On Optimizing Back-Substitution Methods for Neural Network Verification

Tom Zelazny , Haoze Wu , Clark Barrett , and Guy Katz

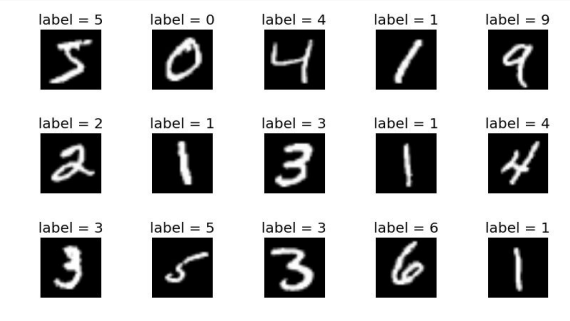
Introduction and background

Neural Networks

- ❏ Show state-of-the-art performance on many tasks.
- ❏ Are quickly being adopted in many applications.
- ❏ Including safety critical applications.

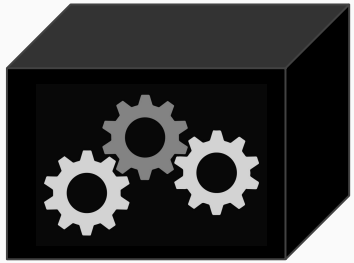
Neural Networks

Neural networks are artifacts learned from data.



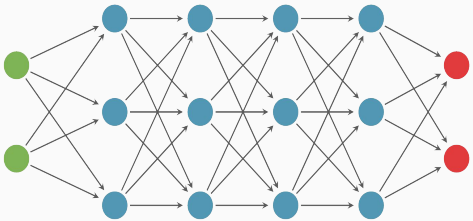
input/output pairs

in



machine learning algorithm

out



Neural network

Neural Networks

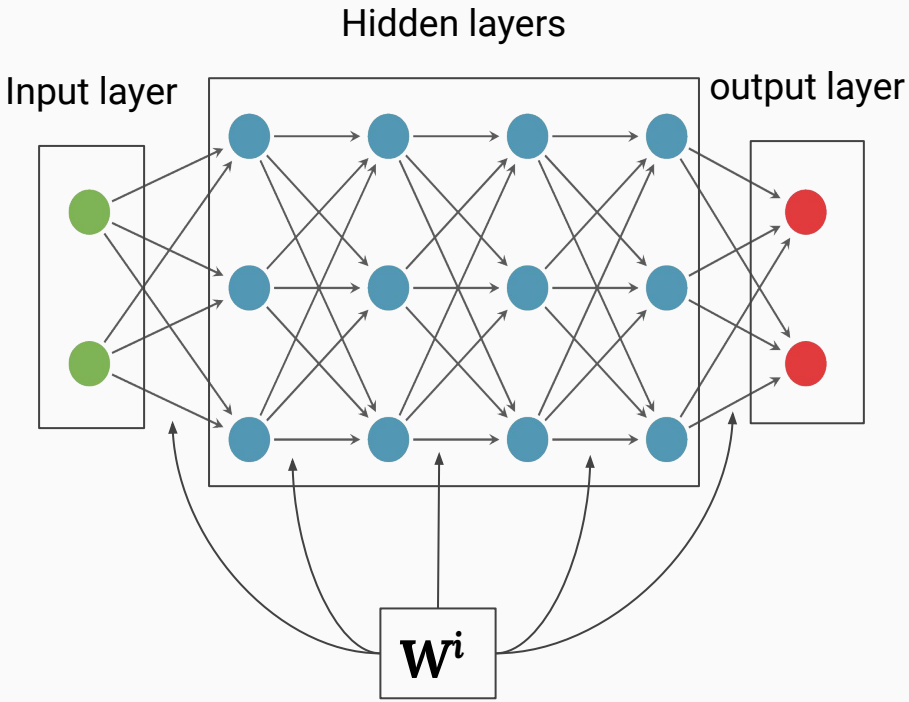
In this presentation and example we discuss a fully connected feed forward neural network with ReLU activations and ***k+1*** layers.
Concretely, a Neural network function: $\mathrm{N} : \mathbb{R}^m \rightarrow \mathbb{R}^n$, *is defined recursively as:*

$$\begin{cases} N_0(\boldsymbol{x}) = x \\ N_i(\boldsymbol{x}) = \sigma(W^{i-1} N_{i-1}(\boldsymbol{x}) + b^{i-1}) \quad 1 \leq i \leq k \end{cases}$$

Where $\mathrm{N}_i(x)$ denotes the values of the neurons in the i^{th} layer ($0 \leq i \leq k$), $\mathrm{N}(x) \coloneqq \mathrm{N}_k(x)$. $\boldsymbol{W^{i-1}}$ is a weight matrix, $\boldsymbol{b^{i-1}}$ is a bias vector and $\boldsymbol{\sigma}$ is defined as:

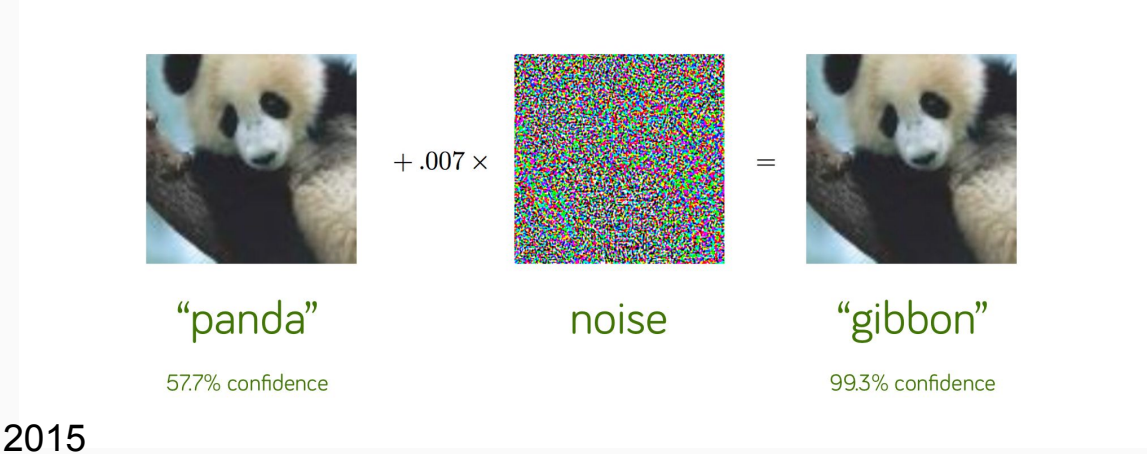
$$\begin{cases} \sigma(x) \equiv \mathrm{ReLU}(x) & \text{for } i < k \\ \sigma(x) \equiv I(x) = x & \text{for } i = k \end{cases}$$

What is a neural network



Neural Network verification

- ❏ Neural networks can fail in very unpredictable ways.
- ❏ In safety critical uses, we want guarantees.
- ❏ Traditional tools are incompatible.
- ❏ The verification problem is hard (NP-complete).



Goodfellow et al., 2015

Neural Network verification problem

Given a neural network $\mathrm{N}: x \longrightarrow y$, an input domain $\mathcal{D}_i \boldsymbol{\epsilon} \mathbb{R}^m$ and an output region $\mathcal{D}_o \boldsymbol{\epsilon} \mathbb{R}^n$, determine if for all $x_0 \in \mathcal{D}_i$ we have $\mathrm{N}(x_0) \in \mathcal{D}_o$?

Bounds on neurons

A key component in many state-of-the-art verification tools is computing lower and upper bounds on the values that neurons in the network can obtain for a specific input domain

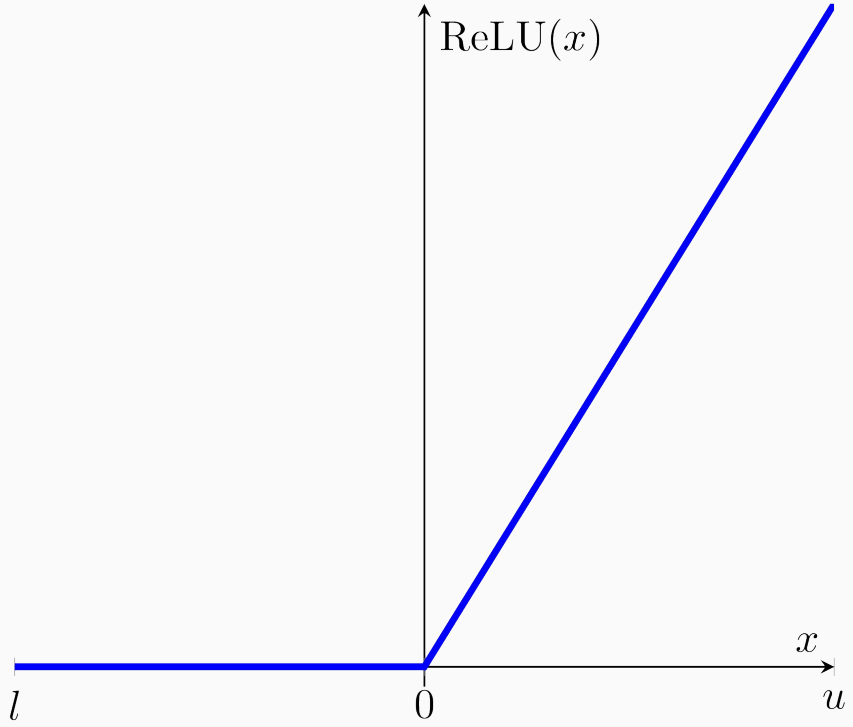
Neural Network verification problem

We assume:

- Neural network has a **single** output neuron
- Is **fully connected, feed forward** and with **ReLU activations**
- The verification problem can be reduced to finding the minimum and/or maximum values for the output neuron

$$\min_{\boldsymbol{x} \in \mathcal{D}_i} (N(\boldsymbol{x})) \qquad \max_{\boldsymbol{x} \in \mathcal{D}_i} (N(\boldsymbol{x}))$$

ReLU definition reminder



$$\mathrm{ReLU}(x) = \max(0, x)$$

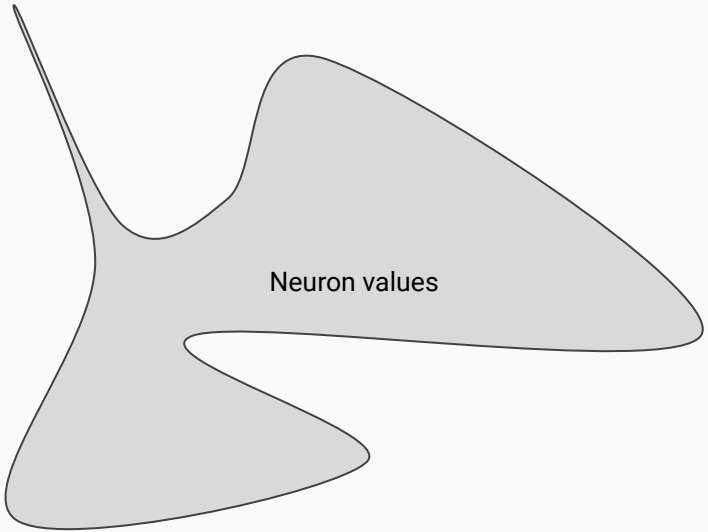
Neural Network verification problem

Solution(?): encode as a mixed integer programming (MIP) instance, and solve using a MIP solver.

Effective for smaller neural networks, but does not scale well.

Abstraction

Neural networks are non-convex, finding the minimum or maximum values of a neuron is generally hard

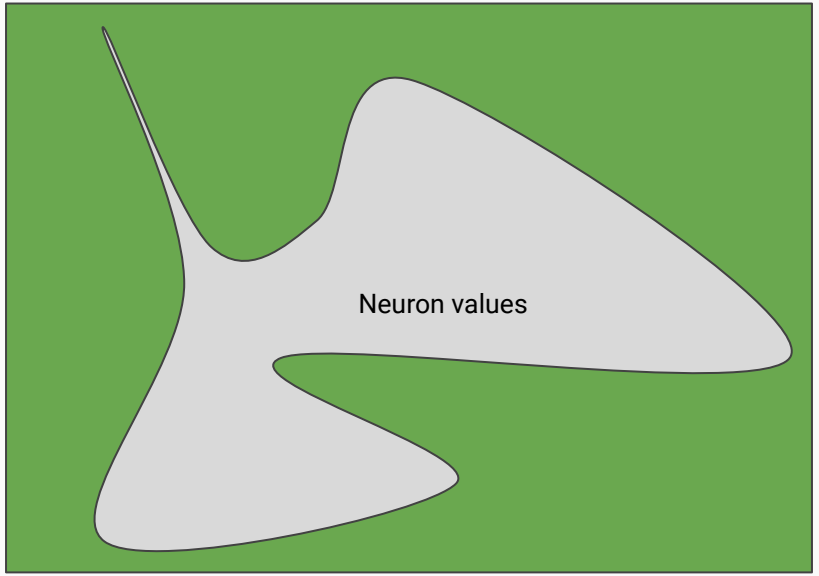


Abstraction

Over-approximation can obtain lower and upper bounds.

Green Box is easier to reason about and contains all values the neuron may obtain.

Also contains other values, so we can only use it to infer lower and upper bounds.



