Maxpolynomial Division with Application to Neural Network Simplification

Georgios Smyrnis[1,2] Petros Maragos[1,2] George Retsinas[1]

[1]School of ECE, National Technical University of Athens, Athens, Greece
[2]Robot Perception and Interaction Unit, Athena Research Center, Maroussi, Greece

May 2020

Outline

Basics of Max-plus Algebra

Max-Plus Algebra

Max-plus (or tropical) algebra is defined as the algebra performed on the max-plus semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$.

Basics of Max-plus Algebra

Max-Plus Algebra

Max-plus (or tropical) algebra is defined as the algebra performed on the max-plus semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$.

Maxpolynomials

Maxpolynomials (or tropical polynomials) are polynomials which are formed using the operations of the max-plus semiring:

$$p(\boldsymbol{x}) = \max_{i=1}^{k} \left(\boldsymbol{a}_i^T \boldsymbol{x} + b_i \right), \boldsymbol{x} \in \mathbb{R}^d$$

Maxpolynomial Examples

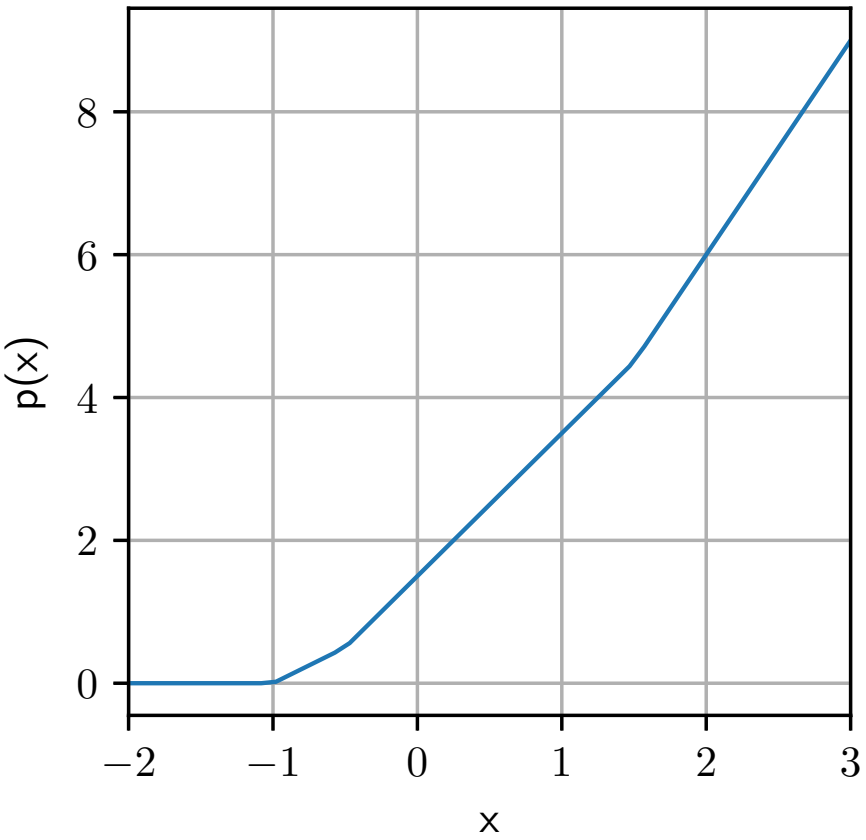

Figure 1:
$p(x) = \max(3x, 2x + 1.5, x + 1, 0)$

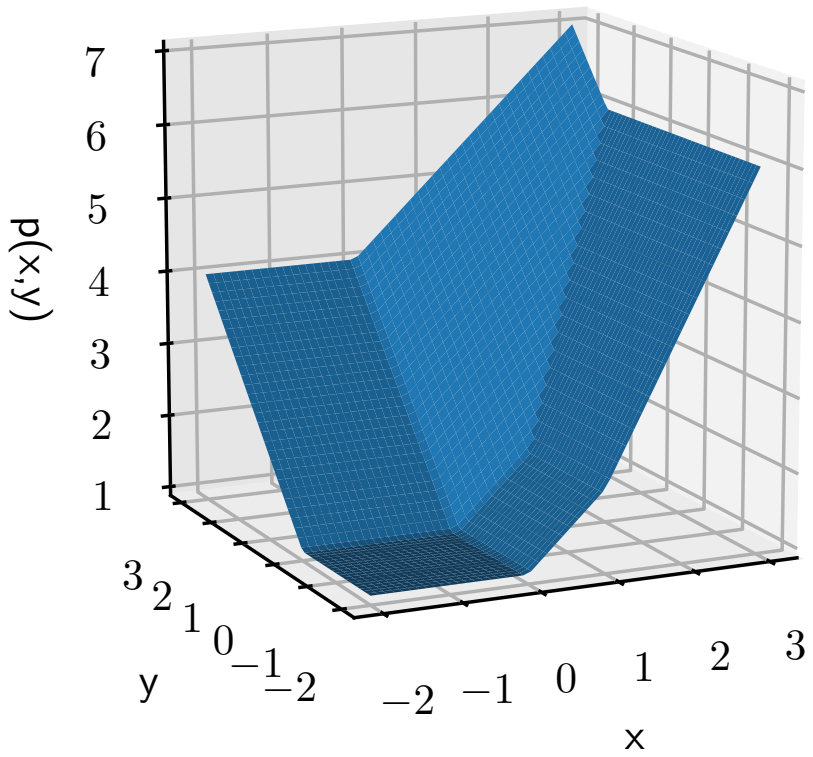

Figure 2:
$p(x, y) = \max(2x, x+y+1, x+1, y+1, 1)$

(Extended) Newton Polytope

Let $p(\boldsymbol{x}) = \max_{i=1}^{k} \left(\boldsymbol{a}_i^T \boldsymbol{x} + b_i\right)$ be a maxpolynomial.

