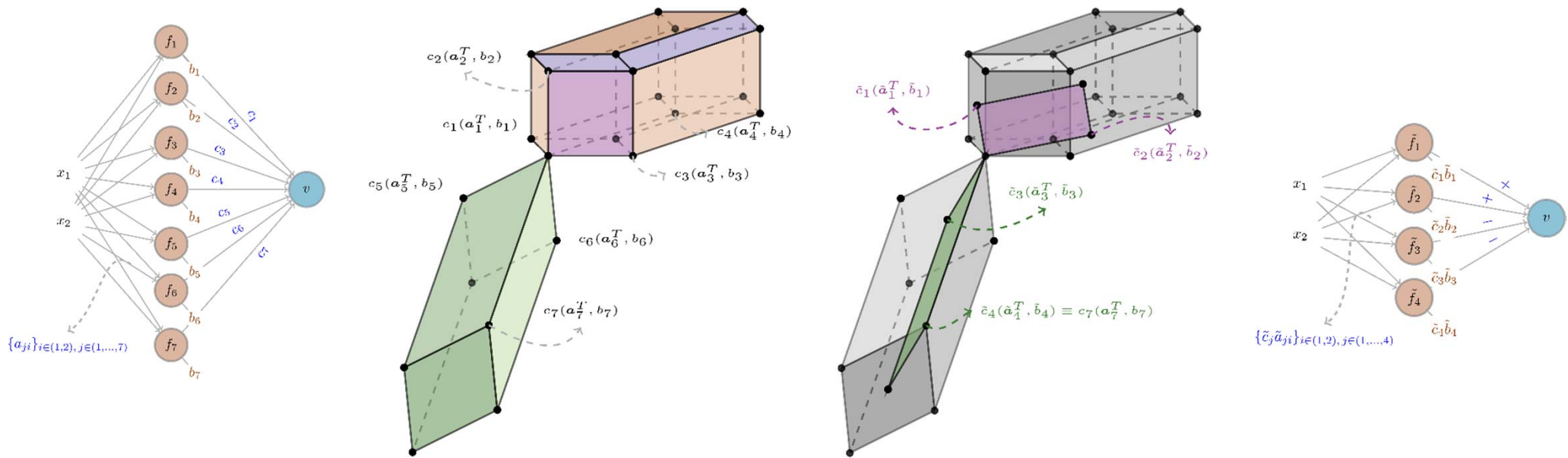


*Approximately* equivalent  
networks



# Zonotope K-Means

K-means on the positive and negative *zonotope generators*



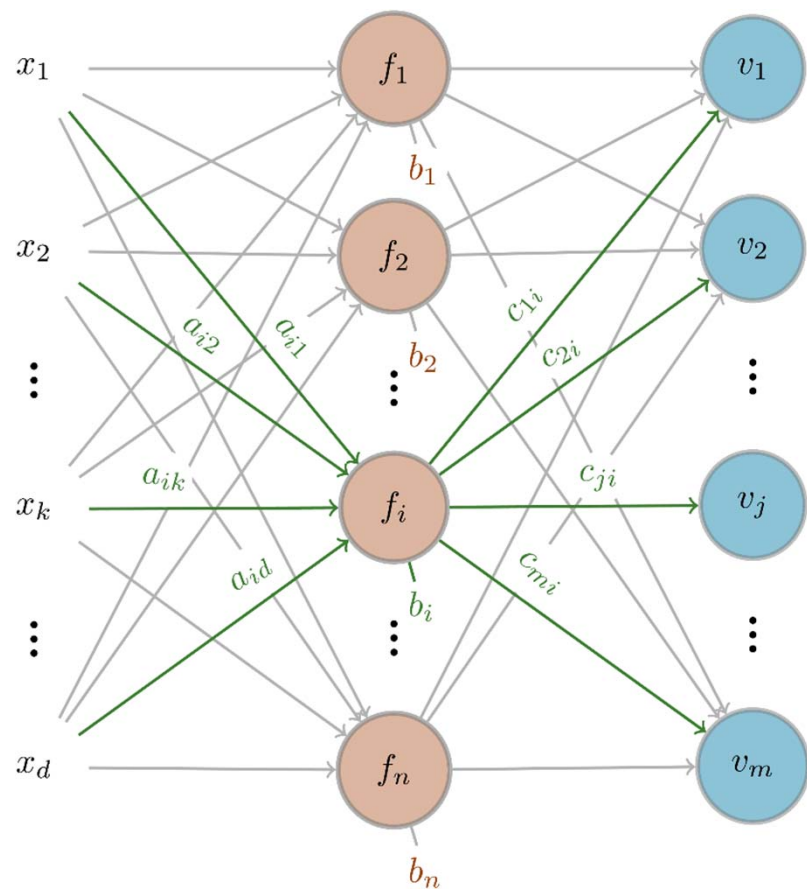
Single-output network

Original zonotopes

Approximated Zonotopes

Reduced network

# Neural Path K-means



**Generalization** for multi-output networks

K-means on the vectors associated with the *neural paths*

# Performance Results

## Binary Classification Experiments

Percentage of Remaining Neurons	MNIST 3/5			MNIST 4/9		
	Smyrnis et al., 2020	Zonotope K-means	Neural Path K-means	Smyrnis et al., 2020	Zonotope K-means	Neural Path K-means
100% (Original)	99.18 $\pm$ 0.27	99.38 $\pm$ 0.09	99.38 $\pm$ 0.09	99.53 $\pm$ 0.09	99.53 $\pm$ 0.09	99.53 $\pm$ 0.09
5%	99.12 $\pm$ 0.37	99.42 $\pm$ 0.07	99.25 $\pm$ 0.04	98.99 $\pm$ 0.09	99.52 $\pm$ 0.09	99.48 $\pm$ 0.15
1%	99.11 $\pm$ 0.36	99.39 $\pm$ 0.05	99.32 $\pm$ 0.03	99.01 $\pm$ 0.09	99.46 $\pm$ 0.05	99.35 $\pm$ 0.17
0.5%	99.18 $\pm$ 0.36	99.41 $\pm$ 0.05	99.22 $\pm$ 0.11	98.81 $\pm$ 0.09	99.35 $\pm$ 0.24	98.84 $\pm$ 1.18