Agenda

We are going to go see:

- ❏ Interval arithmetic
- ❏ Back-substitution (SoTA)
- ❏ Our optimization

Interval Arithmetic

A simple type of such an abstraction is ***Interval Arithmetic***

Interval arithmetic

for x given as a linear sum of bounded variables y_j:

$$x = \sum_j c_j \cdot y_j \qquad \text{for all } j\text{: } y_j \in [l_j, u_j]$$

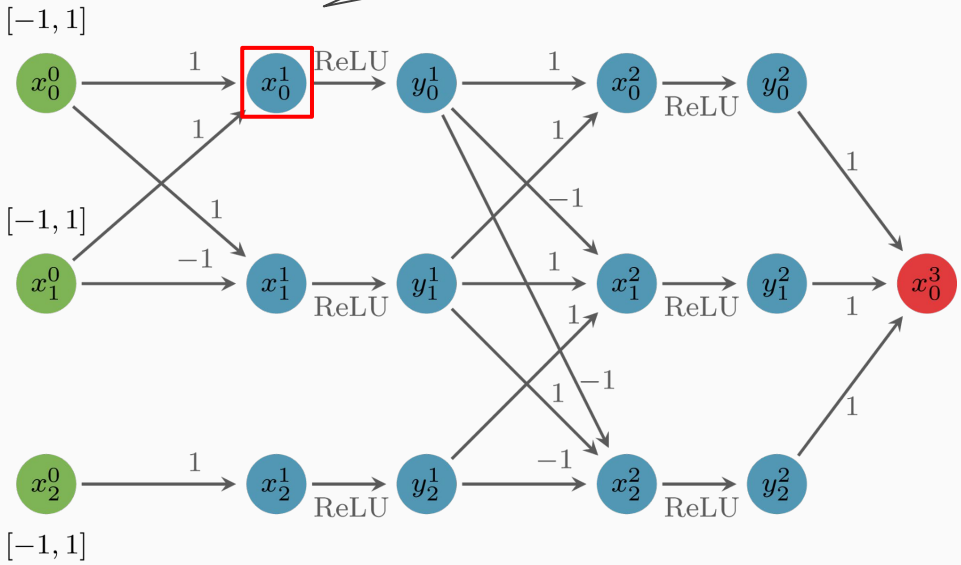
We can obtain a concrete bound for x:

$$x = \sum_j c_j \cdot y_j \leq \sum_{j:c_j<0} c_j \cdot l_j + \sum_{j:c_j>0} c_j \cdot u_j$$

Interval arithmetic - example

$$\max(x_0^1) = \max(x_0^0 + x_1^0) \leq 1 + 1 = 2$$

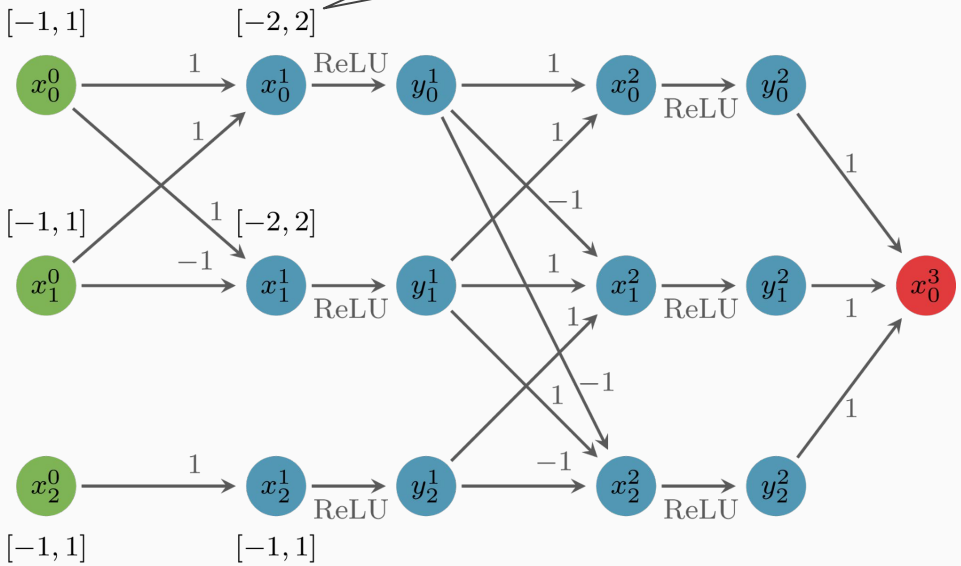
$$\mathcal{D}_i = \{-1 \leq x_i^0 \leq 1 \mid i \in \{0, 1, 2\}\}$$