(Extended) Newton Polytope

Let $p(\boldsymbol{x}) = \max_{i=1}^{k} \left(\boldsymbol{a}_i^T \boldsymbol{x} + b_i\right)$ be a maxpolynomial.

Definition 1 ((Extended) Newton Polytope)

We define as the (Extended) Newton Polytope of p the following

$$\mathrm{Newt}(p) = \mathrm{conv}\left\{\boldsymbol{a}_i, i = 1, \ldots, k\right\}$$
$$\mathrm{ENewt}(p) = \mathrm{conv}\left\{(\boldsymbol{a}_i, b_i), i = 1, \ldots, k\right\}$$

where conv signifies the convex hull of the given set.

(Extended) Newton Polytope

Let $p(\boldsymbol{x}) = \max_{i=1}^{k} \left(\boldsymbol{a}_i^T \boldsymbol{x} + b_i\right)$ be a maxpolynomial.

Definition 1 ((Extended) Newton Polytope)

We define as the (Extended) Newton Polytope of p the following

$$\mathrm{Newt}(p) = \mathrm{conv}\left\{\boldsymbol{a}_i, i = 1, \ldots, k\right\}$$
$$\mathrm{ENewt}(p) = \mathrm{conv}\left\{(\boldsymbol{a}_i, b_i), i = 1, \ldots, k\right\}$$

where conv signifies the convex hull of the given set.

Theorem 2 ([Charisopoulos and 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 Polytopes

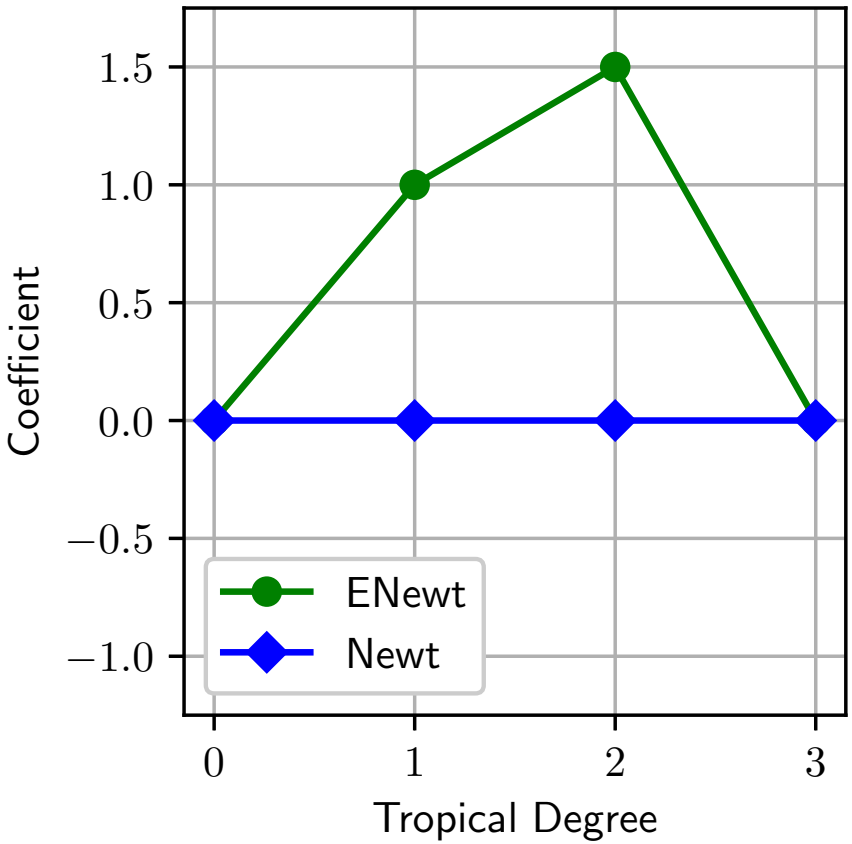

Figure 3: Polytopes of $\max(3x, 2x + 1.5, x + 1, 0)$.

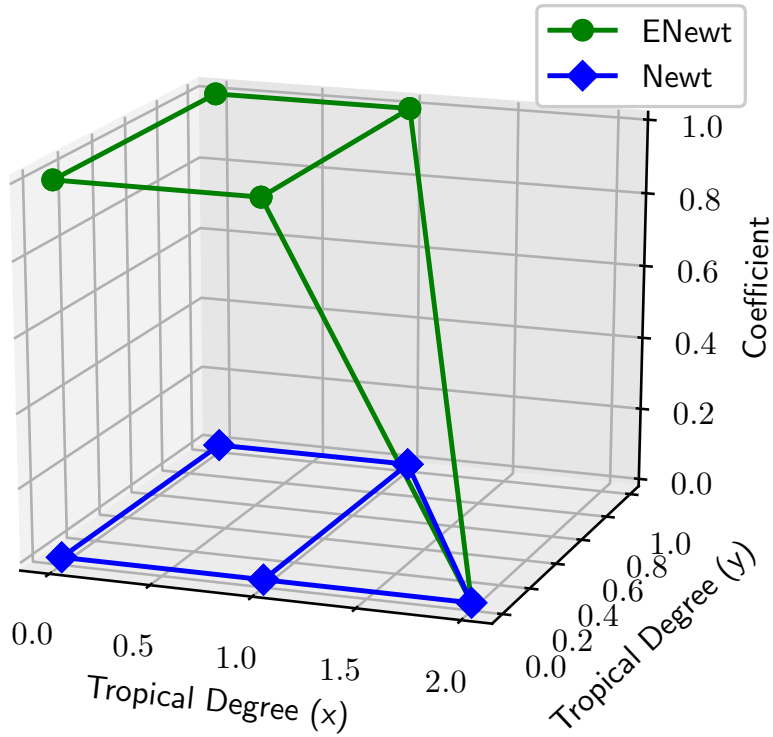

Figure 4: Polytopes of $\max(2x, x + y + 1, x + 1, y + 1, 1)$

Maxpolynomial Division

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

$$p(\boldsymbol{x}) = \max\left(q(\boldsymbol{x}) + d(\boldsymbol{x}), r(\boldsymbol{x})\right)$$

