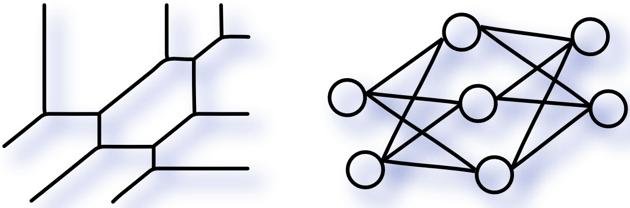


Combinatorial Geometry of Deep Neural Networks

Liwen Zhang* • Gregory Naitzat** • Lek-Heng Lim**

* Facebook
** The University of Chicago

Overview

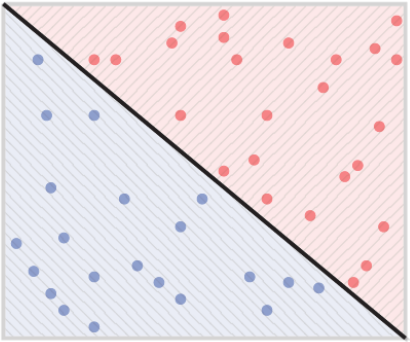
- **Result I.** A large class of neural networks and **tropical rational functions** are equivalent:

$$\sigma^{(L)} \circ \rho^{(L)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}(x) \quad \Longleftrightarrow \quad \underbrace{\frac{f(x)}{g(x)} = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n}{b_0 + b_1 x + b_2 x^2 + \cdots + b_m x^m}}_{\text{multivariate, over tropical semiring}}$$

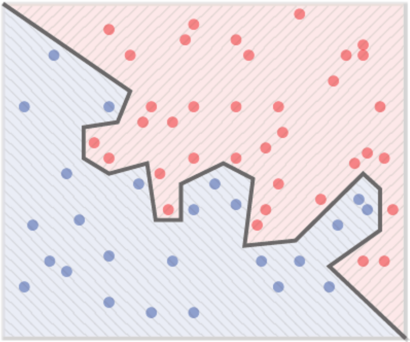
A new way to look at and study neural networks.

- **Result II.** We use our approach to show that deep neural networks are exponentially more expressive than shallow networks.

ReLU network: a complex classification problem requires decision boundary with **many linear pieces**

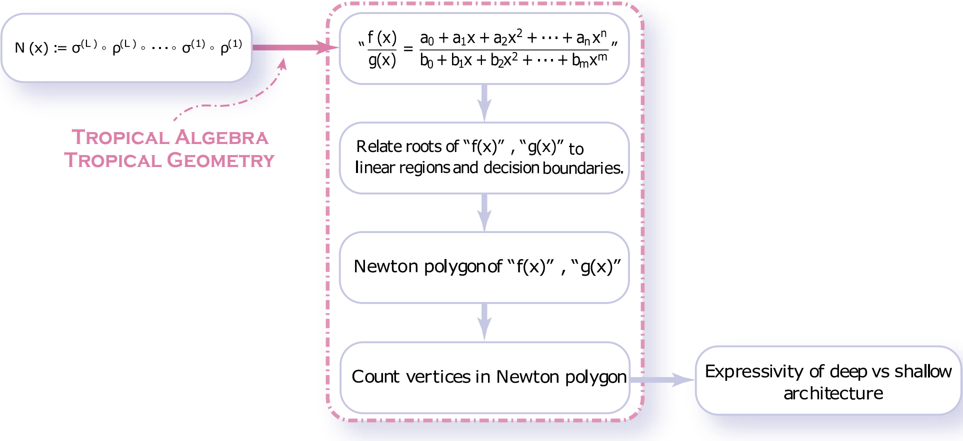


A simple decision boundary.



A complex decision boundary.