0.3%	99.18 $\pm$ 0.36	99.25 $\pm$ 0.37	99.19 $\pm$ 0.41	98.81 $\pm$ 0.09	98.22 $\pm$ 1.38	98.22 $\pm$ 1.33

## Multiclass Classification Experiments

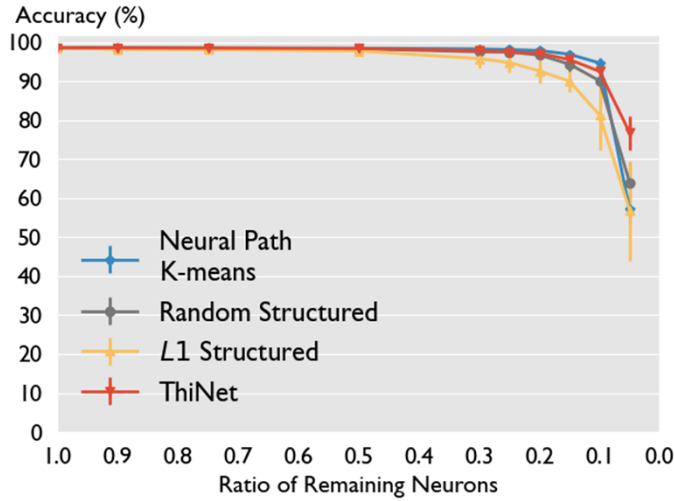
Percentage of Remaining Neurons	MNIST		Fashion-MNIST	
	Smyrnis and Maragos, 2020	Neural Path K-means	Smyrnis and Maragos, 2020	Neural Path K-means
100% (Original)	98.60 $\pm$ 0.03	98.61 $\pm$ 0.11	88.66 $\pm$ 0.54	89.52 $\pm$ 0.19
50%	96.39 $\pm$ 1.18	<b>98.13 <math>\pm</math> 0.28</b>	83.30 $\pm$ 2.80	<b>88.22 <math>\pm</math> 0.32</b>
25%	95.15 $\pm$ 2.36	<b>98.42 <math>\pm</math> 0.42</b>	82.22 $\pm$ 2.85	<b>86.67 <math>\pm</math> 1.12</b>
10%	93.48 $\pm$ 2.57	<b>96.89 <math>\pm</math> 0.55</b>	80.43 $\pm$ 3.27	<b>86.04 <math>\pm</math> 0.94</b>
5%	92.93 $\pm$ 2.59	<b>96.31 <math>\pm</math> 1.29</b>	—	<b>83.68 <math>\pm</math> 1.06</b>

[ P. Misiakos, G. Smyrnis, G. Retsinas and P. Maragos, “*Neural Network Approximation based on Hausdorff distance of Tropical Zonotopes*”, Proc. ICLR 2022 ]