$$\mathcal{D}_o \equiv [0, 6\frac{1}{5}]$$

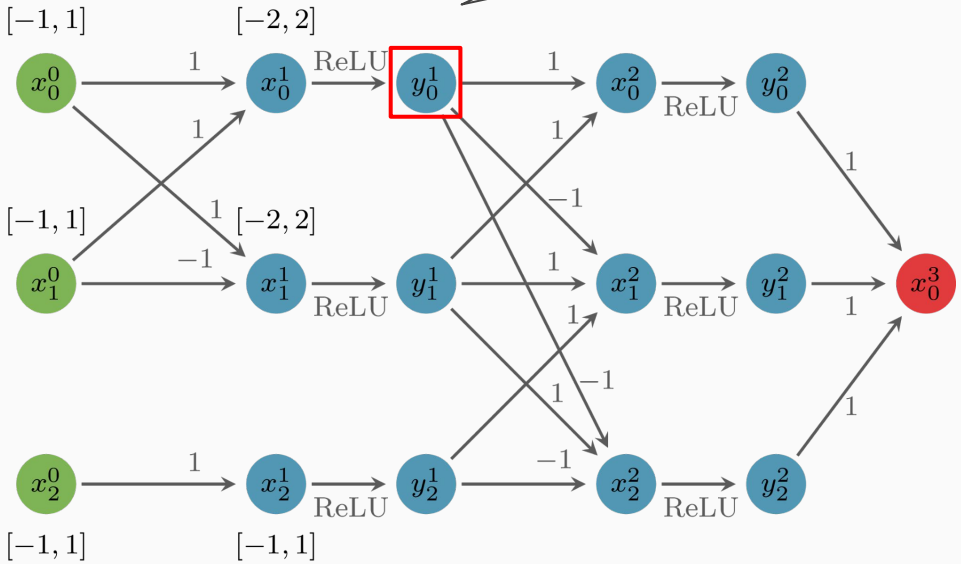
Interval arithmetic - example

$$\max(x_0^1) = \max(x_0^0 + x_1^0) \leq 1 + 1 = 2$$



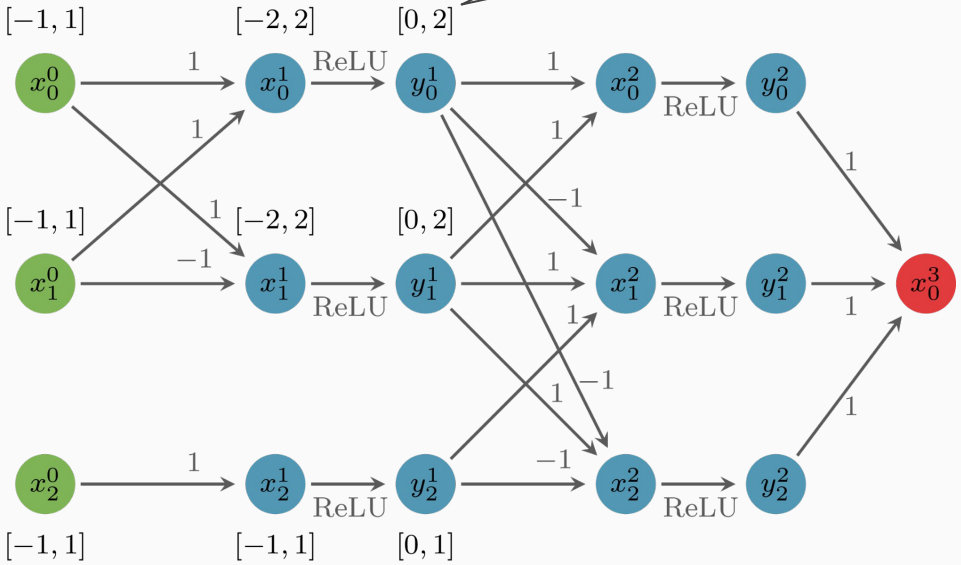
Interval arithmetic - example

$$y_0^1 = \sigma(x_0^1)$$



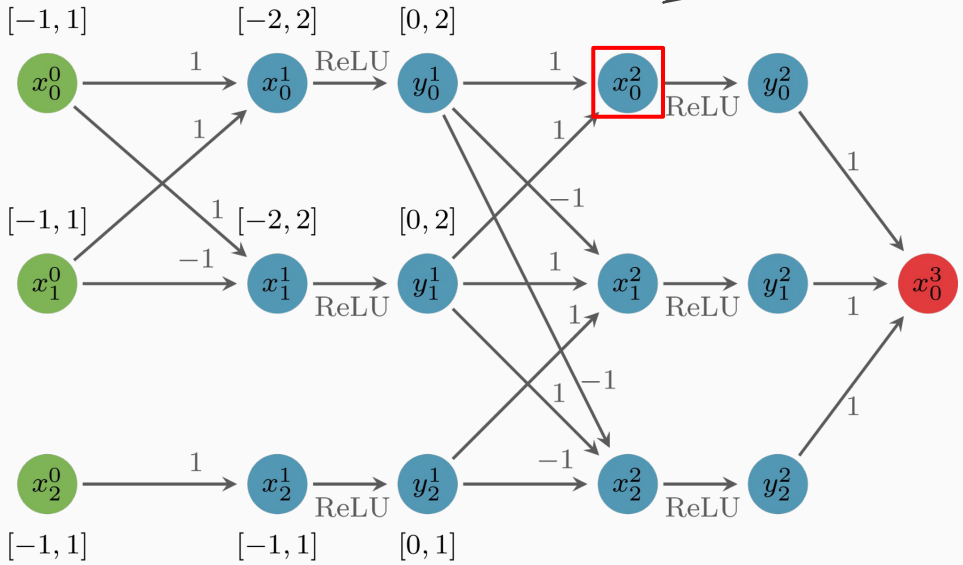
Interval arithmetic - example

$$y_0^1 = \sigma(x_0^1)$$



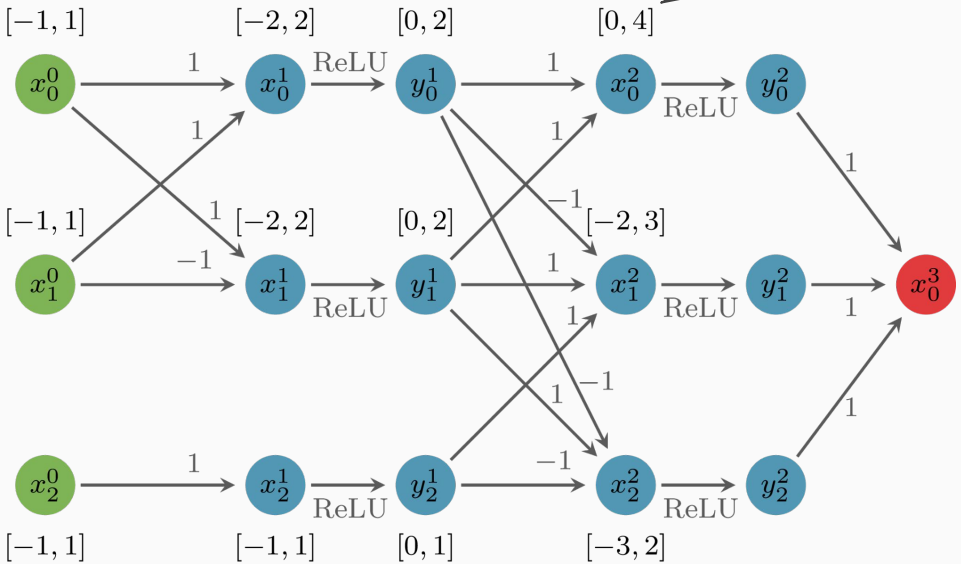
Interval arithmetic - example

$$\max(x_0^2) = \max(y_0^1 + y_1^1) \leq 2 + 2 = 4$$



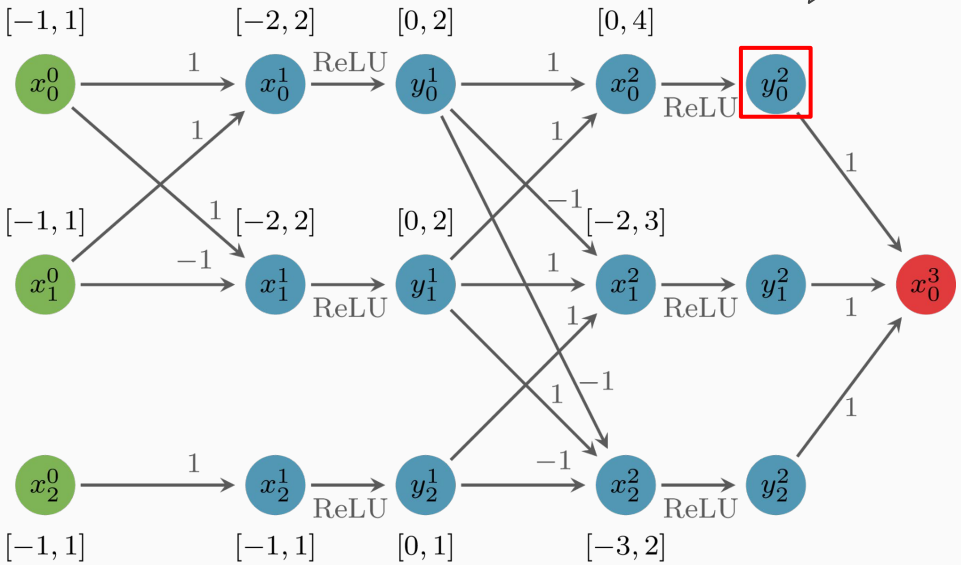
Interval arithmetic - example

$$\max(x_0^2) = \max(y_0^1 + y_1^1) \leq 2 + 2 = 4$$



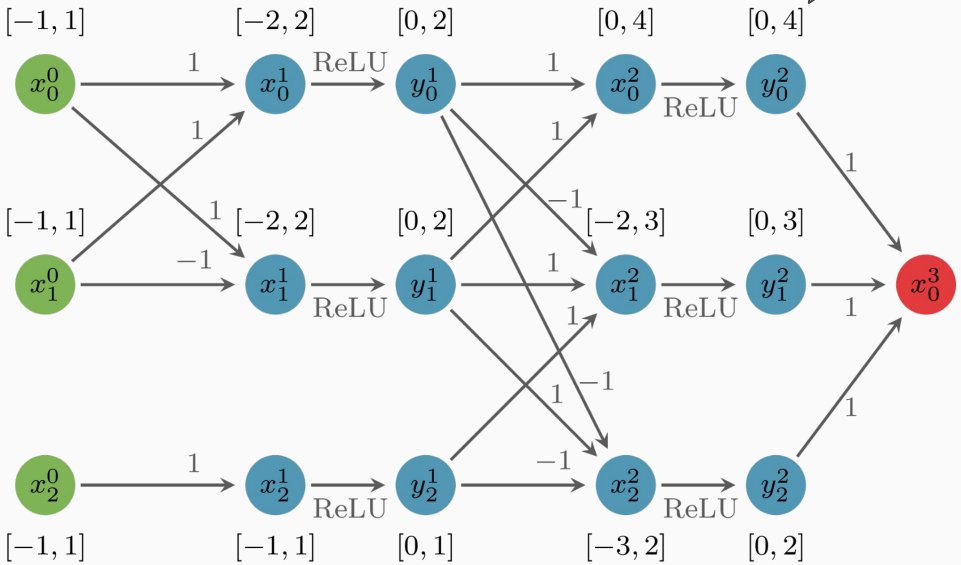
Interval arithmetic - example

$$y_0^2 = \sigma(x_0^2) = x_0^2$$

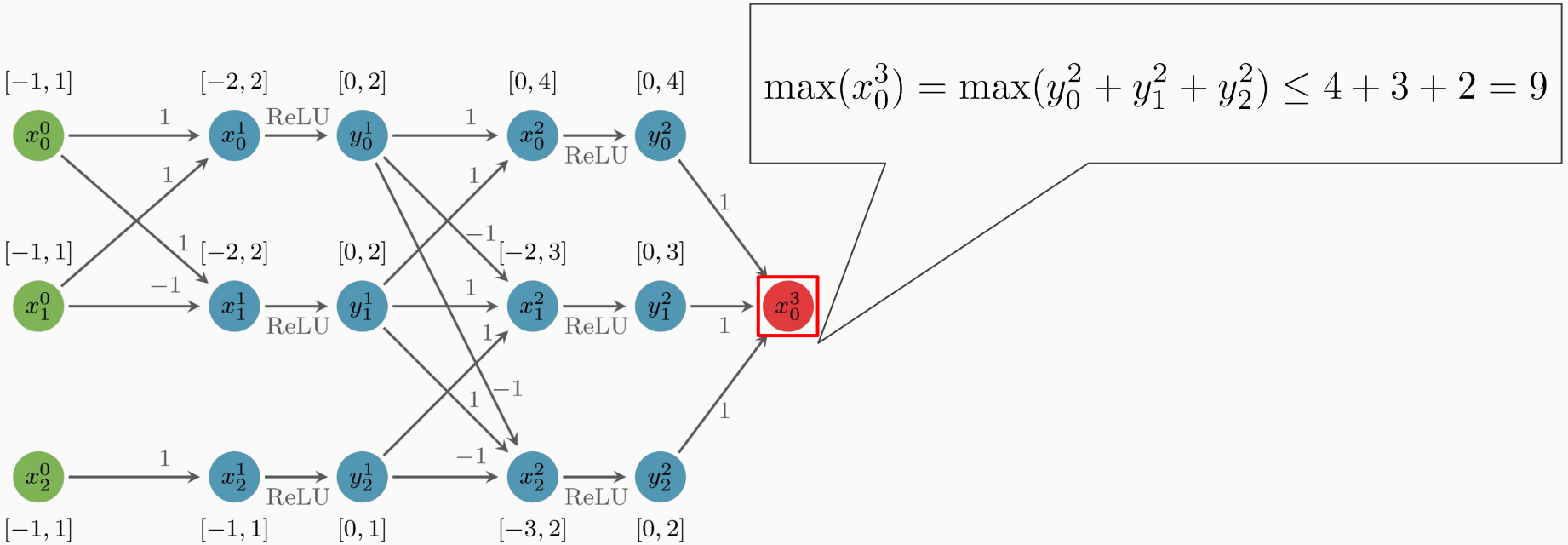


Interval arithmetic - example

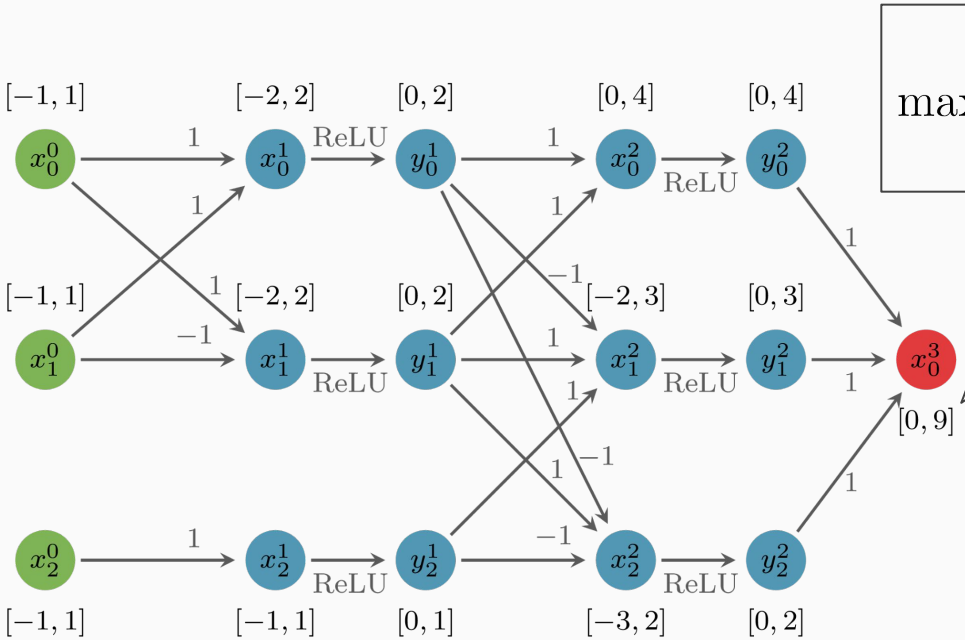
$$y_0^2 = \sigma(x_0^2) = x_0^2$$



Interval arithmetic - example



Interval arithmetic - example



$$\max(x_0^3) = \max(y_0^2 + y_1^2 + y_2^2) \leq 4 + 3 + 2 = 9$$

Interval arithmetic

$x_0^3 \leq 9$ Is not enough to prove the property $\mathcal{D}_o \equiv [0, 6\frac{1}{5}]$

Can we do better?

Back-substitution

Agenda

We are going to go see:

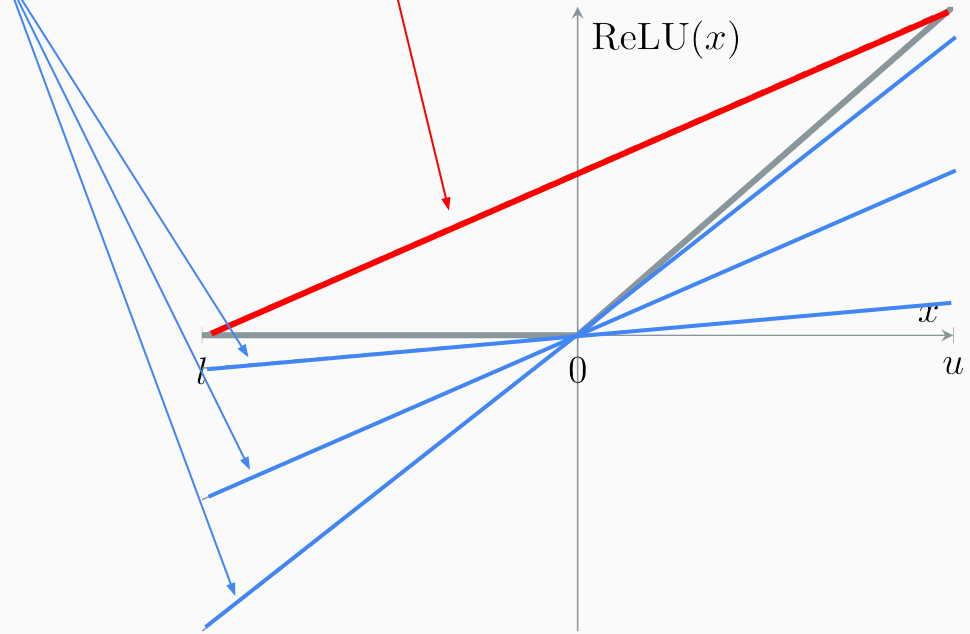
- ❏ Interval arithmetic
- ❏ Back-substitution (SoTA)
- ❏ Our optimization

Back-substitution

- ***Interval Arithmetic*** ignores dependencies between neurons
- **Back-substitution** uses a linear relaxation of the activation functions to compute better bounds while still being fast.

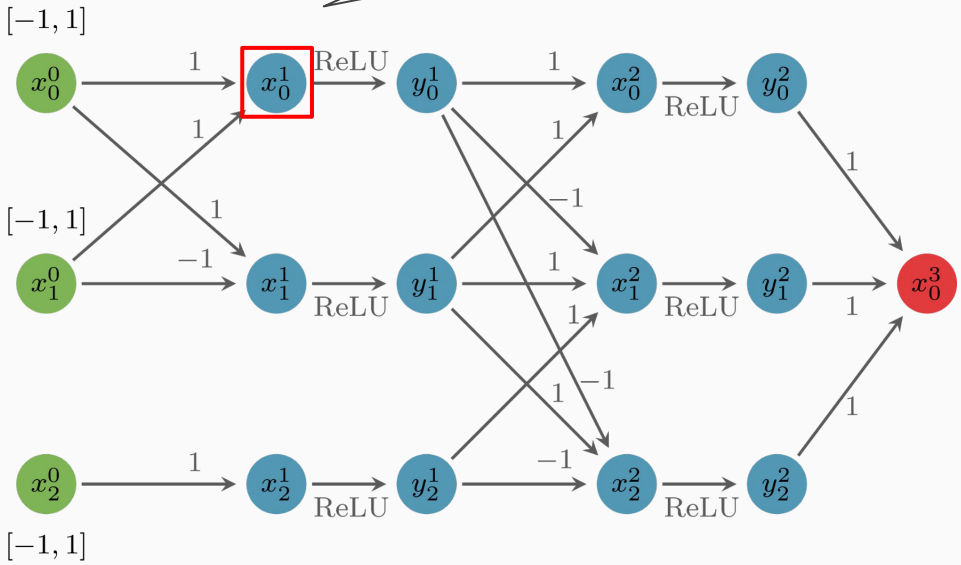
Linear relaxation

$$\begin{cases} \sigma(x) = 0 & \text{if } u \leq 0 \\ \sigma(x) = x & \text{if } l \geq 0 \\ \alpha x \leq \sigma(x) \leq \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \leq \alpha \leq 1 \end{cases}$$