Overview


$N(x) := \sigma^{(L)} \circ \rho^{(L)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}$
TROPICAL ALGEBRA
TROPICAL GEOMETRY
$“\frac{f(x)}{g(x)} = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n}{b_0 + b_1 x + b_2 x^2 + \cdots + b_m x^m}”$
Relate roots of “f(x)” , “g(x)” to linear regions and decision boundaries.
Newton polygon of “f(x)” , “g(x)”
Count vertices in Newton polygon
Expressivity of deep vs shallow architecture

Tropical semiring

- **Tropical semiring** $\mathbb{T} := (\mathbb{R} \cup \{-\infty\}, \oplus, \odot)$:

$$x \oplus y := \max\{x, y\} \qquad \text{(tropical addition)}$$
$$x \odot y := x + y \qquad \text{(tropical multiplication)}$$

- $-\infty$ is the additive identity: $-\infty \oplus x = \max\{-\infty, x\} = x$.
- 0 is the multiplicative identity: $0 \odot x = 0 + x = x$.
- **Actually a semifield** (i.e., no additive inverse) but historically called a semiring.
- Isomorphic alternative: $\min$ in place of $\max$, $+\infty$ in place of $-\infty$
- Many algebraic objects and notions generalize to tropical settings (e.g., matrices, polynomials, tensors, rank, determinant, degree, etc) but interpretation changes.
- **Tropical algebraic geometry**: $\mathbb{T}$ in place of $\mathbb{C}$ in algebraic geometry (roughly).
- Provides fresh perspectives and new powerful techniques [Maclagan–Sturmfels 2015].

Examples

- **Symplectic Geometry.** Gromov–Witten invariants can be found tropically [Mikhalkin 2005]
- **Integer Programming.** $A \in \mathbb{Z}_+^{m\times n}$, $b \in \mathbb{Z}_+^m$, $c \in \mathbb{R}^n$ (assume $A^\mathsf{T}\mathbb{1} = a\mathbb{1}$ and $b^\mathsf{T}\mathbb{1} = ma$),

  $$\text{maximize } c^\mathsf{T}x \quad \text{subject to } Ax = b, \; x \in \mathbb{Z}_+^n.$$

  Solution is coefficient of monomial $y_1^{b_1} \odot y_2^{b_2} \odot \cdots \odot y_m^{b_m}$ in dth tropical power of

  $$c_1 \odot y_1^{a_{11}} \odot y_2^{a_{21}} \odot \cdots \odot y_m^{a_{m1}} \oplus \cdots \oplus c_n \odot y_1^{a_{1n}} \odot y_2^{a_{2n}} \odot \cdots \odot y_m^{a_{mn}}.$$

- **Computer Science.** Floyd's algorithm for shortest path in weighted digraph G = tropical power of adjacency matrix $A_G \in \mathbb{R}^{n\times n}$, i.e., $A_G^{\odot n} = A_G \odot A_G \odot \cdots \odot A_G$. Hungarian assignment method = tropical Gaussian elimination for computing tropical determinant,

  $$\operatorname{tropdet}(X) = \bigoplus\nolimits_{\pi\in\mathfrak{S}_n} x_{1\pi(1)} \odot x_{2\pi(2)} \odot \cdots \odot x_{n\pi(n)},$$

  x_{ij} = cost of assigning job i to worker j.

Tropical power

- **Tropical power.** $n \in \mathbb{N}$,

$$x^n := x^{\odot n} := \underbrace{x \odot \cdots \odot x}_{n \text{ times}} = n \cdot x$$

  where $\cdot$ is standard multiplication.

- **Tropical monomial.** $c \in \mathbb{R} \cup \{-\infty\}, a_i \in \mathbb{N}$,

$$cx^\alpha := c \odot x_1^{a_1} \odot x_2^{a_2} \odot \cdots \odot x_d^{a_d}$$

  where $\alpha = (a_1, \ldots, a_d) \in \mathbb{N}^d$.