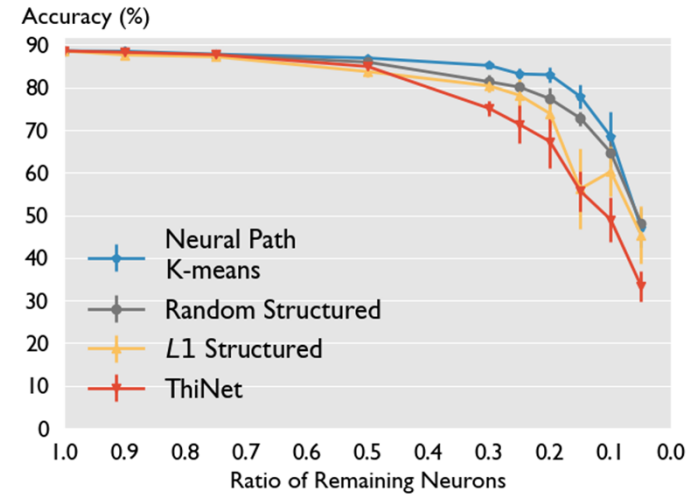
# Comparison with ThiNet and Baselines

LeNet5

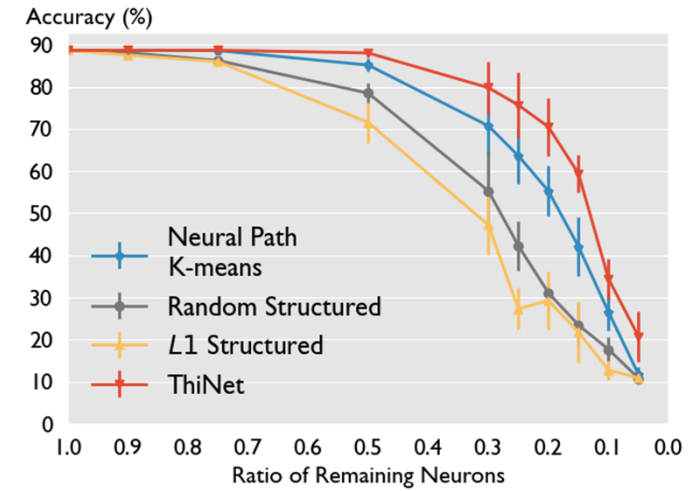
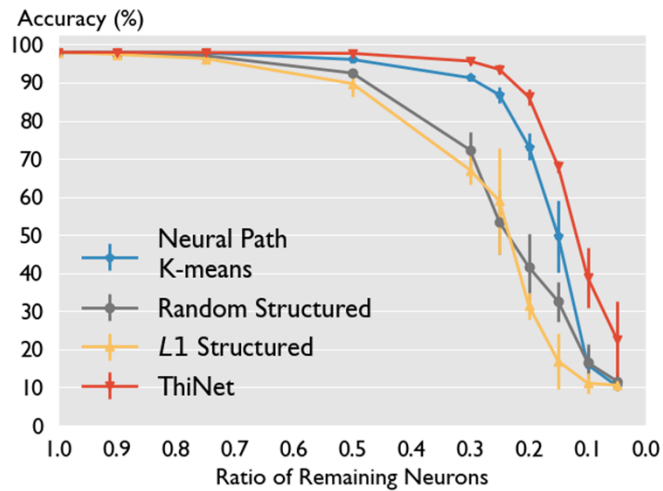
## MNIST