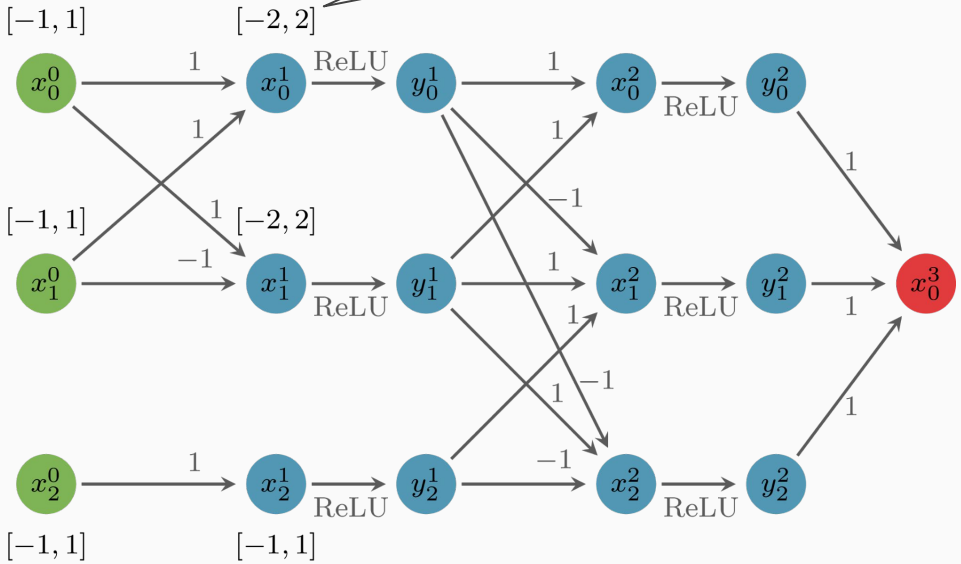
Back-substitution - example

$$\max(x_0^1) = \max(x_0^0 + x_1^0) \leq 1 + 1 = 2$$



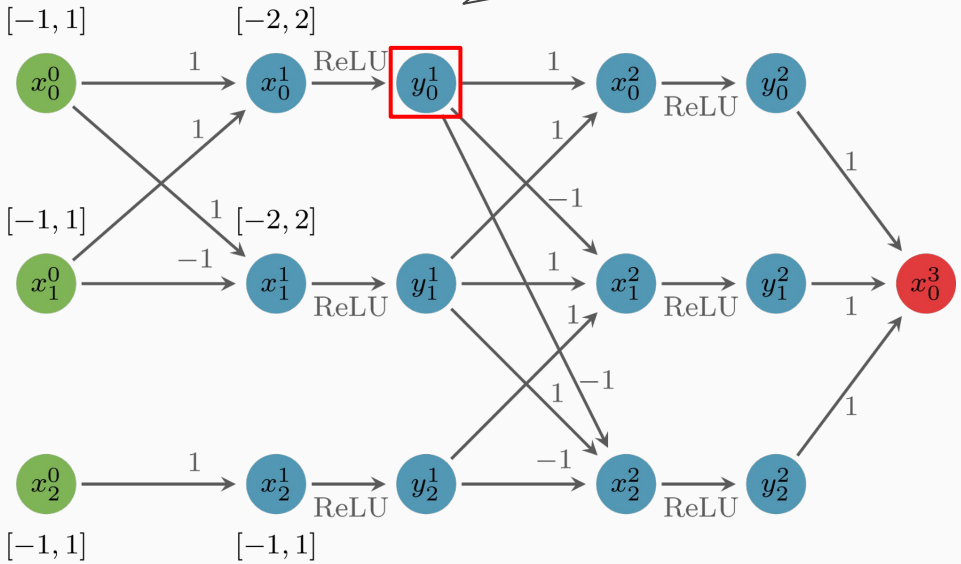
Back-substitution - example

$$\max(x_0^1) = \max(x_0^0 + x_1^0) \leq 1 + 1 = 2$$



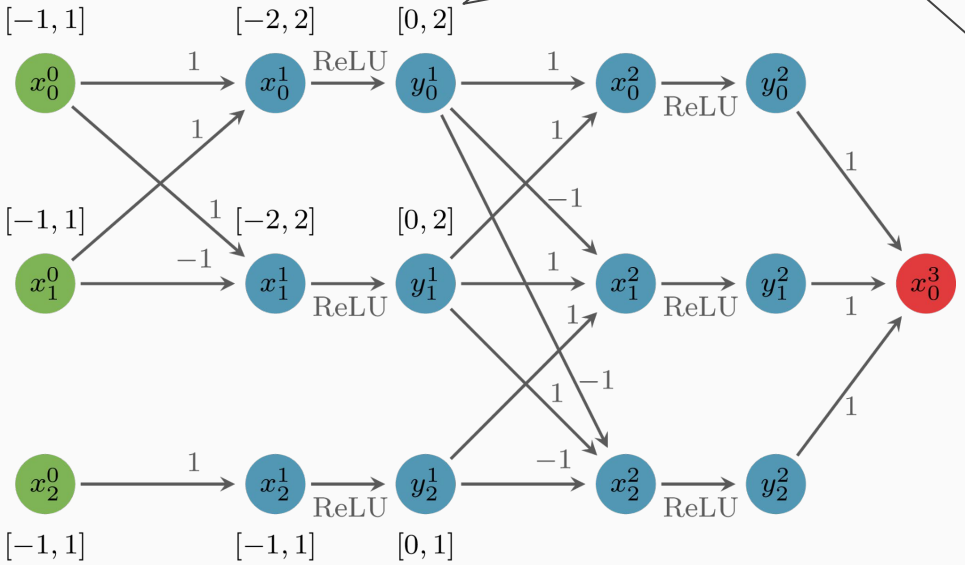
Back-substitution - example

$$\max(y_0^1) = \max(\sigma(x_0^1)) \leq \max(\tfrac{1}{2}\left(x_0^1 + 2\right)) = \max(\tfrac{1}{2}\left(x_0^0 + x_1^0 + 2\right))$$

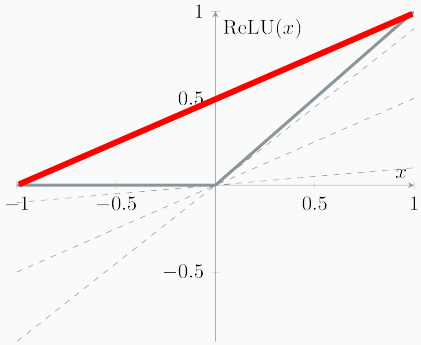


Back-substitution - example

$$\max(y_0^1) = \max(\sigma(x_0^1)) \leq \max\left(\tfrac{1}{2}\left(x_0^1 + 2\right)\right) = \max\left(\tfrac{1}{2}\left(x_0^0 + x_1^0 + 2\right)\right)$$

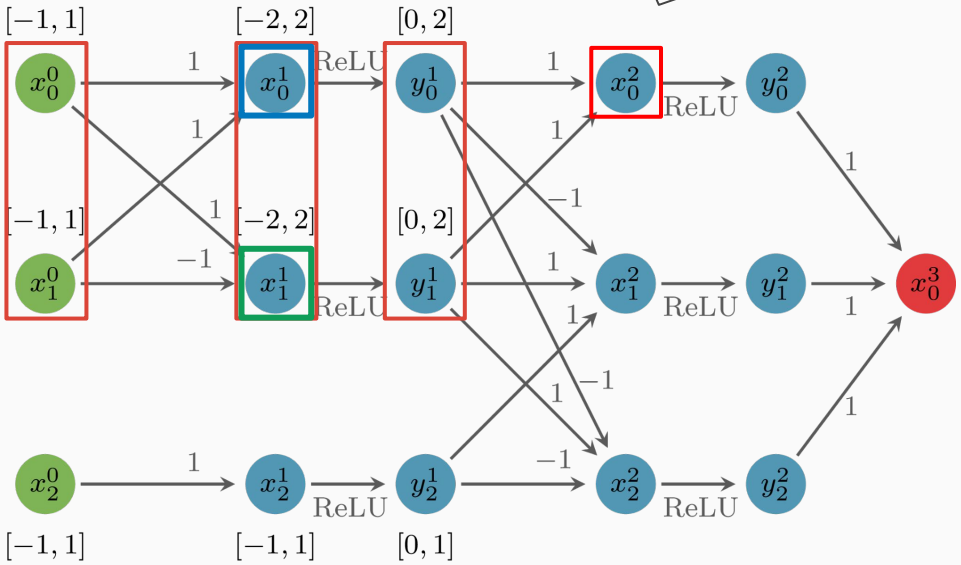


$$\begin{cases} 0 \leq \sigma(x) \leq 0 & \text{if } u \leq 0 \\ x \leq \sigma(x) \leq x & \text{if } l \geq 0 \\ \alpha x \leq \sigma(x) \leq \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \leq \alpha \leq 1 \end{cases}$$



Back-substitution - example

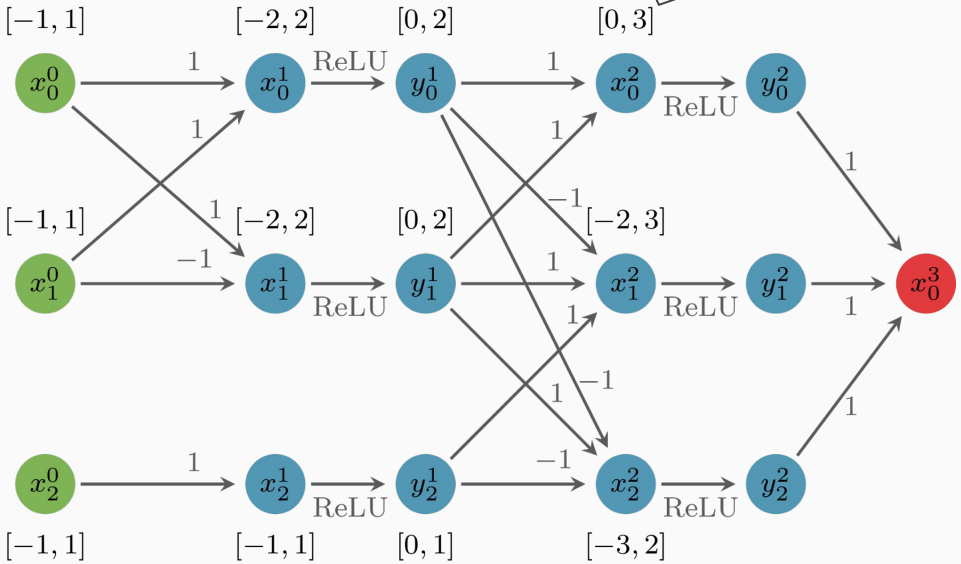
$$\max(x_0^2) = \max(y_0^1 + y_1^1) = \max(\sigma(x_0^1) + \sigma(x_1^1))$$
$$\leq \max(\frac{1}{2}(x_0^0 + x_1^0 + 2) + \frac{1}{2}(x_0^0 - x_1^0 + 2)) = \max(x_0^0 + 2) \leq 1 + 2 = 3$$



$$\begin{cases} \sigma(x) = 0 & \text{if } u \leq 0 \\ \sigma(x) = x & \text{if } l \geq 0 \\ \alpha x \leq \sigma(x) \leq \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \leq \alpha \leq 1 \end{cases}$$

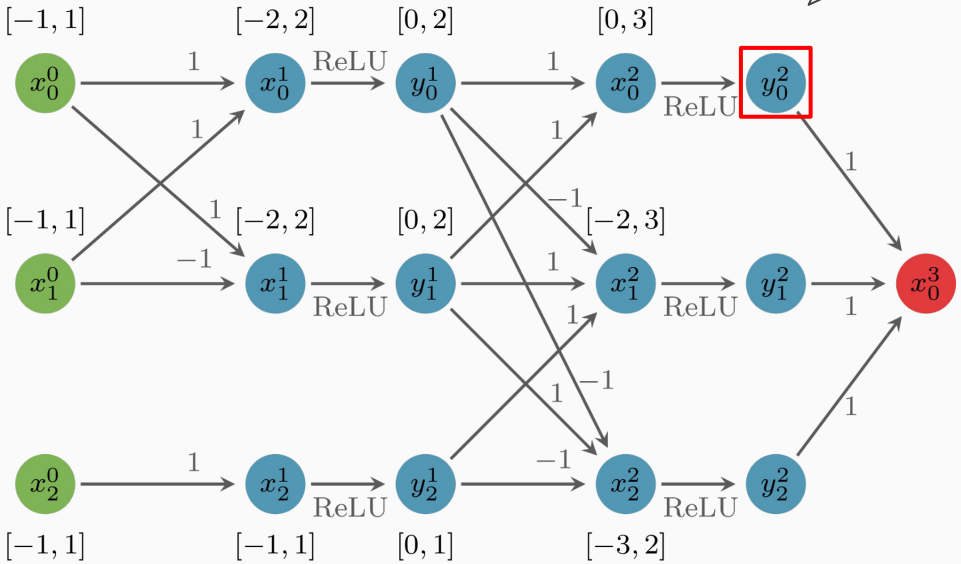
Back-substitution - example

$$\max(x_0^2) = \max(y_0^1 + y_1^1) = \max(\sigma(x_0^1) + \sigma(x_1^1))$$
$$\leq \max(\frac{1}{2}\left(x_0^0 + x_1^0 + 2\right) + \frac{1}{2}\left(x_0^0 - x_1^0 + 2\right)) = \max(x_0^0 + 2) \leq 1 + 2 = 3$$