Maxpolynomial Division

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

$$p(\boldsymbol{x}) = \max\left(q(\boldsymbol{x}) + d(\boldsymbol{x}), r(\boldsymbol{x})\right)$$

Problem!

The above is not always feasible!

Approximate Maxpolynomial Division

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

$$p(\boldsymbol{x}) \geq \max\left(q(\boldsymbol{x}) + d(\boldsymbol{x}), r(\boldsymbol{x})\right)$$

Approximate Maxpolynomial Division

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

$$p(\boldsymbol{x}) \geq \max\left(q(\boldsymbol{x}) + d(\boldsymbol{x}), r(\boldsymbol{x})\right)$$

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

Algorithm for Approximate Maxpolynomial Division

1. Let C be the set of possible vectors $\boldsymbol{c}$ by which we can shift $\mathrm{Newt}(d)$ (each of which corresponds to a term in q).

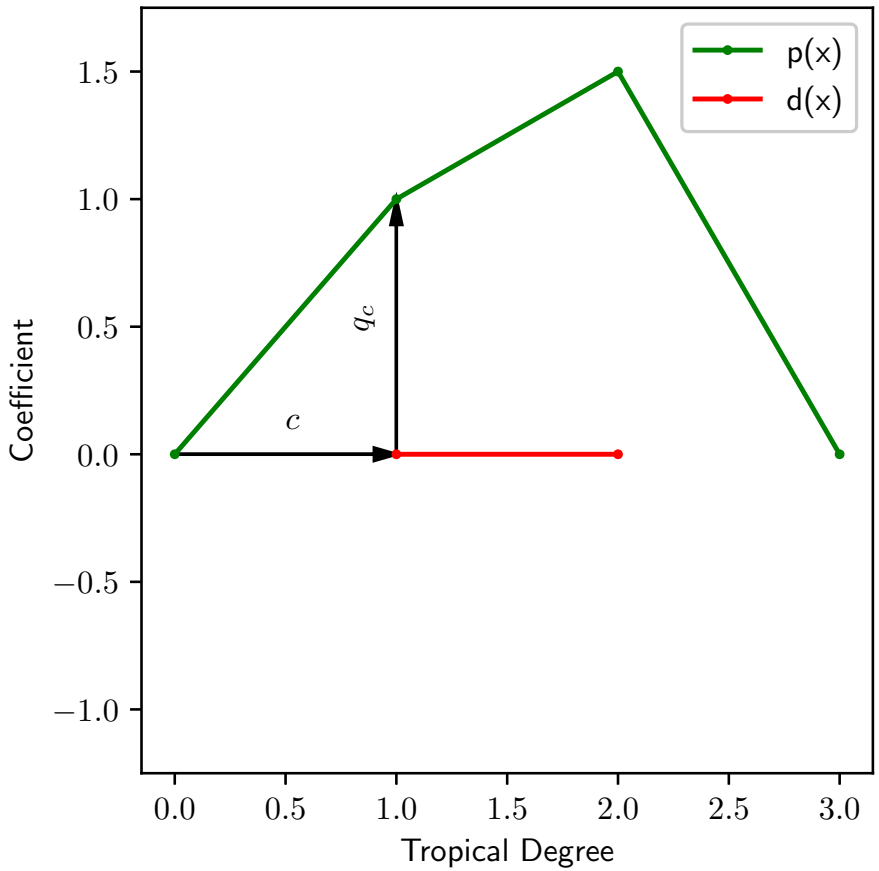

Figure 5: Division Method.