## Fashion-MNIST



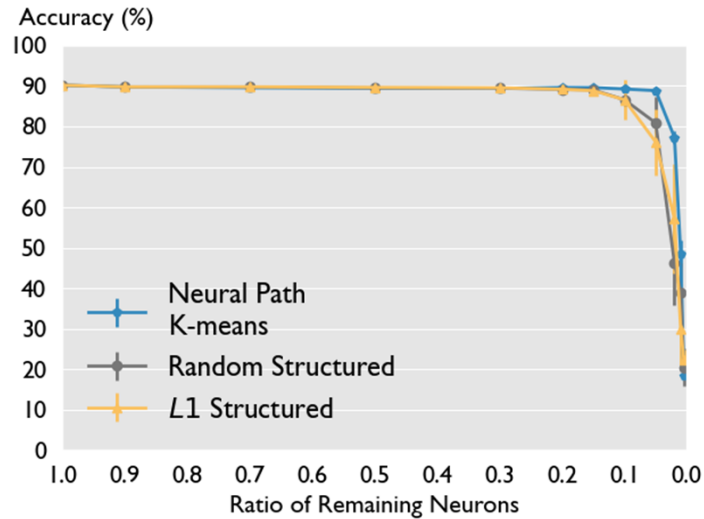
custom deep NN



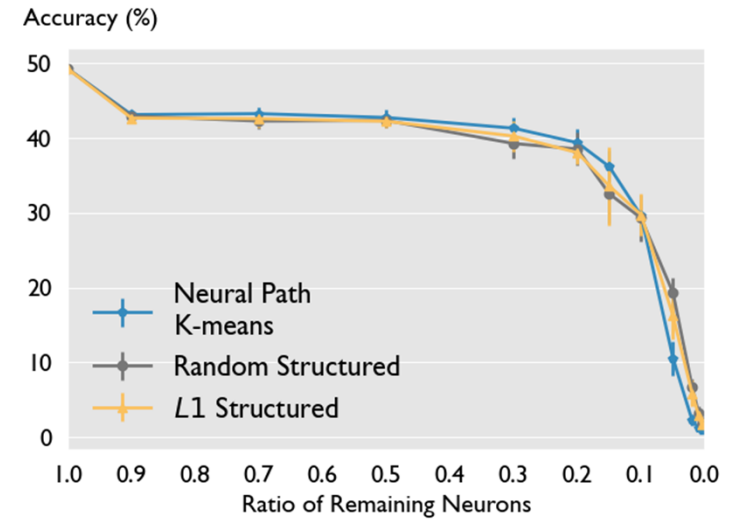
# Comparison with baselines

CIFAR-VGG

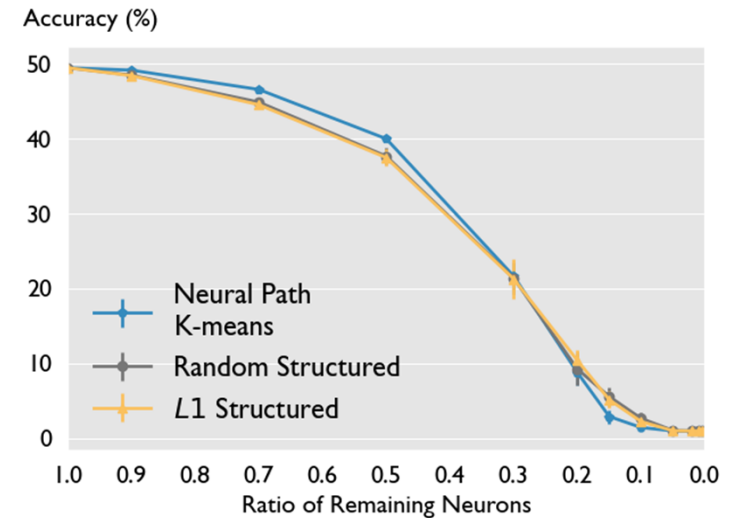
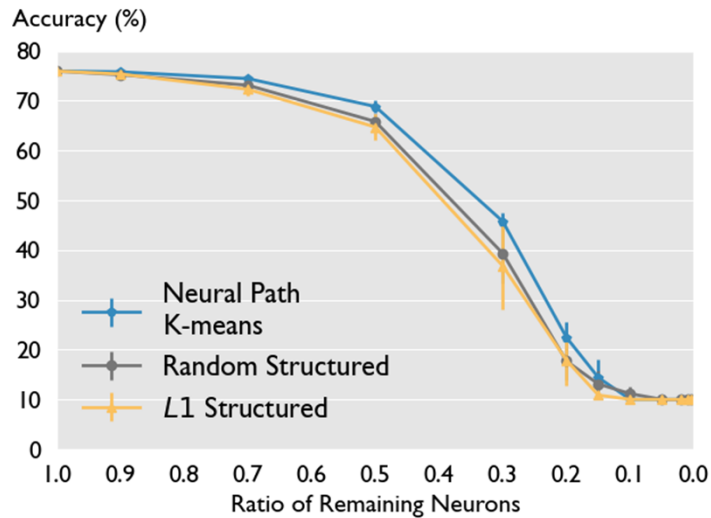
## CIFAR10



## CIFAR100



AlexNet



# Tropical Regression and Piecewise-Linear Surface Fitting

## References:

- P. Maragos and E. Theodosis, “[Multivariate Tropical Regression and Piecewise-Linear Surface Fitting](#)”, *Proc. ICASSP*, 2020.
- P. Maragos, V. Charisopoulos and E. Theodosis, “[Tropical Geometry and Machine Learning](#)”, *Proceedings of the IEEE*, 2021.

# Optimal Regression for Fitting Euclidean vs Tropical Lines

**Problem:** Fit a curve to data  $(x_i, y_i)$ ,  $i = 1, \dots, m$

**Euclidean:**

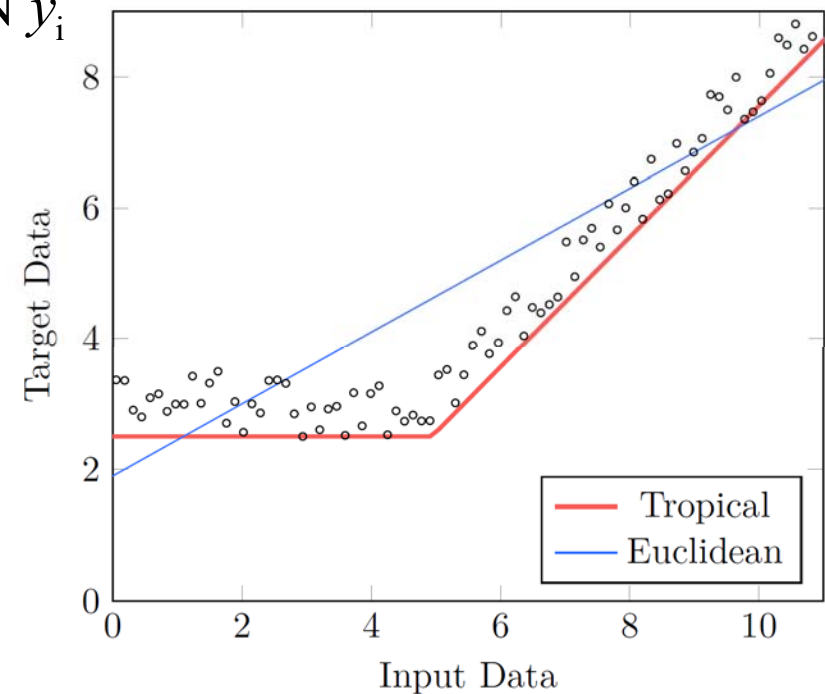
Fit a straight line  $y = ax + b$  by minimizing  $\ell_2$ -norm of error:

$$a = \frac{\sum x_i y_i - (\sum x_i)(\sum y_i) / m}{\sum (x_i)^2 - (\sum x_i)^2 / m}, \quad b = \frac{1}{m} \sum y_i - ax_i$$