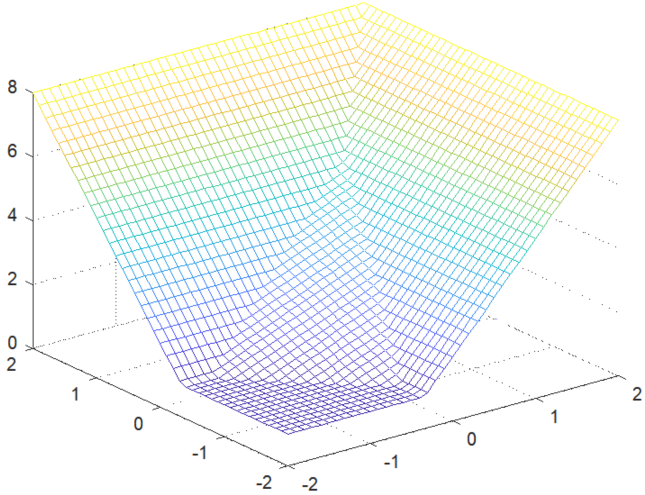
- **Tropical polynomial.** Tropical sum of tropical monomials

$$f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$$

  where $\alpha_i = (a_{i1}, \ldots, a_{id}) \in \mathbb{N}^d$ and $c_i \in \mathbb{R} \cup \{-\infty\}$, $i = 1, \ldots, r$.

Tropical polynomial

$1 \oplus 2x^3 \oplus 2y^3 \oplus 3xy = \max\{1, 2+3x, 2+3y, 3+x+y\}$, tropical polynomials are **convex** piecewise linear functions with **integer** slopes



Tropical division

- **Tropical negative power** $x^{-n} = (-x)^n$; it follows that **tropical division** is given by:

$$\frac{x}{y} \coloneqq x \oslash y \coloneqq x - y$$

- **Tropical rational function** is a (standard) difference of two tropical polynomials $f(x)$ and $g(x)$

$$r(x) \coloneqq f(x) \oslash g(x) = f(x) - g(x)$$

- Tropical polynomial (or rational) **map** $F : \mathbb{R}^d \to \mathbb{R}^n$ is a **vector valued function** where each component is tropical polynomial (or rational) function; $F(x) = (f_1(x), \ldots, f_n(x))$ where $f_i(x)$ is tropical polynomial (or rational function).

Deep neural networks

- Deep neural networks produce state of the art results across variety of applications in machine learning, especially in computer vision.

Classifier	Error rate (%)
30 Random Deep Leaning (RDL) models (10 CNNs, 10 RNNs, and 10 DNN)	0.18[18]
Committee of 5 CNNs, 6-layer 784-50-100-500-1000-10-10	0.21[17]
Committee of 35 CNNs, 1-20-P-40-P-150-10	0.23[8]
6-layer 784-50-100-500-1000-10-10	0.27[16]
6-layer 784-40-80-500-1000-2000-10	0.31[15]
6-layer 784-2500-2000-1500-1000-500-10	0.35[23]

MNIST

Research Paper	Error rate (%)
ShakeDrop regularization[13]	2.31
Improved Regularization of Convolutional Neural Networks with Cutout[12]	2.56
Shake-Shake regularization[11]	2.86
Fractional Max-Pooling[10]	3.47
Neural Architecture Search with Reinforcement Learning[9]	3.65
Wide Residual Networks[8]	4.0
Densely Connected Convolutional Networks[7]	5.19
Convolutional Deep Belief Networks on CIFAR-10[6]	21.1