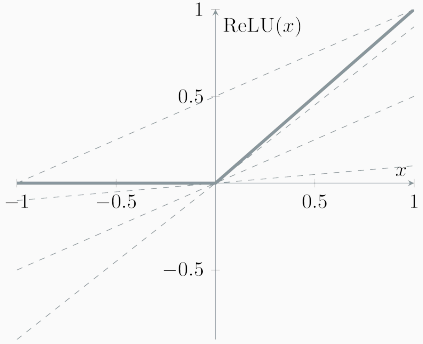


Back-substitution - example

$$\max(y_0^2) = \max(\sigma(x_0^2)) = \max(x_0^2)$$

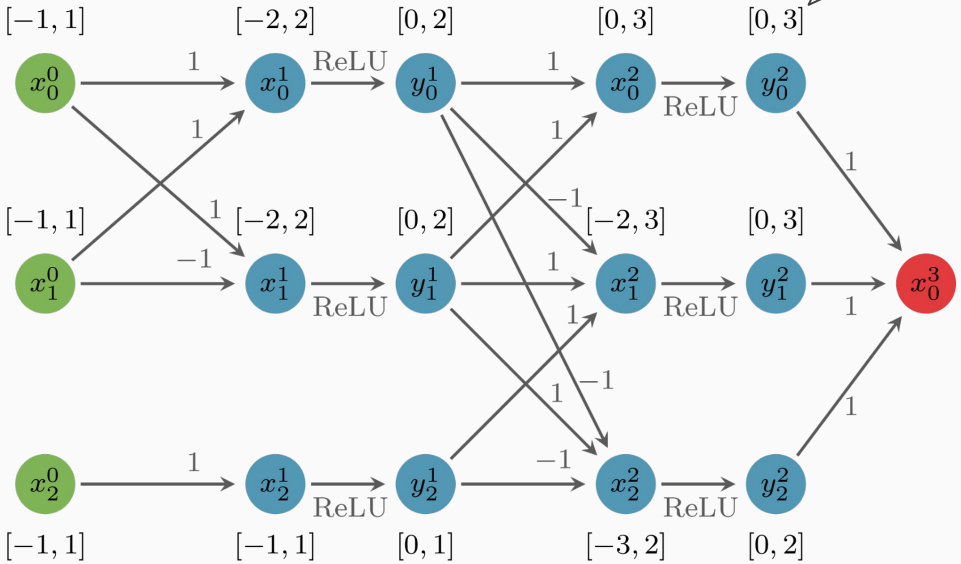


$$\begin{cases} \sigma(x) = 0 & \text{if } u \le 0 \\ \sigma(x) = x & \text{if } l \ge 0 \\ \alpha x \le \sigma(x) \le \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \le \alpha \le 1 \end{cases}$$



Back-substitution - example

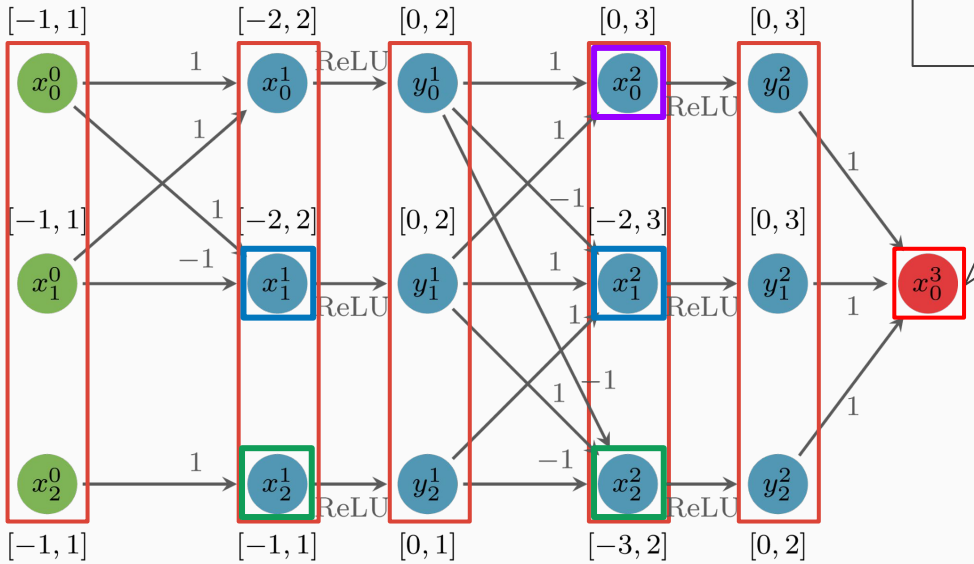
$$\max(y_0^2) = \max(\sigma(x_0^2)) = \max(x_0^2)$$



Back-substitution - example

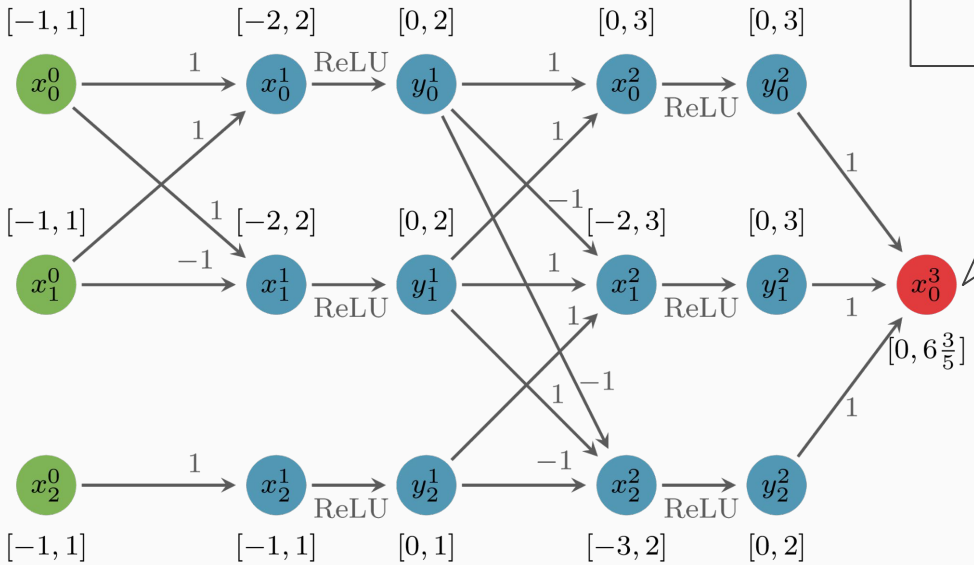
$$\begin{cases} \sigma(x) = 0 & \text{if } u \leq 0 \\ \sigma(x) = x & \text{if } l \geq 0 \\ \alpha x \leq \sigma(x) \leq \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \leq \alpha \leq 1 \end{cases}$$

$$\begin{aligned} \max(x_0^3) &= \max(y_0^2 + y_1^2 + y_2^2) = \max(\sigma(x_0^2) + \sigma(x_1^2) + \sigma(x_2^2)) \\ &\leq \max(x_0^2 + \frac{3}{5}(x_1^2 + 2) + \frac{2}{5}(x_2^2 + 3)) \\ &= \max(2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5}) = \max(2\sigma(x_1^1) + \frac{1}{5}\sigma(x_2^1) + \frac{12}{5}) \\ &\leq \max(2 \cdot \frac{1}{2}(x_0^0 - x_1^0 + 2) + \frac{1}{5} \cdot \frac{1}{2}(x_2^0 + 1) + \frac{12}{5}) \\ &= \max(x_0^0 - x_1^0 + \frac{1}{10}x_2^0 + 4\frac{1}{2}) \leq 1 + 1 + \frac{1}{10} + 4\frac{1}{2} = 6\frac{3}{5} \end{aligned}$$



Back-substitution - example

$$
\begin{aligned}
\max(x_0^3) &= \max(y_0^2 + y_1^2 + y_2^2) = \max(\sigma(x_0^2) + \sigma(x_1^2) + \sigma(x_2^2)) \\
&\leq \max(x_0^2 + \frac{3}{5}\left(x_1^2 + 2\right) + \frac{2}{5}\left(x_2^2 + 3\right)) \\
&= \max(2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5}) = \max(2\sigma(x_1^1) + \frac{1}{5}\sigma(x_2^1) + \frac{12}{5}) \\
&\leq \max(2 \cdot \frac{1}{2}\left(x_0^0 - x_1^0 + 2\right) + \frac{1}{5} \cdot \frac{1}{2}\left(x_2^0 + 1\right) + \frac{12}{5}) \\
&= \max(x_0^0 - x_1^0 + \frac{1}{10}x_2^0 + 4\frac{1}{2}) \leq 1 + 1 + \frac{1}{10} + 4\frac{1}{2} = 6\frac{3}{5}
\end{aligned}
$$



Back-substitution

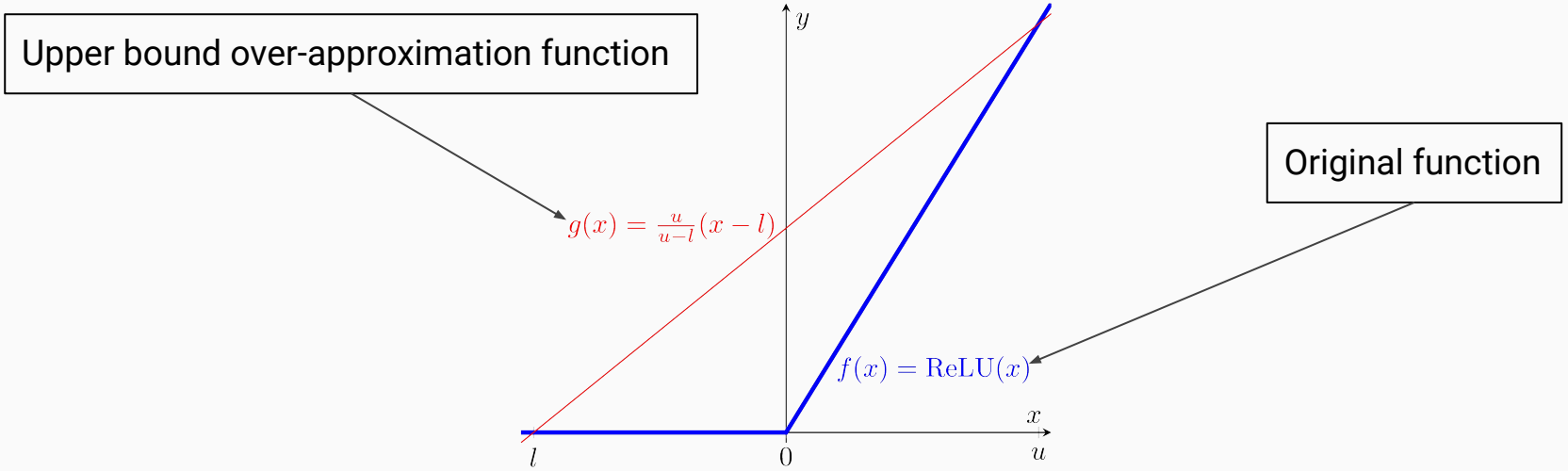
$$x_0^3 \leq 6\frac{3}{5}$$

Can we do better?

Over-approximation errors in back-substitution

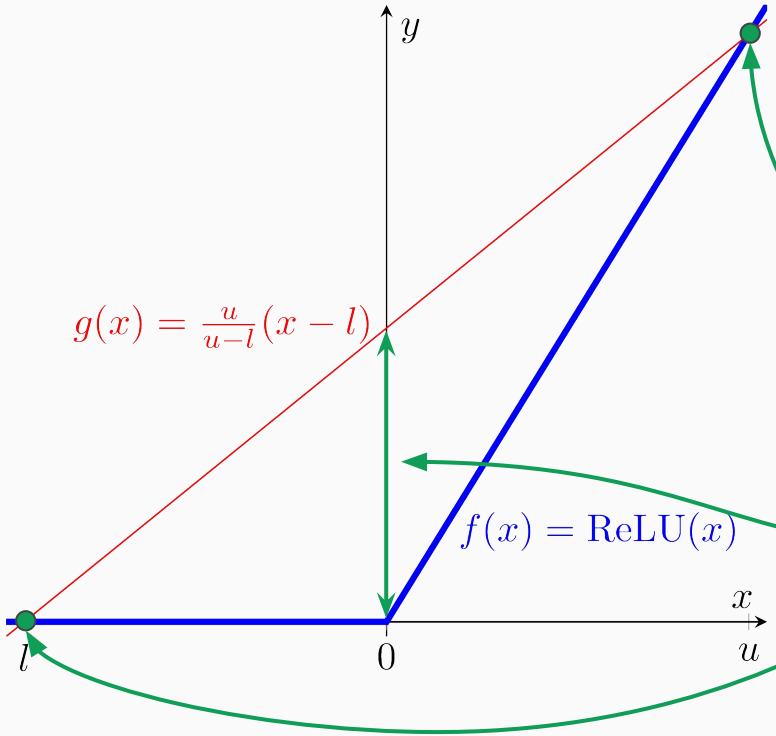
Over-approximation error

Definition 1 (Bound error) *Let $f : \mathbb{R}^n \to \mathbb{R}$, and let $g(\boldsymbol{x})$ be an upper bound for f over domain $\mathcal{D}$, such that we have: $\forall \boldsymbol{x} \in \mathcal{D} : g(\boldsymbol{x}) \geq f(\boldsymbol{x})$. We define the error of g with respect to f as the function: $E(\boldsymbol{x}) = g(\boldsymbol{x}) - f(\boldsymbol{x})$. The case for a lower bound is symmetrical.*