**Tropical:**

Fit a tropical line  $y = \max(a + x, b)$  by minimizing some  $\ell_p$ -norm of error:

Greatest Subsolution:  $a = \text{MIN}_i y_i - x_i$ ,  $b = \text{MIN}_i y_i$





# Solve Max-plus Equations

- **Problems:**

(1) Exact problem: Solve  $\delta_A(\mathbf{x}) = \mathbf{A} \boxplus \mathbf{x} = \mathbf{b}$ ,  $\mathbf{A} \in \overline{\mathbb{R}}^{m \times n}$ ,  $\mathbf{b} \in \overline{\mathbb{R}}^m$

(2) Approximate Constrained: Min  $\|\mathbf{A} \boxplus \mathbf{x} - \mathbf{b}\|_{p=1 \dots \infty}$  s.t.  $\mathbf{A} \boxplus \mathbf{x} \leq \mathbf{b}$

- **Theorem:** (a) The **greatest (sub)solution** of (1) and unique solution of (2) is

$$\hat{\mathbf{x}} = \varepsilon_A(\mathbf{b}) = \mathbf{A}^* \boxplus' \mathbf{b} = \left[ \bigwedge_i b_i - a_{ij} \right], \quad \mathbf{A}^* \triangleq -\mathbf{A}^T$$

and yields the **Greatest Lower Estimate (GLE)** of data  $\mathbf{b}$ :

$$\delta_A(\varepsilon_A(\mathbf{b})) = \mathbf{A} \boxplus (\mathbf{A}^* \boxplus' \mathbf{b}) \leq \mathbf{b}$$

- (b) **Min Max Absolute Error (MMAE) unconstrained unique solution:**

$$\tilde{\mathbf{x}} = \hat{\mathbf{x}} + \mu, \quad \mu = \|\mathbf{A} \boxplus \hat{\mathbf{x}} - \mathbf{b}\|_{\infty} / 2$$

- **Geometry:** Operators  $\delta, \varepsilon$  are vector dilation and erosion, and the **GLE**  $\mathbf{b} \mapsto \delta\varepsilon(\mathbf{b})$  is an opening (**lattice projection**).

- **Complexity:**  $O(mn)$

**Sparse solutions:** [Tsiamis & Maragos 2019],  
[Tsilivis et al. 2021]



# Optimally Fitting Tropical Lines to Data

**Problem:** Fit a tropical line  $y = \max(a + x, b)$  to noisy data  $(x_i, f_i)$ ,  $i = 1, \dots, m$ , where  $f_i = y_i + \text{error}$  by minimizing  $\ell_{1, \dots, \infty}$  norm of error:

**Greatest Subsolution (GLE):**  $\hat{w} = (\hat{a}, \hat{b})$ ,  $\hat{a} = \text{MIN}_i f_i - x_i$ ,  $\hat{b} = \text{MIN}_i f_i$

**Min Max Abs. Error (MMAE) Solution:**  $\tilde{w} = \hat{w} + \mu$ ,  $\mu = \|\text{GLE error}\|_{\infty} / 2$

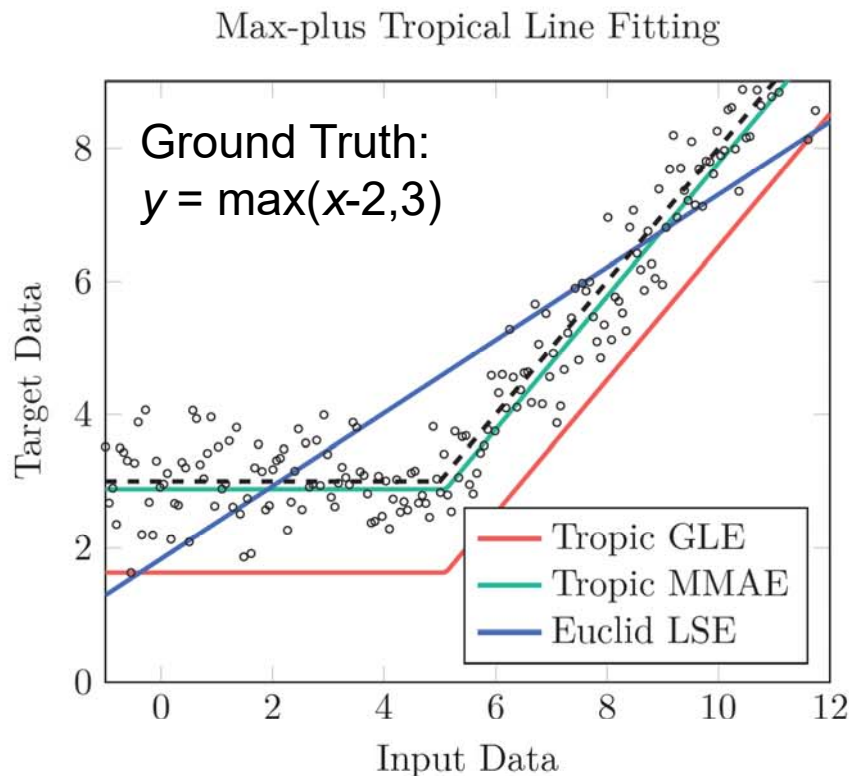
$$\underbrace{\begin{bmatrix} x_1 & 0 \\ \vdots & \vdots \\ x_m & 0 \end{bmatrix}}_{\mathbf{X}} \boxplus \underbrace{\begin{bmatrix} a \\ b \end{bmatrix}}_{\mathbf{w}} = \underbrace{\begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix}}_{\mathbf{f}} \implies \underbrace{\begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix}}_{\hat{\mathbf{w}}} = \underbrace{\begin{bmatrix} \bigwedge_i f_i - x_i \\ \bigwedge_i f_i \end{bmatrix}}_{\mathbf{X}^* \boxplus' \mathbf{f}}$$