Algorithm for Approximate Maxpolynomial Division

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

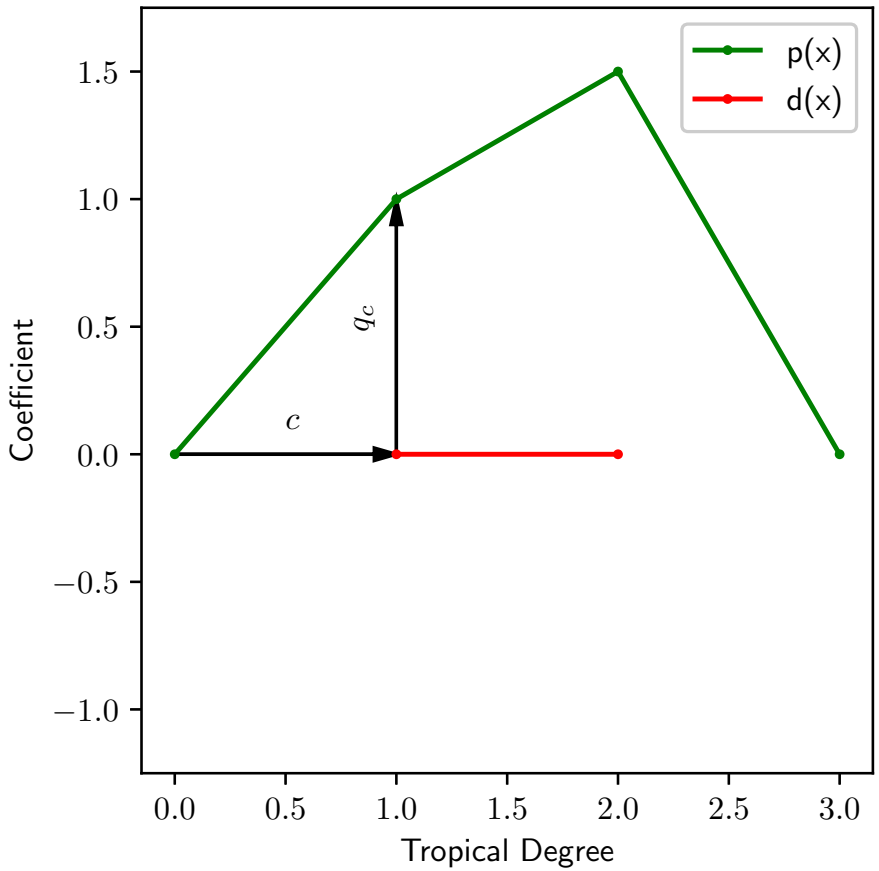

Figure 5: Division Method.

Algorithm for Approximate Maxpolynomial Division

1. Let C be the set of possible vectors $\boldsymbol{c}$ by which we can shift $\mathrm{Newt}(d)$ (each of which corresponds to a term in q).

2. We raise the shifted version of $\mathrm{ENewt}(d)$ as high as possible so that it still lies below $\mathrm{ENewt}(p)$, and we mark the vertical shift as q_c.

3. We set the quotient equal to:

$$q(\boldsymbol{x}) = \max_{\boldsymbol{c} \in C} \left(q_{\boldsymbol{c}} + \boldsymbol{c}^T \boldsymbol{x} \right)$$

and add all terms not covered by a shift $\boldsymbol{c}$ to the remainder r.

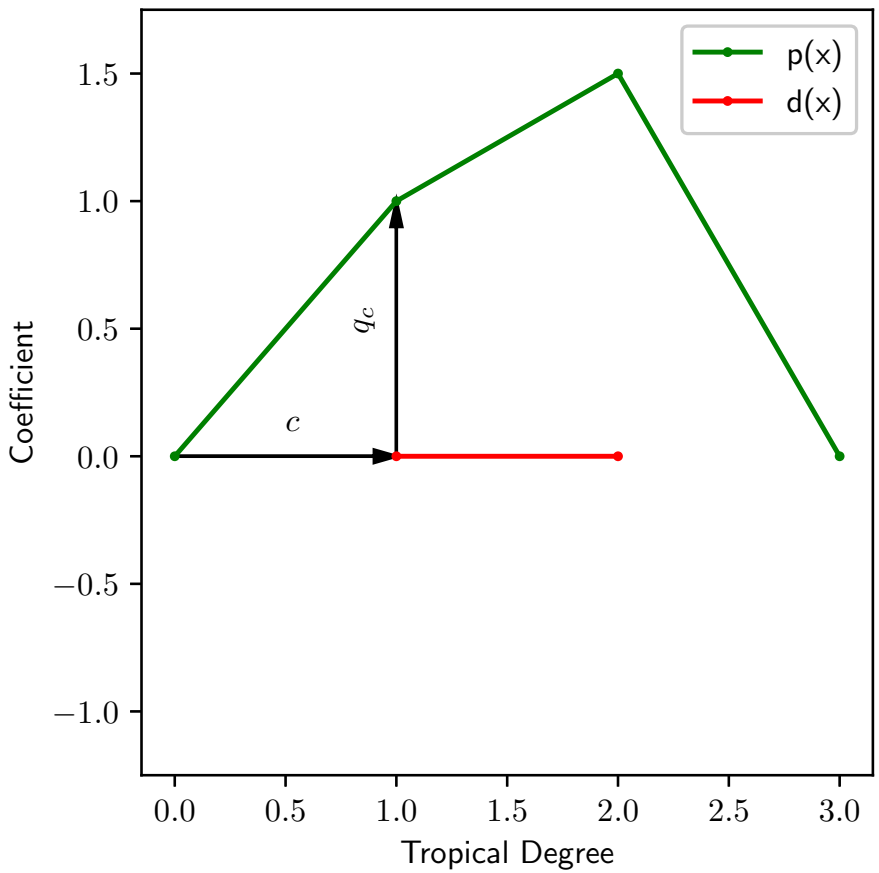

Figure 5: Division Method.

Division Example (1)