CIFAR-10

Neural networks

Feedforward neural network with L **layers**, is given by the continuous function $\nu : \mathbb{R}^d \to \mathbb{R}^{n_L}$ of the form

$$\nu(x) := \sigma^{(L)} \circ \rho^{(L)} \circ \cdots \circ \sigma^{(1)} \circ \rho^{(1)}(x),$$

where $\rho^{(i)} : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}$ is affine **preactivation**

$$\rho^{(i)}(y) = A^{(i)}y + b^{(i)}$$

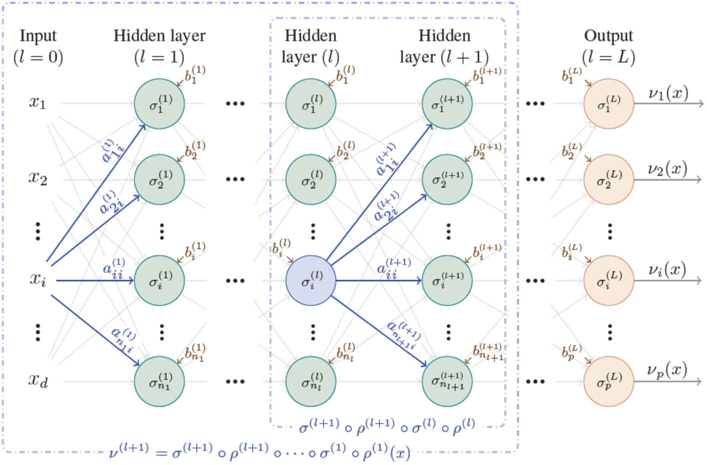
and $\sigma^{(i)}(x) := \max(x, t_i)$ is **activation** function $t_i \in \mathbb{R}$ threshold. Write σ if $t_i = 0$.

$$\nu(x) = \sigma\left(A^{(L)}\sigma\left(A^{(L-1)}\sigma\left(\cdots\sigma\left(A^{(1)}x + b^{(1)}\right) + b^{(L-1)}\right)\right) + b^{(L)}\right)$$

Collectively $A^{(i)}, b^{(i)}, i = 1, \ldots L$ form **parameters** of the network, and are determined during the training, usually by some form of SGD.

Neural networks

$$\nu(x) = \sigma\left(A^{(L)}\sigma\left(A^{(L-1)}\sigma\left(\cdots\sigma\left(A^{(1)}x + b^{(1)}\right) + b^{(L-1)}\right)\right) + b^{(L)}\right)$$



Why neural networks?

Classical result: Two-layer neural network ν, i.e., $L = 2$, **can approximate any function** $f \in L^p(\mathbb{R}^d)$, $1 \leq p \leq \infty$, arbitrary well in norm [Hornik et al. 1989, Hornik 1990],

$$\|f - \nu\|_p < \varepsilon.$$

Doesn't explain deep neural networks: Why do we need more layers?

Why does nonsmooth activation $\sigma(x) = \max(0, x)$ work better than smooth activations like $\sigma(x) = \tanh(x)$ or $1/(1 - e^{-x})$?

Many other mysteries.

THEOREM (ZHANG–NAITZAT–L 2018)

The following families of functions are equivalent:

1. *feedforward ReLU neural network with integer weights;*
2. *tropical rational maps;*
3. *continuous piecewise linear maps with integer coefficients.*

Assume integer weights, i.e., $A^{(i)} \in \mathbb{Z}^{n_{i-1} \times n_i}$ from now on.

Mild assumption: real weights can be approximated arbitrarily closely by rational weights; since parameters may be scaled by any positive constant, may clear denominators to get integer weights.

Neural networks and tropical algebra

The proof of $1 \implies 2$ is constructive.

Proposition (Zhang–Naitzat–L 2018)

Assume the lth layer of neural net is given by a tropical rational map $\nu^{(l)}(x) = F^{(l)}(x) \oslash G^{(l)}(x)$, then

$$\nu^{(l+1)}(x) = F^{(l+1)}(x) \oslash G^{(l+1)}(x)$$

where $F^{(l+1)}(x)$ and $G^{(l+1)}(x)$ depend on $F^{(l)}(x)$, $G^{(l)}(x)$. (Expressions later)

Goal: Study geometry of tropical polynomial maps $F^{(l)}(x)$ and $G^{(l)}(x)$.