Over-approximation error

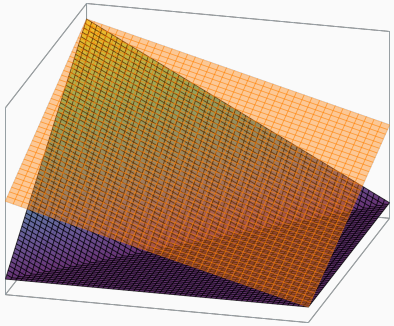
$$\forall x \in [l, u] : E(x) = \frac{u}{u-l}(x-l) - \sigma(x)$$



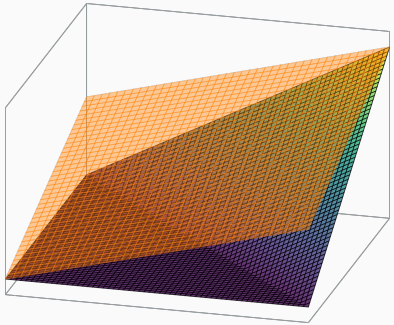
$$E(l) = E(u) = 0$$

$$E(0) = -\frac{ul}{u-l}$$

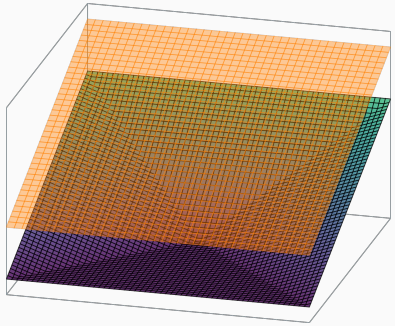
Bound detachment example



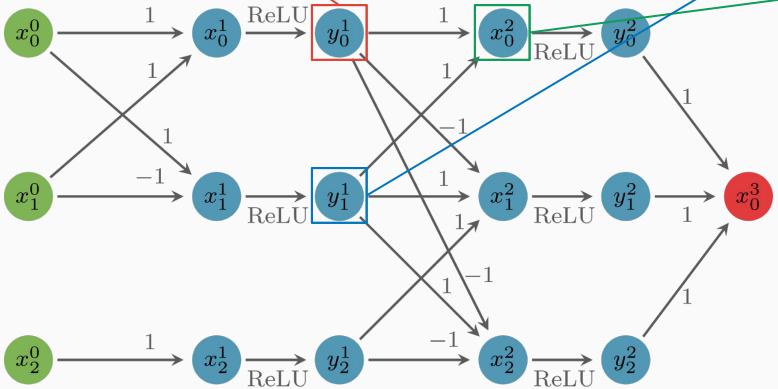
$\sigma(x_0^0 + x_1^0) \leq \frac{1}{2}\left(x_0^0 + x_1^0 + 2\right)$



$\sigma(x_0^0 - x_1^0) \leq \frac{1}{2}\left(x_0^0 - x_1^0 + 2\right)$

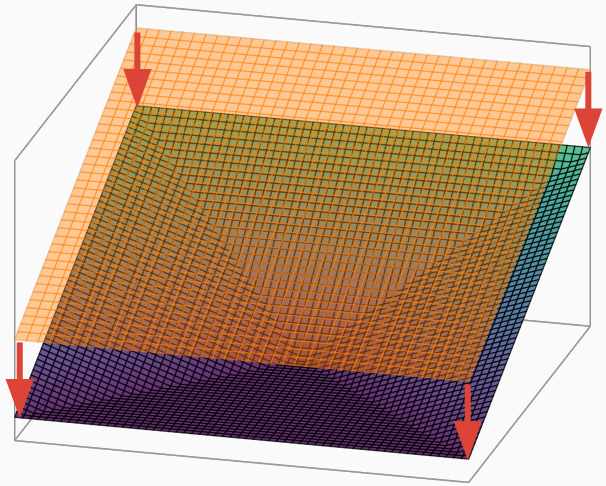


$\sigma(x_0^0 + x_1^0) + \sigma(x_0^0 - x_1^0) \leq x_0^0 + 2$

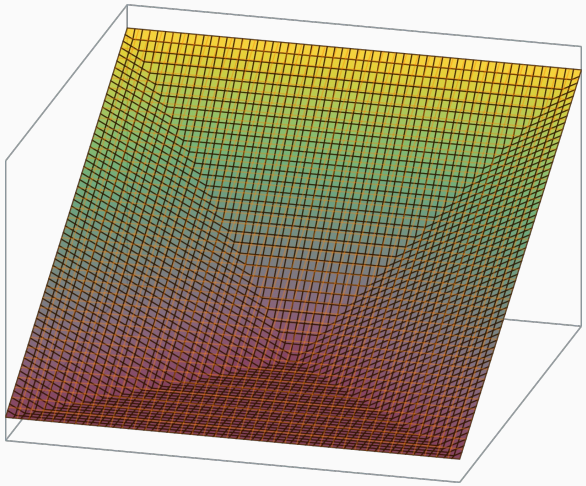


$$x_0^2 = \sigma(x_0^1) + \sigma(x_1^1) = \sigma(x_0^0 + x_1^0) + \sigma(x_0^0 - x_1^0)$$

Bound detachment fixing

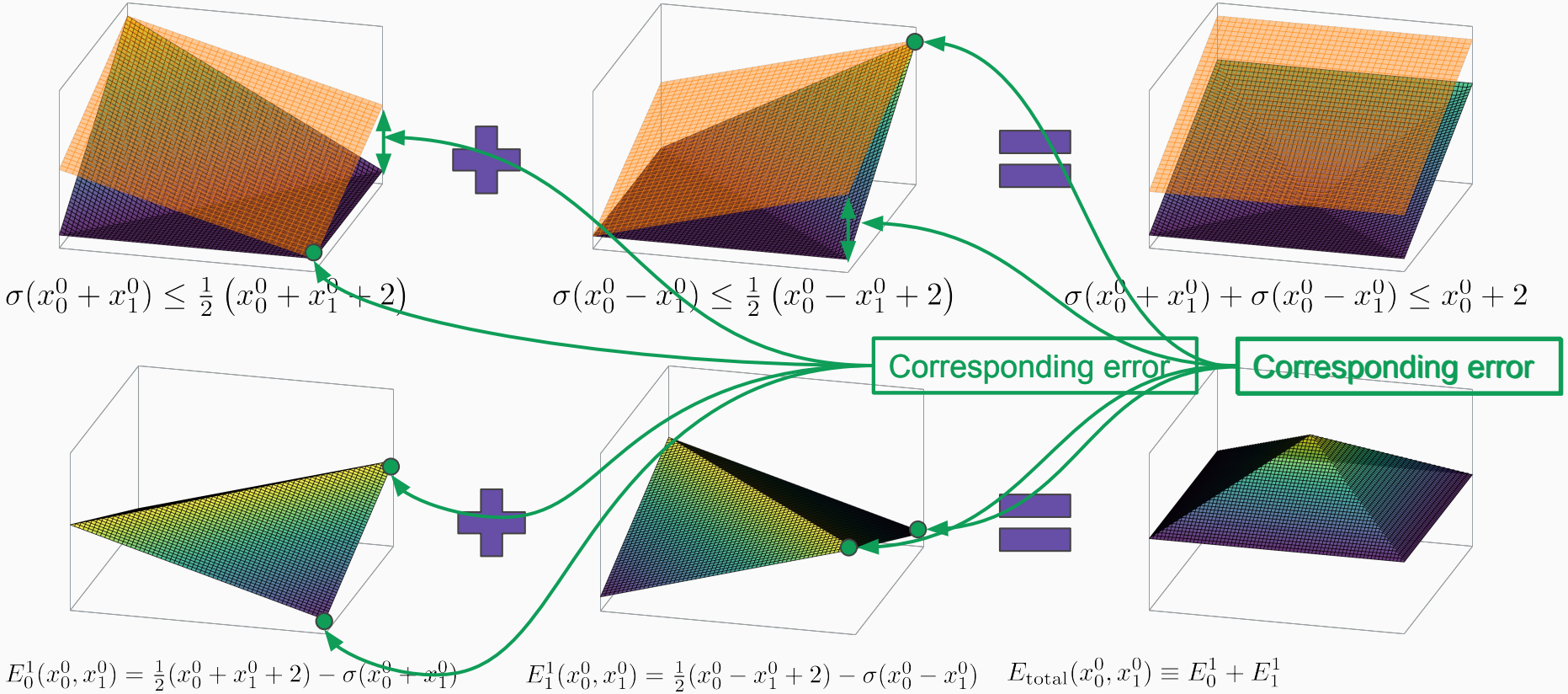


$$\sigma(x_0^0 + x_1^0) + \sigma(x_0^0 - x_1^0) \leq x_0^0 + 2$$

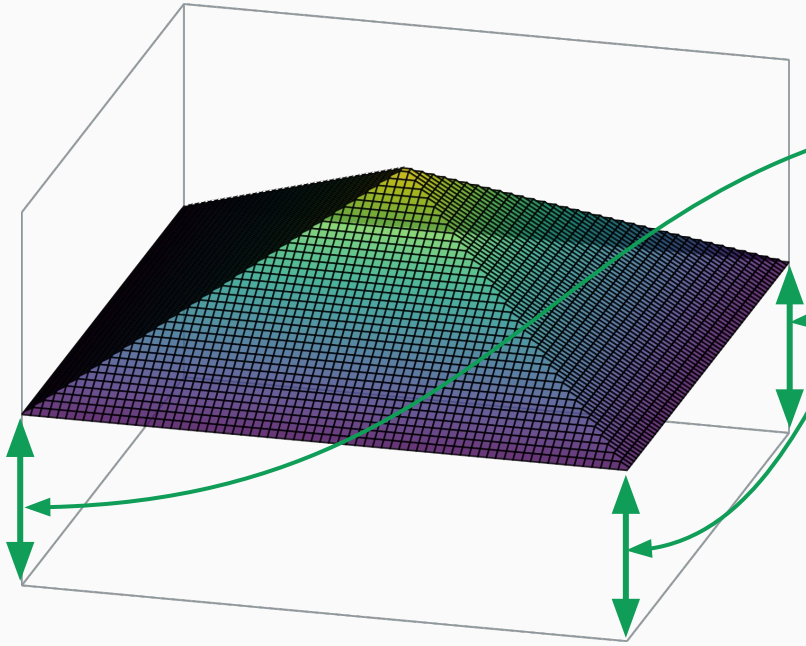


$$\sigma(x_0^0 + x_1^0) + \sigma(x_0^0 - x_1^0) \leq x_0^0 + 2 - \min(E_{\text{total}}) = x_0^0 + 1$$

Bound detachment example



Bound detachment cause



$$E_{\text{total}}(x_0^0, x_1^0) \geq 1$$

$$\min(E_{\text{total}}) = 1$$

$$\begin{aligned} E_{\text{total}}(x_0^0, x_1^0) &\equiv E_0^1 + E_1^1 \\ &= x_0^0 + 2 - \sigma(x_0^0 + x_1^0) - \sigma(x_0^0 - x_1^0) \end{aligned}$$

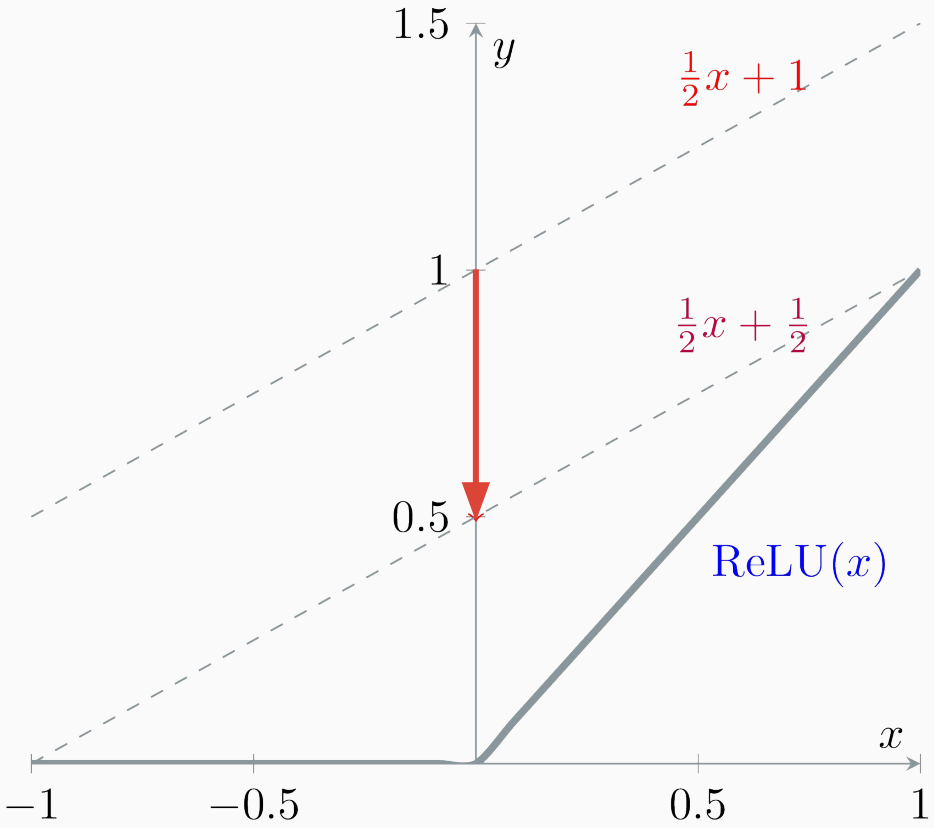
DeepMIP

Agenda

We are going to go see:

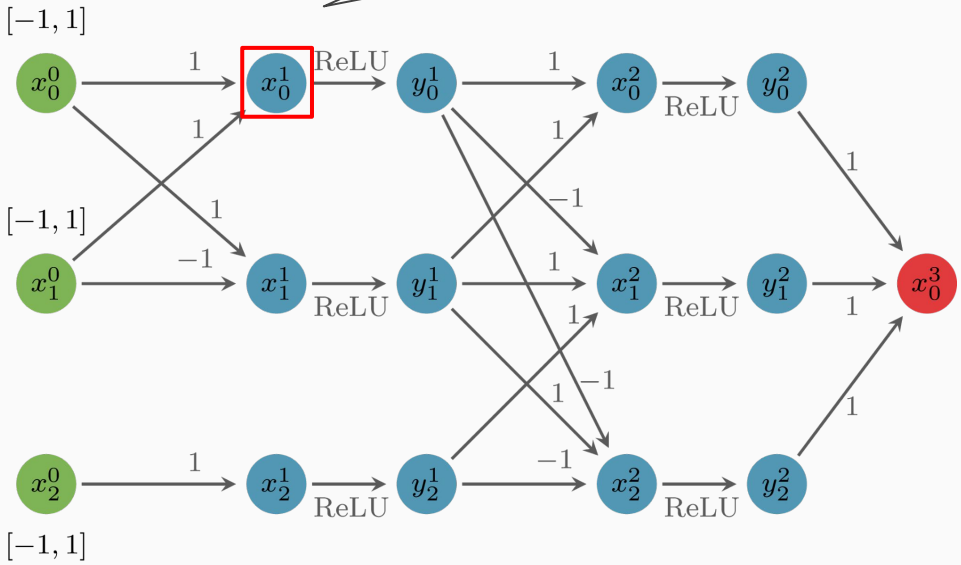
- ❏ Interval arithmetic
- ❏ Back-substitution (SoTA)
- ❏ Our optimization

DeepMIP



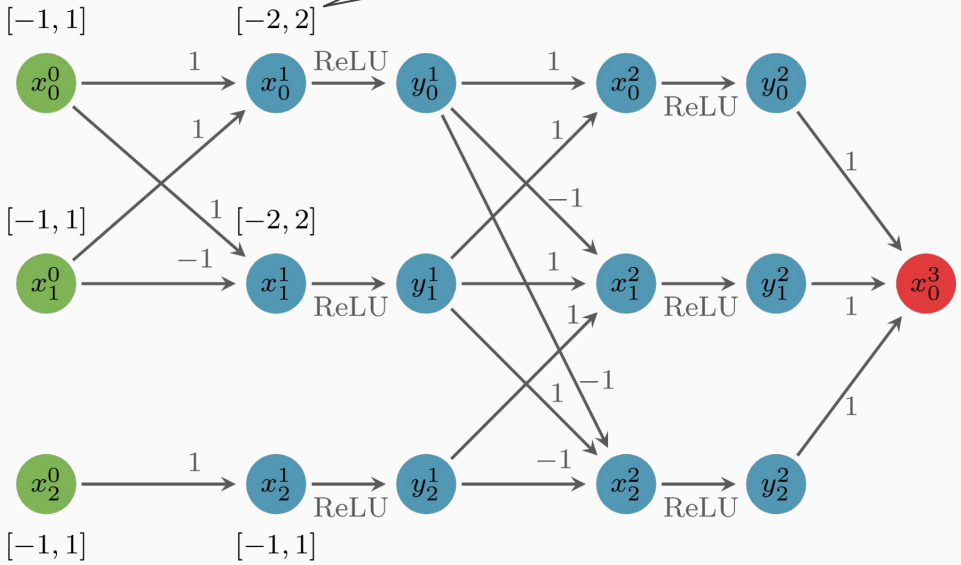
Back-substitution - example

$$\max(x_0^1) = \max(x_0^0 + x_1^0) \leq 1 + 1 = 2$$



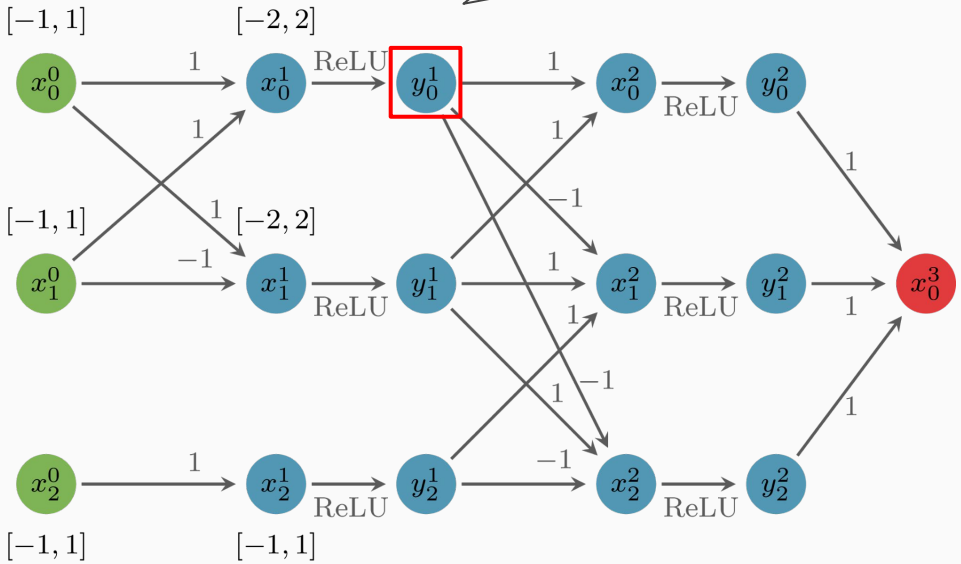
Back-substitution - example

$$\max(x_0^1) = \max(x_0^0 + x_1^0) \leq 1 + 1 = 2$$



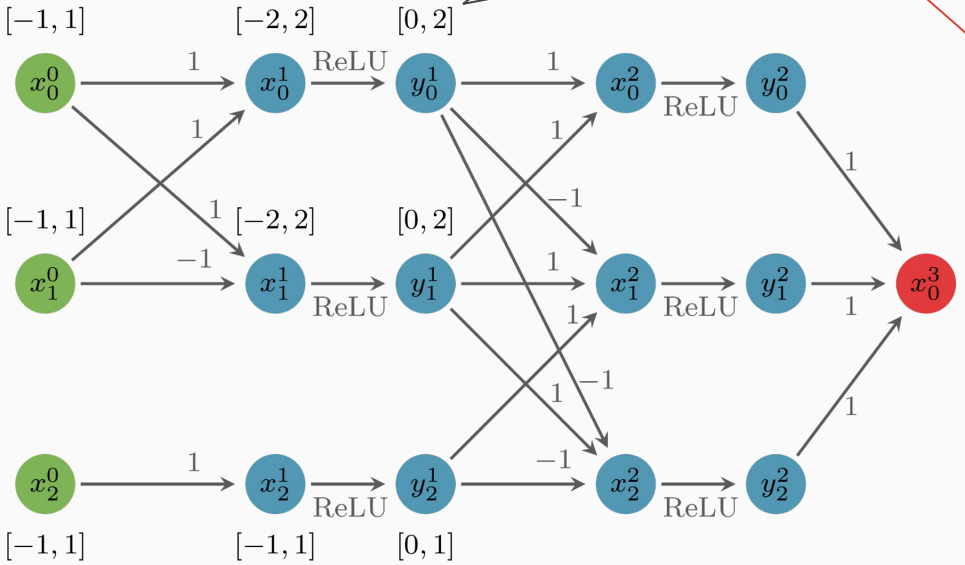
Back-substitution - example

$$\max(y_0^1) = \max(\sigma(x_0^1)) \leq \max(\tfrac{1}{2}\left(x_0^1 + 2\right)) = \max(\tfrac{1}{2}\left(x_0^0 + x_1^0 + 2\right))$$

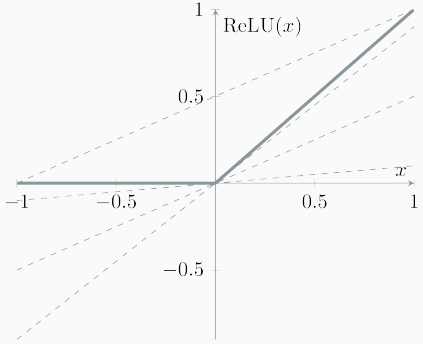


Back-substitution - example

$$\max(y_0^1) = \max(\sigma(x_0^1)) \leq \max\left(\frac{1}{2}\left(x_0^1 + 2\right)\right) = \max\left(\frac{1}{2}\left(x_0^0 + x_1^0 + 2\right)\right)$$



$$\begin{cases} \sigma(x) = 0 & \text{if } u \leq 0 \\ \sigma(x) = x & \text{if } l \geq 0 \\ \alpha x \leq \sigma(x) \leq \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \leq \alpha \leq 1 \end{cases}$$

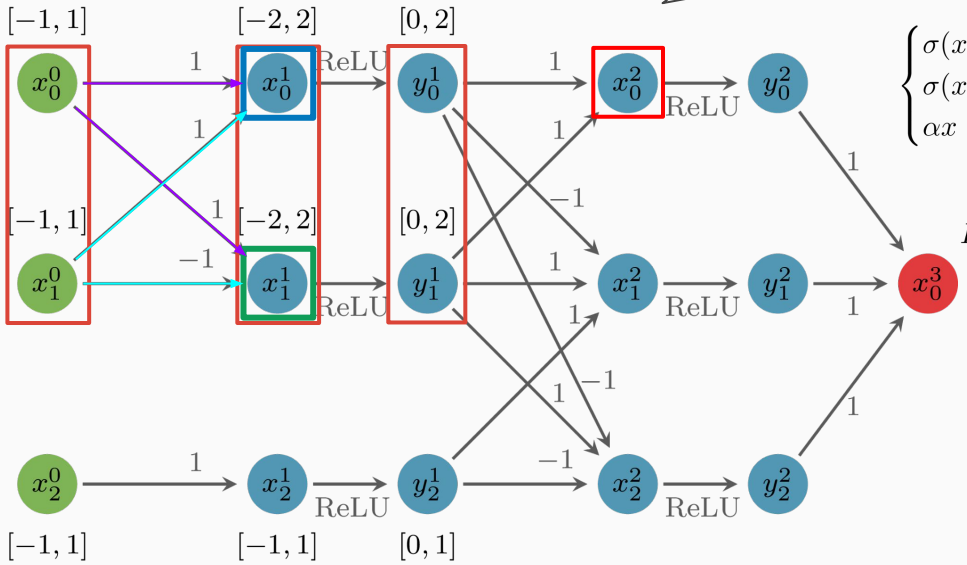


Back-substitution with error correction - example

$$
\begin{aligned}
\max(x_0^2) &= \max(y_0^1 + y_1^1) = \max(\sigma(x_0^1) + \sigma(x_1^1)) \\
&= \max(\frac{1}{2}(x_0^1 + 2) + \frac{1}{2}(x_1^1 + 2) - E^1) \\
&= \max(\frac{1}{2}(x_0^0 + x_1^0 + 2) + \frac{1}{2}(x_0^0 - x_1^0 + 2) - E^1) \\
&= \max(x_0^0 + 2 - E^1) \leq \max(x_0^0 + 2) - \min(E^1) \leq 2
\end{aligned}
$$

= 1
(MIP solver)

= 3



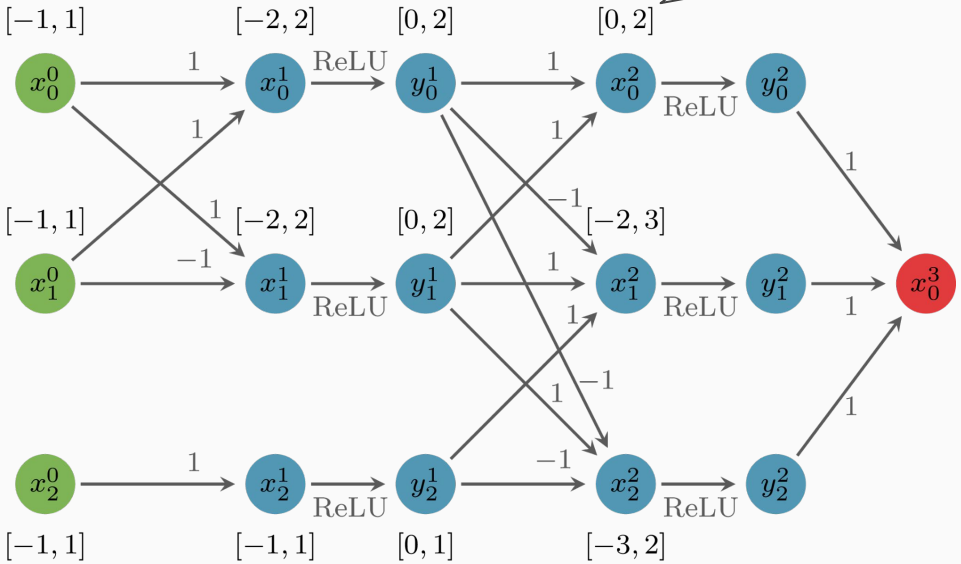
$$
\begin{cases}
\sigma(x) = 0 & \text{if } u \leq 0 \\
\sigma(x) = x & \text{if } l \geq 0 \\
\alpha x \leq \sigma(x) \leq \frac{u}{u-l}(x - l) & \text{otherwise, for any } 0 \leq \alpha \leq 1
\end{cases}
$$

$$
\begin{aligned}
E^1 &= \frac{1}{2}(x_0^1 + 2) + \frac{1}{2}(x_1^1 + 2) - (\sigma(x_0^1) + \sigma(x_1^1)) \\
&= \frac{1}{2}(x_0^0 + x_1^0 + 2) + \frac{1}{2}(x_0^0 - x_1^0 + 2) - (\sigma(x_0^0 + x_1^0) + \sigma(x_0^0 - x_1^0)) \\
&= x_0^0 + 2 - (\sigma(x_0^0 + x_1^0) + \sigma(x_0^0 - x_1^0))
\end{aligned}
$$

Better than the previous upper bound of 3!