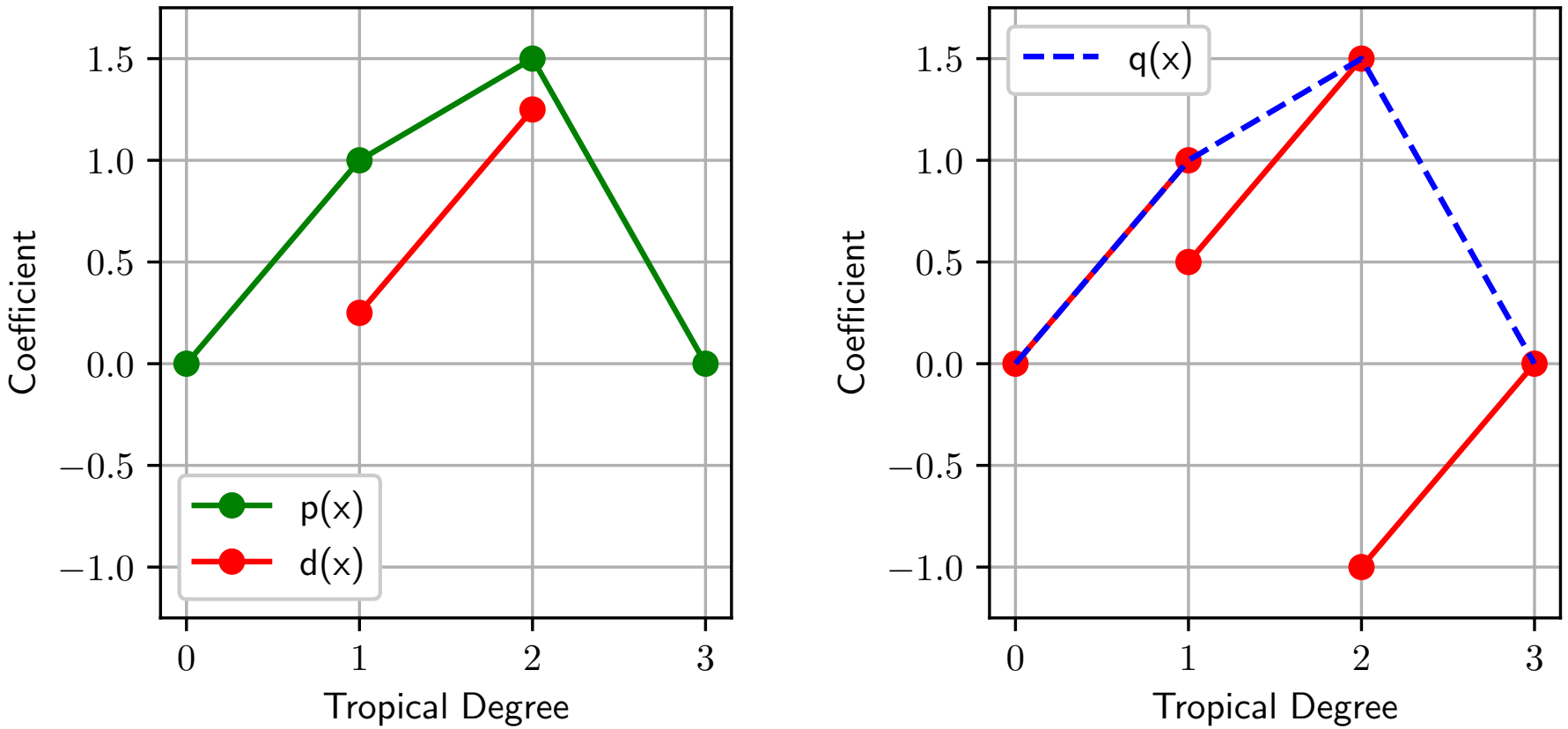

Figure 6: Division of $p(x) = \max(3x, 2x + 1.5, x + 1, 0)$ by $d(x) = \max(x + 1, 0)$.

Division Example (2)