Tropical geometry of neural network

Each layer in ReLU neural network is a tropical rational map

$$\nu^{(i+1)} = F^{(i+1)} - G^{(i+1)} = F^{(i+1)} \oslash G^{(i+1)}$$

and

$$\begin{aligned}
F_j^{(i+1)} &= H_j^{(i+1)} \oplus G_j^{(i+1)}, \\
G_j^{(i+1)} &= \left[\bigodot_{k=1}^{n_i} (F_k^{(i)})^{a_{jk,-}^{(i+1)}} \right] \odot \left[\bigodot_{k=1}^{n_i} (G_k^{(i)})^{a_{jk,+}^{(i+1)}} \right], \\
H_j^{(i+1)} &= \left[\bigodot_{k=1}^{n_i} (F_k^{(i)})^{a_{jk,+}^{(i+1)}} \right] \odot \left[\bigodot_{k=1}^{n_i} (G_k^{(i)})^{a_{jk,-}^{(i+1)}} \right] \odot b_j^{(i+1)}.
\end{aligned}$$

Tropical hypersurface

We use only the most basic notions from tropical algebraic geometry.

- Tropical analogue of roots of polynomials: **tropical hypersurface**.
  - tropical polynomial is **"vanishing"** at x if its value at x is attained by more than one monomials: $c_i x^{\alpha_i} = c_j x^{\alpha_j}$ for some $\alpha_i \neq \alpha_j$;
  - **tropical hypersurface** is the set of all x where tropical polynomial is "vanishing"

$$\mathcal{T}(f) := \left\{ x \in \mathbb{R}^d : c_i x^{\alpha_i} = c_j x^{\alpha_j} = f(x) \text{ for some } \alpha_i \neq \alpha_j \right\};$$

  - tropical hypersurface divides the domain of $f(x)$ into **convex cells**, on each cell $f(x)$ is **linear**;
  - number of **linear regions** of $f(x)$ is denoted by $\mathrm{lin}(f)$.
- **Goal**: Study hypersurfaces of tropical polynomials to obtain $\mathrm{lin}(f)$.
- **Newton polygon** and its **dual subdivision** will help.

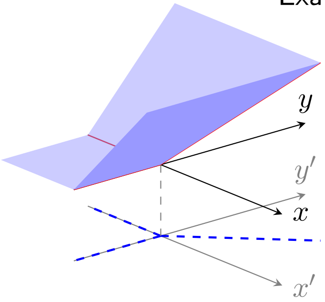
Tropical hypersurface

Definition (Tropical Hypersurface)

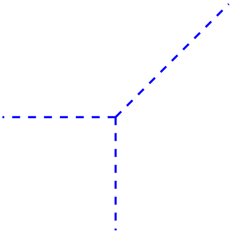
Given $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$, the tropical hypersurface of f is

$$\mathcal{T}(f) := \{x \in \mathbb{R}^d : c_i x^{\alpha_i} = c_j x^{\alpha_j} = f(x) \text{ for some } \alpha_i \neq \alpha_j\}.$$

Example: $\mathcal{T}(x \oplus y \oplus 0)$



Tropical hypersurface is the "corner locus"

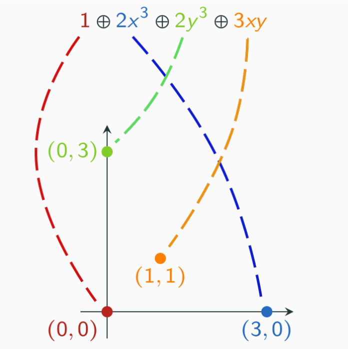


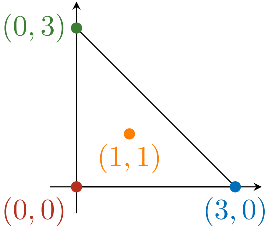
$\mathcal{T}(x \oplus y \oplus 0)$

Newton polygon

The Newton polygon of a d-variate tropical polynomial $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$ is

$$\Delta(f) \coloneqq \mathrm{Conv}\{\alpha_i \in \mathbb{R}^d : c_i \neq -\infty,\ i = 1, \ldots, r\}.$$