# Numerical Examples of Optimally Fitting Tropical Lines to Data

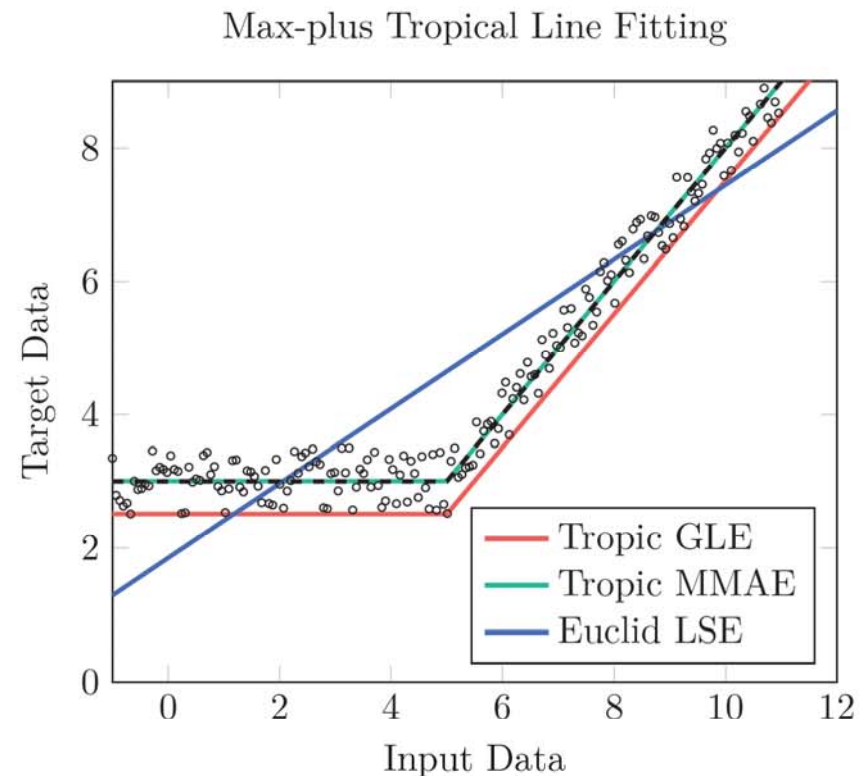
**Problem:** Fit a tropical line  $y = \max(a + x, b)$  to noisy data  $(x_i, f_i)$ ,  $i = 1, \dots, m = 200$ , where  $f_i = y_i + \text{error}$  by minimizing  $\ell_{1, \dots, \infty}$  of error:

**Greatest Subsolution (GLE):**  $\hat{w} = (\hat{a}, \hat{b})$ ,  $\hat{a} = \text{MIN}_i f_i - x_i$ ,  $\hat{b} = \text{MIN}_i f_i$