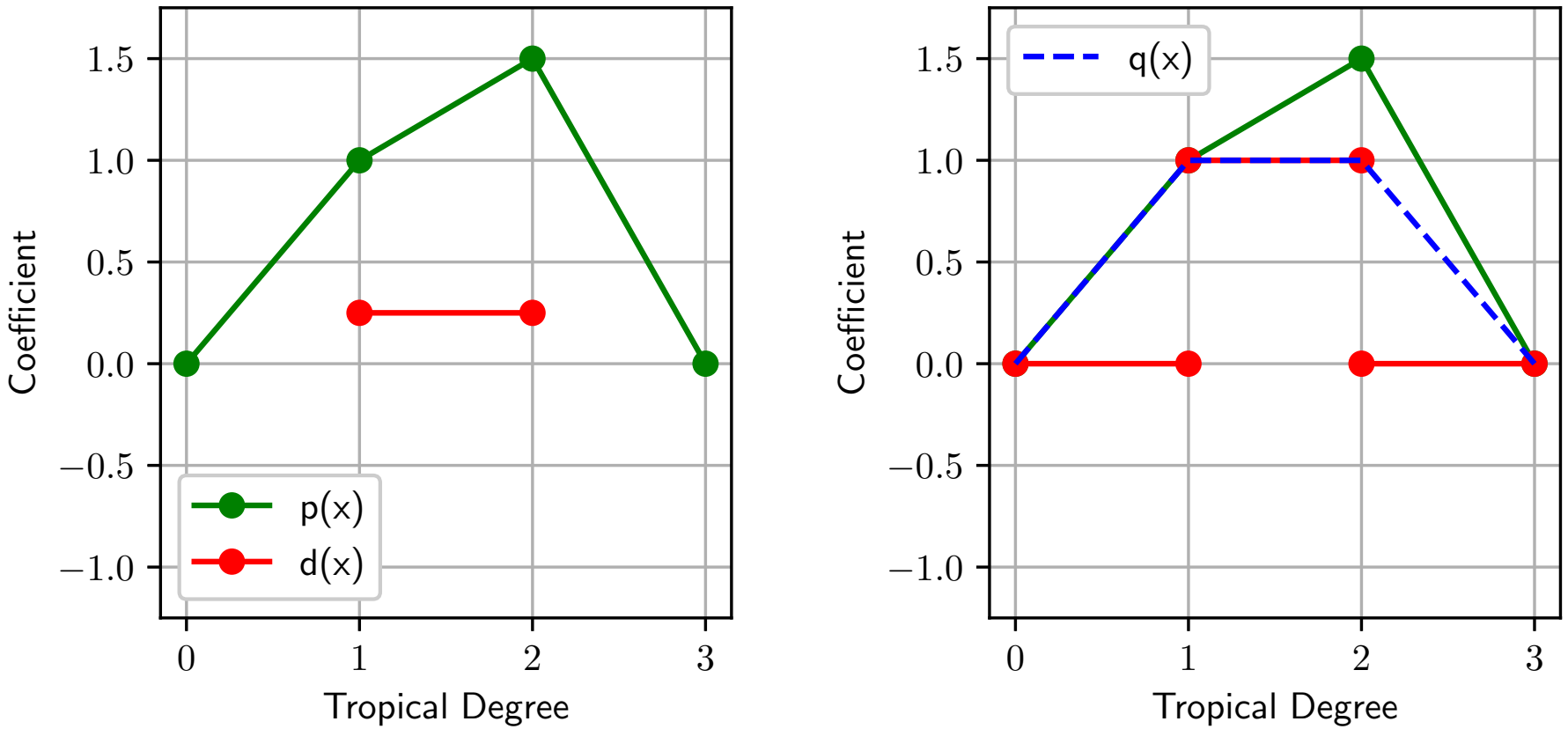

Figure 7: Division of $p(x) = \max(3x, 2x + 1.5, x + 1, 0)$ by $d(x) = \max(x, 0)$.

Application to Neural Network Minimization (1)

We shall examine neural networks with ReLU activations, focusing on simple networks with two layers and one output neuron.

Application to Neural Network Minimization (1)

We shall examine neural networks with ReLU activations, focusing on simple networks with two layers and one output neuron.

Theorem 3 ([Wang, 2004])

A continuous, piecewise linear function is equal to the difference of two maxpolynomials.

Application to Neural Network Minimization (1)

We shall examine neural networks with ReLU activations, focusing on simple networks with two layers and one output neuron.

Theorem 3 ([Wang, 2004])

A continuous, piecewise linear function is equal to the difference of two maxpolynomials.

Theorem 4 ([Zhang et al., 2018])

A neural network with piecewise linear activations can be represented as the difference of two maxpolynomials, or in other words a tropical rational function.

Application to Neural Network Minimization (2)

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

This can be thought of as lifting the inequality restriction of maxpolynomial division, which might be limiting when approximating a network.

Neural Network Minimization Algorithm (1)

For each of the maxpolynomials p of the network, we first find a divisor. This is done by:

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

Neural Network Minimization Algorithm (1)

For each of the maxpolynomials p of the network, we first find a divisor. This is done by:

1. Finding the most important vertices of $\mathrm{ENewt}(p)$, via the weights of the network (based on which combination of neurons is activated).
2. Incrementally adding the differences of these vertices as neurons in the new network.

Neural Network Minimization Algorithm (1)

For each of the maxpolynomials p of the network, we first find a divisor. This is done by:

1. Finding the most important vertices of $\mathrm{ENewt}(p)$, via the weights of the network (based on which combination of neurons is activated).
2. Incrementally adding the differences of these vertices as neurons in the new network.

Afterwards, we find a quotient, via the average difference in activations between the original network and the new network, and add it to the output bias.

Neural Network Minimization Algorithm (2)

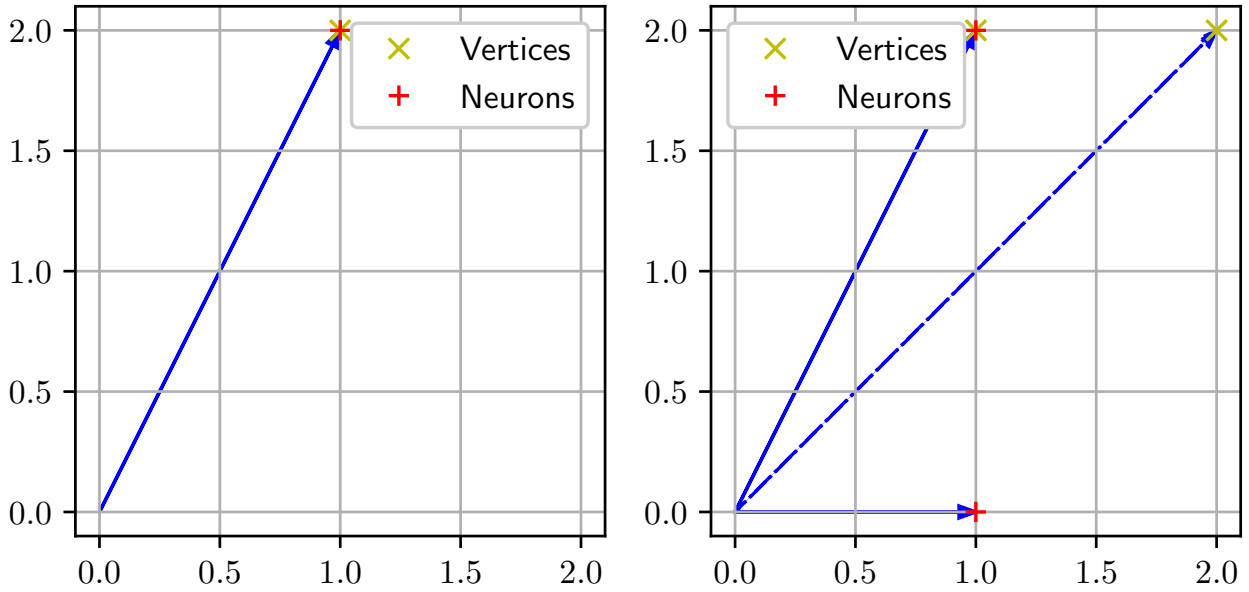

Figure 8: Example of Iterative Step.

The intuition behind the algorithm is that, by adding the difference between each new vertex and a previous neuron, the resulting sum will equal the vertex, and might also be a vertex in the new polytope.