Dual subdivision of Newton polygon

Given a tropical polynomial $f(x) = c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}$

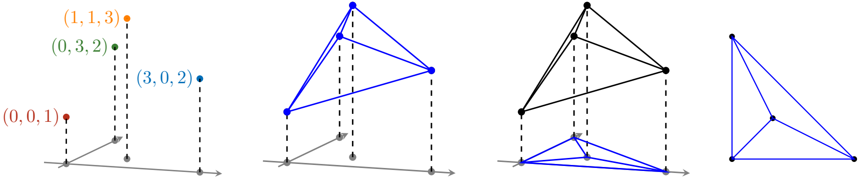
1. Lift each α_i from $\mathbb{R}^d$ into $\mathbb{R}^{d+1}$: $\{(\alpha_i, c_i) \in \mathbb{R}^d \times \mathbb{R} : i = 1, \ldots, r\}$
2. Take their **convex hull**

$$\mathcal{P}(f) \coloneqq \mathrm{Conv}\{(\alpha_i, c_i) \in \mathbb{R}^d \times \mathbb{R} : i = 1, \ldots, r\}$$

3. Define $\pi : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ by $\pi((\alpha, c)) = \alpha$. **The dual subdivision** is

$$\delta(f) \coloneqq \big\{\pi(p) \subset \mathbb{R}^d : p \in \mathrm{UF}\big(\mathcal{P}(f)\big)\big\}$$

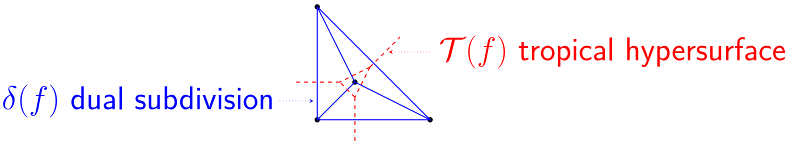
Example: $1 \oplus 2x^3 \oplus 2y^3 \oplus 3xy$



Dual subdivision and tropical hypersurface

- Tropical hypersurface $\mathcal{T}(f)$ is dual to $\delta(f)$:
  - every **vertex** in $\delta(f)$ corresponds to a **"cell" where f is linear**;
  - $\operatorname{lin}(f)$ = number of vertices on the **upper faces** of $\mathcal{P}(f)$.
- **Goal:** Count the number of vertices in the upper faces of $\mathcal{P}(F^\alpha)$.

$$1 \oplus 2x^3 \oplus 2y^3 \oplus 3xy$$



Tropical hypersurface and neural networks

An immediate result:

Proposition (Zhang–Naitzat–L 2018)

Let $\nu : \mathbb{R}^d \to \mathbb{R}$ *be an* L*-layer neural network. Write* $\nu = f \oslash g$ *then*

(i) *A decision boundary* $\mathcal{B} = \{x \in \mathbb{R}^d : \nu(x) = c\}$ *divides* $\mathbb{R}^d$ *into at most* $\operatorname{lin}(f)$ *connected regions above* c *and at most* $\operatorname{lin}(g)$ *connected regions below* c*;*

(ii) *The decision boundary is contained in the tropical hypersurface of the tropical polynomial* $(c \odot g(x)) \oplus f(x)$*, i.e.,*

$$\mathcal{B} \subseteq \mathcal{T}((c \odot g) \oplus f).$$

Tropical hypersurface and neural networks

- Once the connection has been established we want to use results of tropical geometry to study neural network.
- One of the main objects of interest is "zeros" of tropical polynomials.
- We will demonstrate how we can use results on zeros of tropical polynomial to study neural networks.