**Min Max Abs. Error (MMAE) Solution:**  $\tilde{w} = \hat{w} + \mu$ ,  $\mu = \|\text{GLE error}\|_{\infty} / 2$



(a) T-line with Gaussian Noise



(b) T-line with Uniform Noise

# Optimal Fitting Max-Plus Tropical Planes to Data

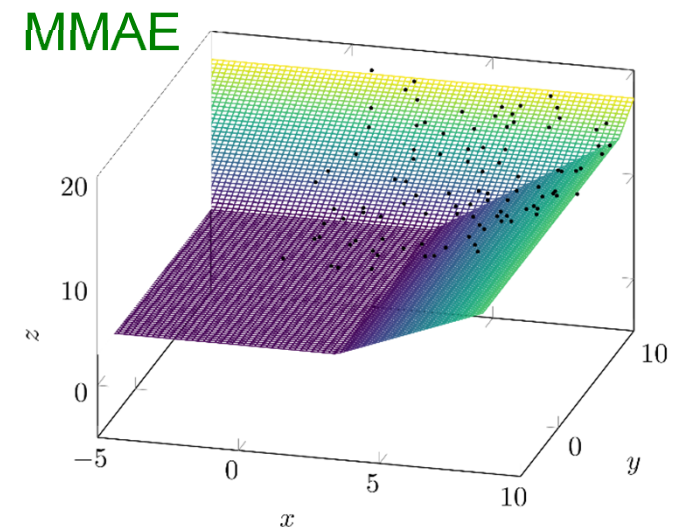
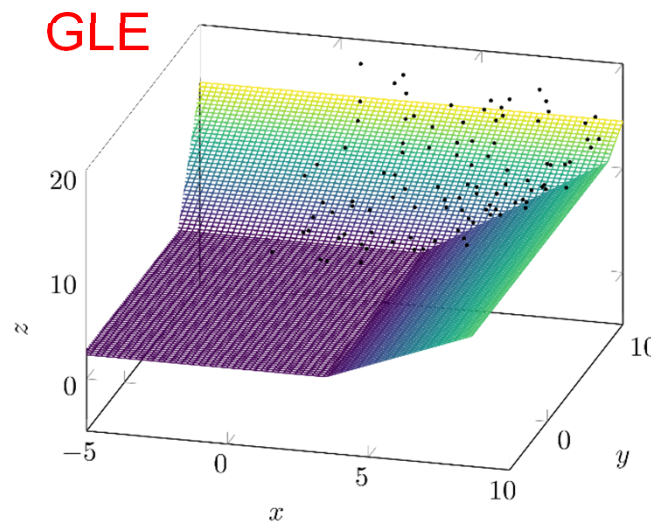
**Problem:** Fit a tropical plane  $z = \max(a + x, b + y, c)$  to noisy data  $(x_i, y_i, f_i)$ , where  $f_i = z_i + \text{error}$ ,  $i = 1, \dots, m = 100$ , by minimizing  $\ell_{1, \dots, \infty}$  norm of error:

**Greatest Subsolution (GLE):**  $\hat{w} = (\hat{a}, \hat{b}, \hat{c})$

**Min Max Abs. Error (MMAE) Solution:**  $\tilde{w} = \hat{w} + \mu$ ,  $\mu = \|\text{GLE error}\|_{\infty} / 2$

$$\underbrace{\begin{bmatrix} x_1 & y_1 & 0 \\ \vdots & \vdots & \vdots \\ x_m & y_m & 0 \end{bmatrix}}_{\mathbf{X}} \boxplus \underbrace{\begin{bmatrix} a \\ b \\ c \end{bmatrix}}_{\mathbf{w}} = \underbrace{\begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix}}_{\mathbf{f}} \implies \underbrace{\begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix}}_{\hat{\mathbf{w}}} = \underbrace{\begin{bmatrix} \bigwedge_i f_i - x_i \\ \bigwedge_i f_i - y_i \\ \bigwedge_i f_i \end{bmatrix}}_{\mathbf{X}^* \boxplus \mathbf{f}}$$