Experimental Application

We train networks for these two datasets:

- IMDB Movie Review (using 2000 samples for our method).
- MNIST (where only pairs of classes 3/5 and 4/9 are considered).

Experimental Application

We train networks for these two datasets:

- IMDB Movie Review (using 2000 samples for our method).
- MNIST (where only pairs of classes 3/5 and 4/9 are considered).

Our networks consist of two fully connected layers in the end. Before these we use as feature extractors:

- IMDB: An embedding layer, either alone or with an LSTM/1D-CNN.
- MNIST: A 2D-CNN.

Experimental Application

We train networks for these two datasets:

- IMDB Movie Review (using 2000 samples for our method).
- MNIST (where only pairs of classes 3/5 and 4/9 are considered).

Our networks consist of two fully connected layers in the end. Before these we use as feature extractors:

- IMDB: An embedding layer, either alone or with an LSTM/1D-CNN.
- MNIST: A 2D-CNN.

We minimize the hidden layer of the fully connected part of each network.

Results - IMDB Dataset (1)

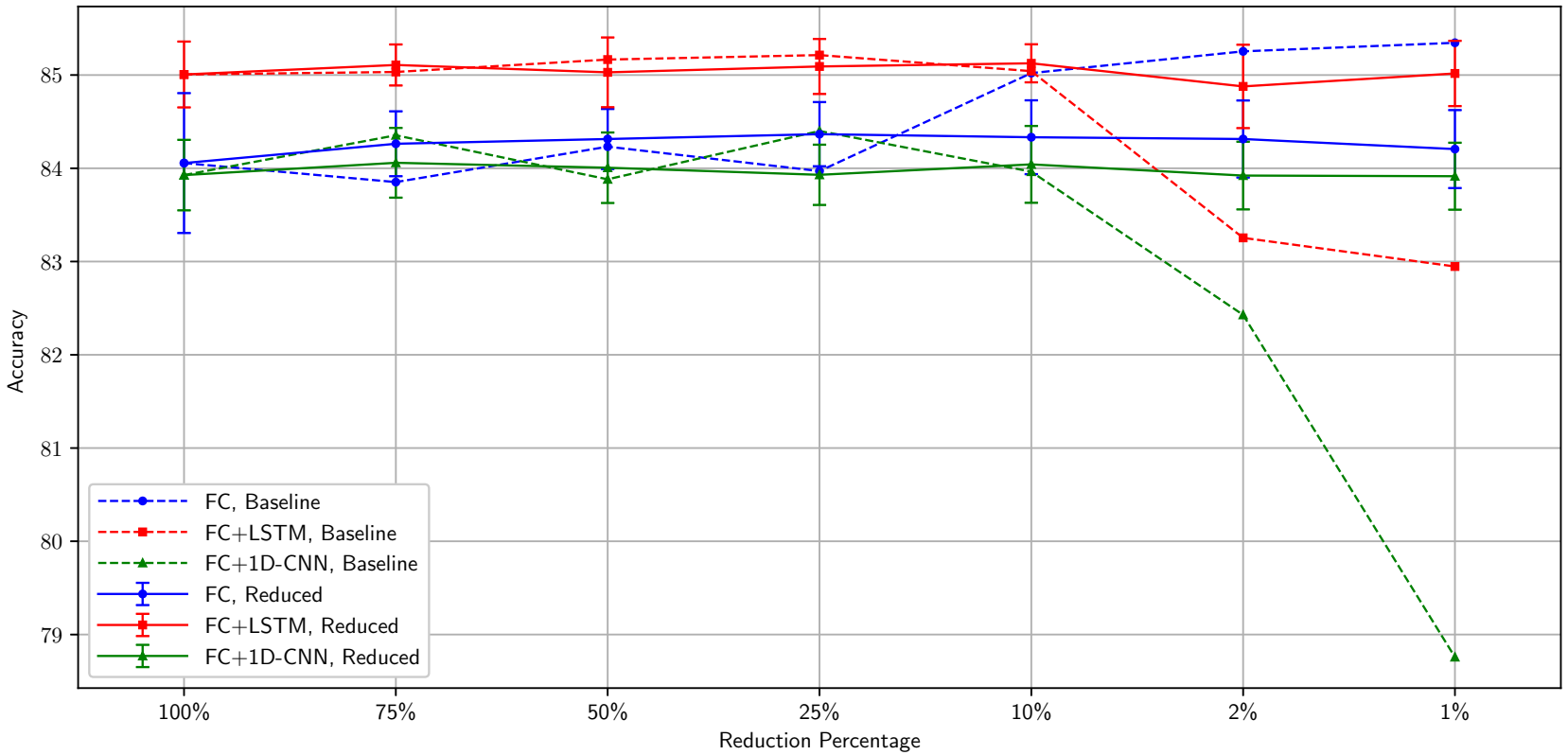

Figure 9: Average accuracy and standard deviation of method on test set (IMDB). Baseline is discarding original model and training new one from scratch. FC is the model with only an embedding layer as feature extractor, while FC+LSTM/1D-CNN also add the appropriate feature extractor.

Results - IMDB Dataset (2)

Model	Runtime of our method with st. dev. (sec)	Training time for all baseline networks (sec)
FC	82.6±4.6	97.2
FC+LSTM	113.4±0.1	630.2
FC+1D-CNN	17.8±0.8	142.3

Table 1: Time taken to train models of all sizes.
Baseline is discarding original model and training new one from scratch.
FC is the model with only an embedding layer as feature extractor, while FC+LSTM/1D-CNN also add the appropriate feature extractor.