Recall that:

$$
\begin{aligned}
F_j^{(i+1)} &= H_j^{(i+1)} \oplus G_j^{(i+1)}, \\
G_j^{(i+1)} &= \left[\bigodot_{k=1}^{n} (F_k^{(i)})^{a_{jk,-}^{(i+1)}}\right] \odot \left[\bigodot_{k=1}^{n} (G_k^{(i)})^{a_{jk,+}^{(i+1)}}\right], \\
H_j^{(i+1)} &= \left[\bigodot_{k=1}^{n} (F_k^{(i)})^{a_{jk,+}^{(i+1)}}\right] \odot \left[\bigodot_{k=1}^{n} (G_k^{(i)})^{a_{jk,-}^{(i+1)}}\right] \odot b_j^{(i+1)}.
\end{aligned}
$$

Question: What are $\mathcal{P}(F^{(i+1)})$ and $\mathcal{P}(G^{(i+1)})$?

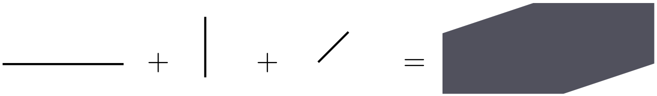
Transformation of tropical hypersurface

Let f, g be tropical polynomials.

- $\mathcal{P}(f^a) = a\mathcal{P}(f)$ (for any $a \in \mathbb{N}$)
  - $(c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r})^a = c_1 x^{a\alpha_1} \oplus \cdots \oplus c_r x^{a\alpha_r}$
- $\mathcal{P}(f \oplus g) = \mathrm{Conv}(\mathcal{P}(f) \cup \mathcal{P}(g))$
  - $(c_1 x^{\alpha_1} \oplus \cdots \oplus c_r x^{\alpha_r}) \oplus (c'_1 x^{\alpha'_1} \oplus \cdots \oplus c'_r x^{\alpha'_r}) =$ $c_1 x^{a\alpha_1} \oplus \cdots \oplus c_r x^{a\alpha_r} \oplus c'_1 x^{\alpha'_1} \oplus \cdots \oplus c'_r x^{\alpha'_r}$
- $\mathcal{P}(f \odot g) = \mathcal{P}(f) + \mathcal{P}(g)$, where "$+$" is Minkowski sum

$$P_1 + P_2 = \{x_1 + x_2 \in \mathbb{R}^d : x_1 \in P_1, x_2 \in P_2\}$$

- Minkowski sum of line segments is called a **zonotope**



Polytopes associated with neural network

- For the first layer, $v_j^{(1)} = F_j^{(1)} - G_j^{(1)}$

f	$\mathcal{P}(f)$
$G_j^{(1)} = \bigodot_{k=1}^{d} (x_k)^{a_{jk,-}^{(1)}}$	point in $\mathbb{R}^{d+1}$
$H_j^{(1)} = \left[\bigodot_{k=1}^{d} (x_k)^{a_{jk,+}^{(1)}}\right] \odot b_j^{(1)}$	point in $\mathbb{R}^{d+1}$
$F_j^{(1)} = H_j^{(1)} \oplus G_j^{(1)}$	line segment in $\mathbb{R}^{d+1}$

- For the second layer, $v_j^{(2)} = F_j^{(2)} - G_j^{(2)}$

f	$\mathcal{P}(f)$
$G_j^{(2)} = \left[\bigodot_{k=1}^{n_1} (F_k^{(1)})^{a_{jk,-}^{(2)}}\right] \odot \left[\bigodot_{k=1}^{n_1} (G_k^{(1)})^{a_{jk,+}^{(2)}}\right]$	zonotope
$H_j^{(2)} = \left[\bigodot_{k=1}^{n_1} (F_k^{(1)})^{a_{jk,+}^{(2)}}\right] \odot \left[\bigodot_{k=1}^{n_1} (G_k^{(1)})^{a_{jk,-}^{(2)}}\right] \odot b_j^{(2)}$	zonotope
$F_j^{(2)} = H_j^{(2)} \oplus G_j^{(2)}$	convex hull of two zonotopes