Ground Truth:  
 $z = \max(x + 5, y + 7, 9)$   
 Noise:  $N(0, 1)$



# Optimal Fitting 2D Higher-degree Tropical Polynomials to Data

**Data** (noisy paraboloid):

3D tuples  $(x_i, y_i, f_i) \in \mathbb{R}^3$

$$f_i = x_i^2 + y_i^2 + \varepsilon_i,$$

$(x_i, y_i) \sim \text{Unif}[-1, 1]$

$$\varepsilon_i \sim \mathcal{N}(0, 0.25^2)$$

**Model:**

Fit  $K$ -rank 2D trop. polynomial

$$p(x, y) = \text{MAX}_{k=1}^K \{a_k x + b_k y + c_k\}$$

by minimizing error  $\|f_i - p(x_i, y_i)\|_\infty$

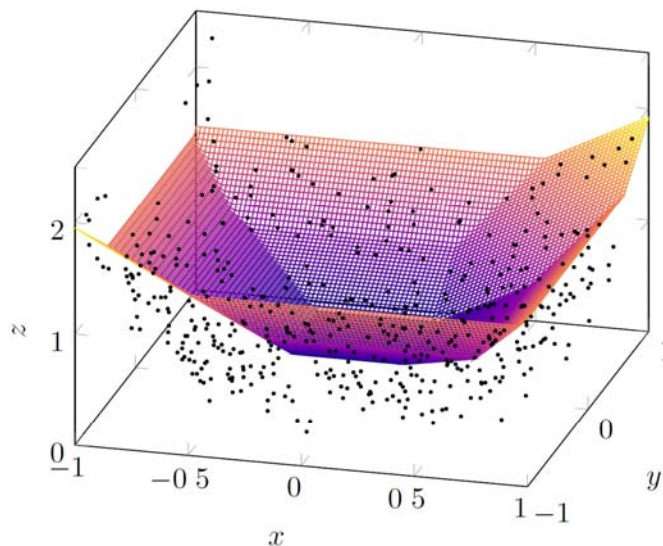
**Estimation algorithm:**

$K$  - means on data gradients  $\rightarrow a_k, b_k$

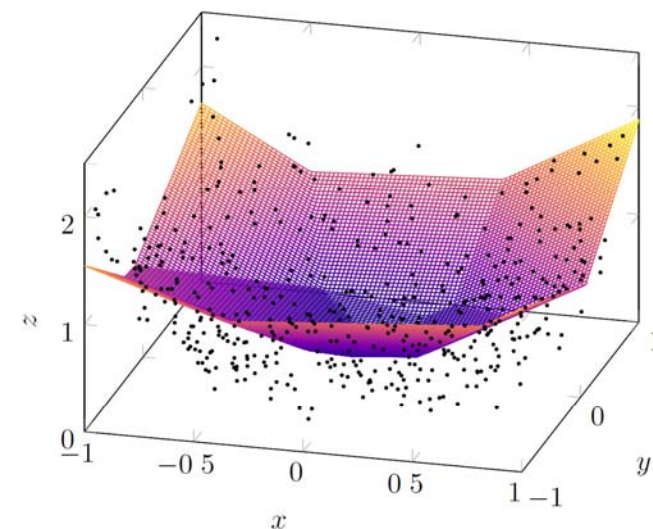
solve max-plus eqns  $\rightarrow c_k$

**Complexity:**  $\approx$  Linear

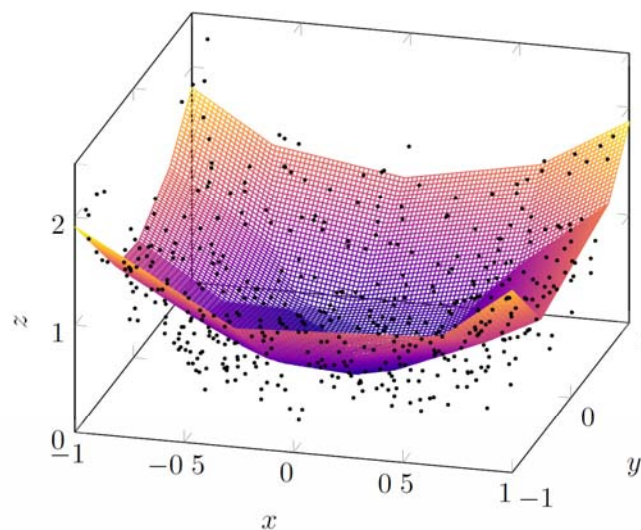
$O(\#\text{data}, \#\text{dimensions})$



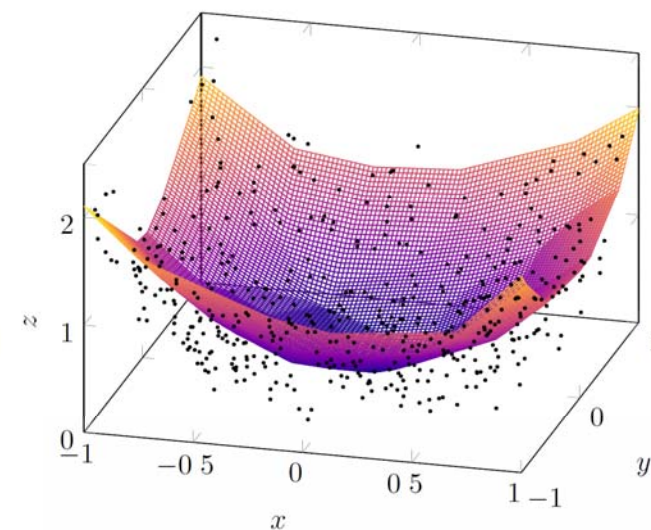
(a) 2D conic ( $K=11$ )



(b)  $K=10$



(c)  $K=25$



(d)  $K=100$



# Conclusions

- **Tropical Geometry**, and its underlying **max-plus algebra**, provide many effective and insightful tools for the analysis of NNs with PWL activations and other ML systems.
- **Morphological NNs** (with max-plus & min-plus nodes): show similar performance and superior compression ability compared to their linear counterparts. (Trained via CCP or SGD/Adam.)
- **Tropical Regression**: Tropical Polynomials for multidimensional data fitting using PWL functions. Low-complexity algorithm based on optimal solutions of systems of max-plus equations.
- **Approximation of NNs**: Tropical geometry offers effective and insightful tools for compression of NNs.
- **Future work**: extensions to deeper networks and to more general functions using max-<sup>\*</sup> algebra on weighted lattices.

# Thank you for your attention!

## Collaborators and References

Charisopoulos, Vasileios  
Dimitriadis, Nikolaos  
Misiakos, Panagiotis

Smyrnis, Georgios  
Theodosis, Emmanouil  
Tsilivis, Nikos

### General Reference:

- P. Maragos, V. Charisopoulos and E. Theodosis, “[\*Tropical Geometry and Machine Learning\*](#)”, Proceedings of the IEEE, 2021.

For more information, demos, and current results:

<http://robotics.ntua.gr> and <http://cvsp.cs.ntua.gr>