Results - MNIST Dataset

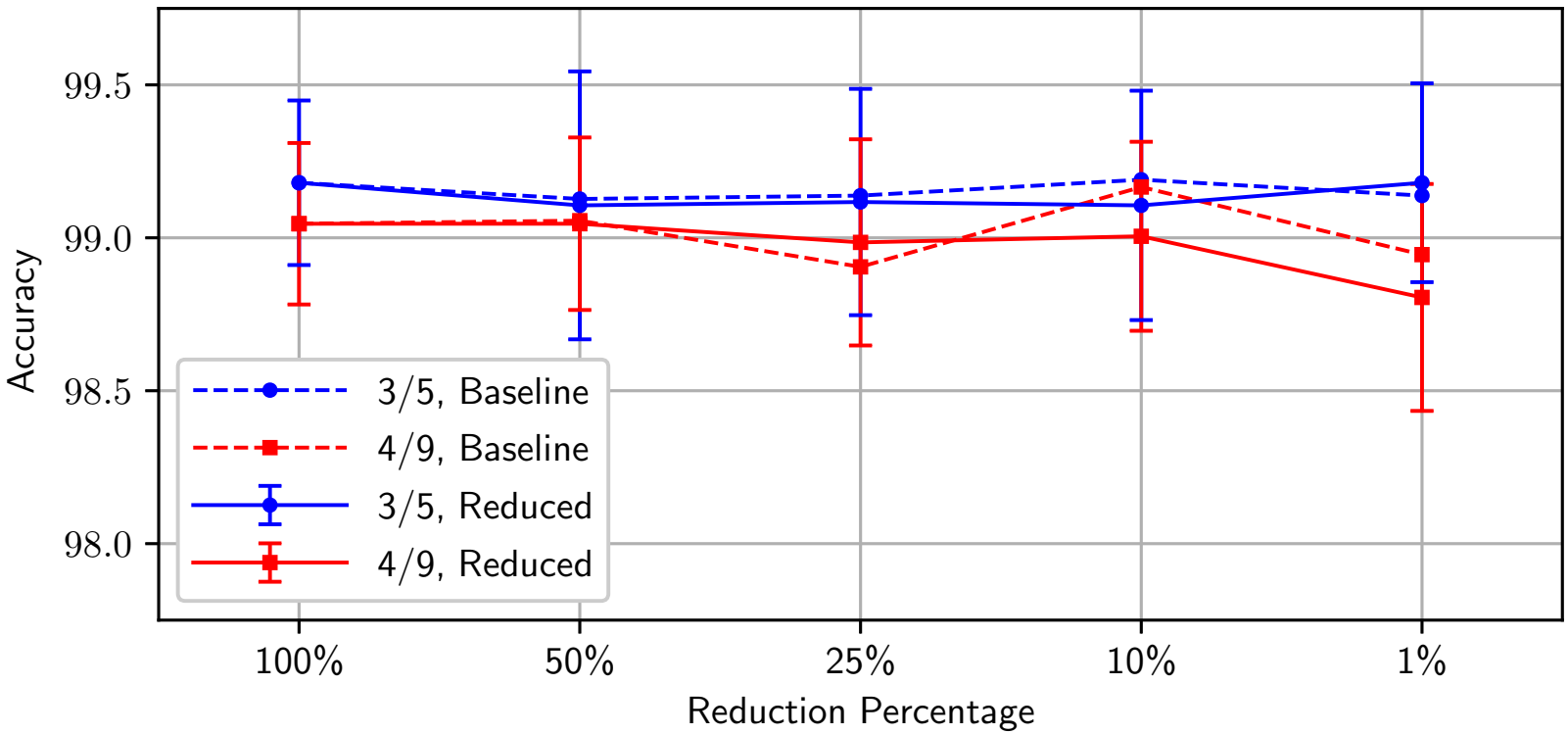

Figure 10: Average accuracy and standard deviation of method on test set (MNIST). Baseline is discarding original model and training new one from scratch.

Conclusions & Future Work

- We have made contributions in the link between neural networks and max-plus algebra, while also introducing a framework for approximate maxpolynomial division.

Conclusions & Future Work

- We have made contributions in the link between neural networks and max-plus algebra, while also introducing a framework for approximate maxpolynomial division.
- We also provided an algorithm to minimize simple, two-layer networks with a single output neuron.

Conclusions & Future Work

- We have made contributions in the link between neural networks and max-plus algebra, while also introducing a framework for approximate maxpolynomial division.
- We also provided an algorithm to minimize simple, two-layer networks with a single output neuron.
- Further research focuses on expanding this algorithm in the case of multiple class problems, as well as more complicated networks.

References

Charisopoulos, V. and Maragos, P. (2017).
Morphological Perceptrons: Geometry and Training Algorithms.
In *Proc. Int'l Symp. Mathematical Morphology (ISMM)*, volume 10225 of *LNCS*, pages 3–15. Springer, Cham.

Charisopoulos, V. and Maragos, P. (2018).
A tropical approach to neural networks with piecewise linear activations.
arXiv preprint arXiv:1805.08749.

Wang, S. (2004).
General Constructive Representations for Continuous Piecewise-Linear Functions.
IEEE Trans. Circ. Syst.-I: Regular Papers, 51(9):1889–1896.

Zhang, L., Naitzat, G., and Lim, L.-H. (2018).
Tropical geometry of deep neural networks.
In *Proc. Int'l Conf. on Machine Learning*, volume 80, pages 5824–5832. PMLR.

Thank you for your attention!

We wish everyone courage and health during the COVID-19 pandemic.

For more information, demos, and current results:
http://cvsp.cs.ntua.gr and http://robotics.ntua.gr