Polytopes associated with neural network

Lemma (Zhang–Naitzat–L 2018)

Let $F_j^{(i)}$, $G_j^{(i)}$ be the tropical polynomials produced by the jth node in the ith layer, then

- *for $i \geq 1$, $\mathcal{P}(G_j^{(i+1)})$ is weighted Minkowski sums of $\mathcal{P}(F_1^{(i)}), \dots, \mathcal{P}(F_{n_i}^{(i)}), \mathcal{P}(G_1^{(i)}), \dots, (G_{n_i}^{(i)})$, given by*

$$\mathcal{P}(G_j^{(i+1)}) = \sum_{k=1}^{n_i} a_{jk,-}^{(i+1)} \mathcal{P}(F_k^{(i)}) + \sum_{k=1}^{n_i} a_{jk,+}^{(i+1)} \mathcal{P}(G_k^{(i)});$$

- *for $i \geq 1$,*

$$\mathcal{P}(F_j^{(i)}) = \mathrm{Conv}\big[\mathcal{P}(G_j^{(i)}) \cup \mathcal{P}(H_j^{(i)})\big].$$

Vertices on upper faces of zonotopes

Theorem (Gritzmann–Sturmfels)

Let $P_1, \ldots, P_k$ be polytopes in $\mathbb{R}^d$ and let m denote the total number of nonparallel edges of $P_1, \ldots, P_k$. Then the number of vertices of $P_1 + \cdots + P_k$ does not exceed

$$2\sum_{j=0}^{d-1}\binom{m-1}{j}.$$

Corollary (Zhang–Naitzat–L 2018)

Let $P \subset \mathbb{R}^{d+1}$ be a zonotope generated by n line segments. Then P has ***at most***

$$\sum_{j=0}^{d}\binom{n}{j}$$

vertices on its upper faces.

Linear regions of the neural network

Study of tropical hypersurfaces leads to the following result:

Theorem (Zhang–Naitzat–L 2018)

Let $\nu : \mathbb{R}^d \to \mathbb{R}^{n_L}$ *be an* L*-layer neural network with layers* $\nu^{(l)} = F^{(l)} - G^{(l)}$ *and let* $n_l \geq d$ *for all* $l = 0, \ldots, L$*, then*

$$\operatorname{lin}(\nu^{(l+1)}) \leq \operatorname{lin}(\nu^{(l)}) \cdot \sum_{i=0}^{d} \binom{n_{l+1}}{i}.$$

Note that $\operatorname{lin}(\nu^{(0)}) = 1$*.*

Here $\operatorname{lin}(f)$ is the number of linear regions of f.

Number of linear regions

Corollary (Raghu et al. 2017, Zhang–Naitzat–L 2018)

Assume $n_i \geq d, i = 1, \dots, L-1$ *and* $n_L = 1$. *The number of linear regions of an* L*-layer ReLU neural network does not exceed*

$$\prod_{i=1}^{L-1} \sum_{j=0}^{d} \binom{n_i}{j} \sim \mathcal{O}(n^{d(L-1)}) \text{ when } n_1 = \cdots = n_{L-1} = n.$$

This upper bound is (almost) tight [Montufar et al. 2014, Raghu et al. 2017].

Summary

- A ReLU neural network with integer coefficients = tropical rational map
$$\nu(x) = f(x) \oslash g(x) \equiv f(x) - g(x).$$
- The geometry of two-layer ReLU networks is the geometry of zonotopes.
- The geometry of L-layer ReLU networks is the geometry of the polytopes that produce dual subdivision of Newton polygons of tropical polynomials.
- Tropical geometry allows us to count the number of linear regions of ReLU networks.
- Deeper networks have exponentially more linear regions than shallow networks.