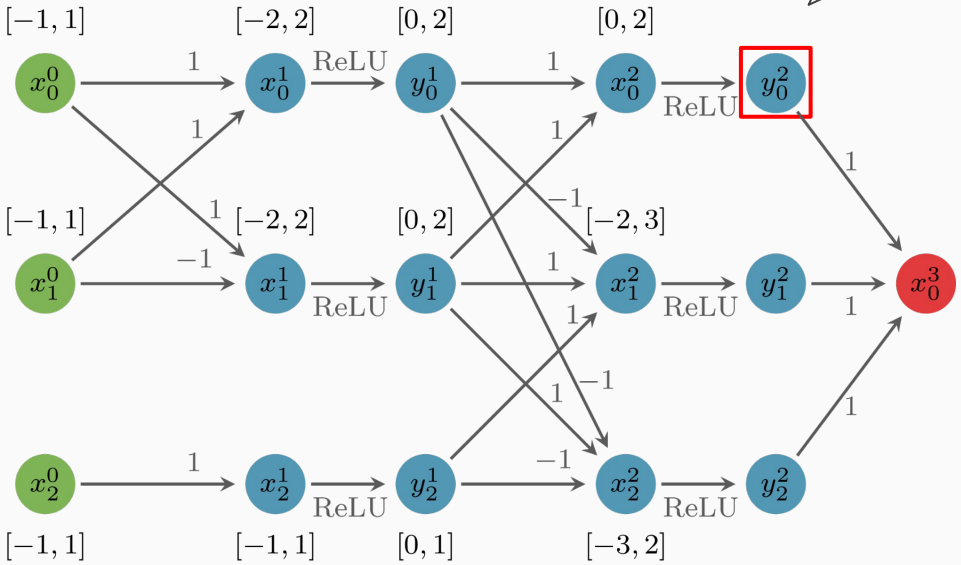
Back-substitution with error correction - example

$$
\begin{aligned}
\max(x_0^2) &= \max(y_0^1 + y_1^1) = \max(\sigma(x_0^1) + \sigma(x_1^1)) \\
&= \max(\frac{1}{2}\left(x_0^1 + 2\right) + \frac{1}{2}\left(x_1^1 + 2\right) - E^1) \\
&= \max(\frac{1}{2}\left(x_0^0 + x_1^0 + 2\right) + \frac{1}{2}\left(x_0^0 - x_1^0 + 2\right) - E^1) \\
&= \max(x_0^0 + 2 - E^1) \leq \max(x_0^0 + 2) - \min(E^1) \leq 2
\end{aligned}
$$

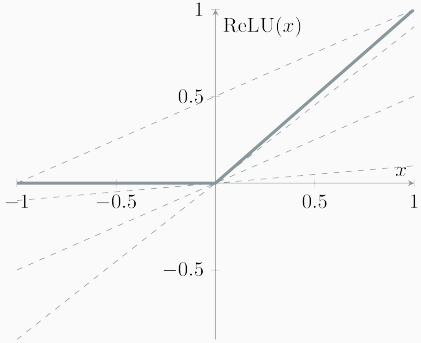


Back-substitution with error correction - example

$$\max(y_0^2) = \max(\sigma(x_0^2)) = \max(x_0^2)$$

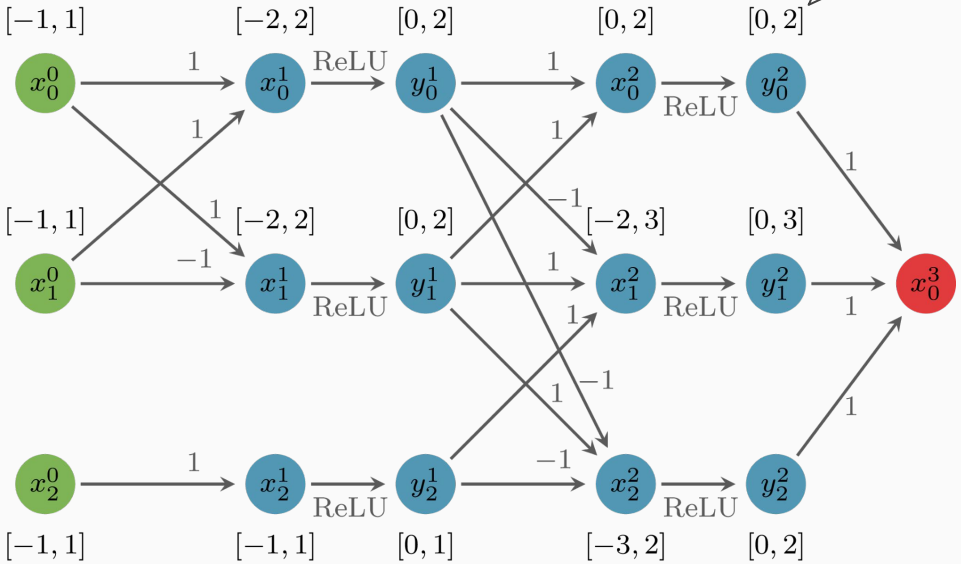


$$\begin{cases} \sigma(x) = 0 & \text{if } u \le 0 \\ \sigma(x) = x & \text{if } l \ge 0 \\ \alpha x \le \sigma(x) \le \frac{u}{u-l}(x-l) & \text{otherwise, for any } 0 \le \alpha \le 1 \end{cases}$$



Back-substitution with error correction - example

$$\max(y_0^2) = \max(\sigma(x_0^2)) = \max(x_0^2)$$



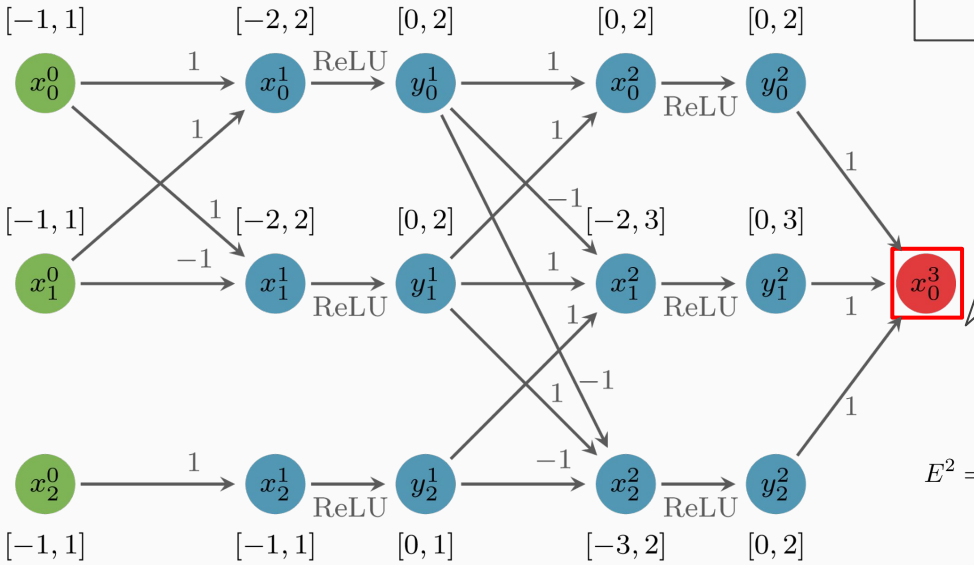
Back-substitution with error correction - example

Since we are already solving MIP problems and have reached the first layer, we can solve it directly instead of the last iteration of back-substitution.
In this particular case, It will yield the same results.

$$
\begin{aligned}
\max(x_0^3) &= \max(y_0^2 + y_1^2 + y_2^2) \\
&= \max(\sigma(x_0^2) + \sigma(x_1^2) + \sigma(x_2^2)) \\
&= \max\left(\sigma(y_0^1 + y_1^1) + \sigma(-y_0^1 + y_1^1 + y_2^1) + \sigma(-y_0^1 + y_1^1 - y_2^1)\right) \\
&= \max(2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5} - E^2) \\
&\leq \max(2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5}) - \min(E^2) \\
&= \max(2\sigma(x_0^0 - x_1^0) + \frac{1}{5}\sigma(x_2^0) + \frac{12}{5}) - \min(E^2) = 6
\end{aligned}
$$

=6⅘

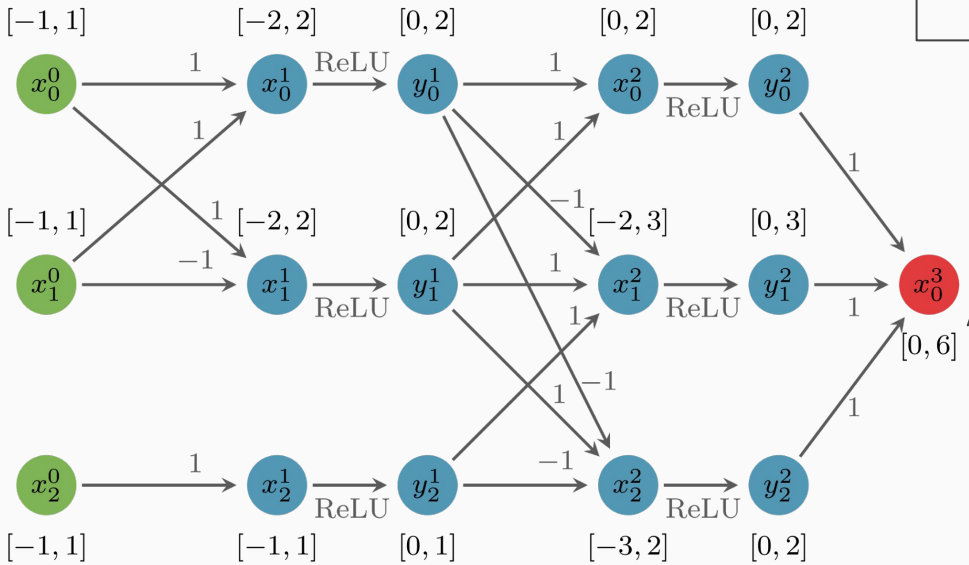
=⅘ (MIP solver)



$$E^2 = 2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5} - (\sigma(y_0^1 + y_1^1) + \sigma(-y_0^1 + y_1^1 + y_2^1) + \sigma(-y_0^1 + y_1^1 - y_2^1))$$

Back-substitution with error correction - example

$$
\begin{aligned}
\max(x_0^3) &= \max(y_0^2 + y_1^2 + y_2^2) \\
&= \max(\sigma(x_0^2) + \sigma(x_1^2) + \sigma(x_2^2)) \\
&= \max\left(\sigma(y_0^1 + y_1^1) + \sigma(-y_0^1 + y_1^1 + y_2^1) + \sigma(-y_0^1 + y_1^1 - y_2^1)\right) \\
&= \max(2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5} - E^2) \\
&\leq \max(2y_1^1 + \frac{1}{5}y_2^1 + \frac{12}{5}) - \min(E^2) \\
&= \max(2\sigma(x_0^0 - x_1^0) + \frac{1}{5}\sigma(x_2^0) + \frac{12}{5}) - \min(E^2) = 6
\end{aligned}
$$



Back-substitution with error correction

$x_0^3 \leq 6$ Is the optimal bound! And enough to prove $\mathcal{D}_o \equiv [0, 6\frac{1}{5}]$

MiniMIP

While DeepMIP produces very strong bounds, for each neuron it must solve multiple MIP instances during back-substitution — many of them for bounds that may already be optimal. This can result in a large overhead, and makes it worthwhile to explore heuristics for only solving some of these instances.

We propose a particular, aggressive heuristic that we call MiniMIP.
Instead of minimizing all error terms during back-substitution, MiniMIP only solves the final query in this series — that is, the query in which the bounds of the current layer are expressed as sums of ReLUs of input neurons.

This approach significantly reduces overhead: exactly one MIP instance is solved in each iteration, regardless of the depth of the layer currently being processed. As we later see in our evaluation, even this is already enough to achieve state-of-the-art performance and very tight bounds; and the resulting queries can be solved very efficiently.

Results

The neural networks used in our evaluation

Dataset	Model	Type	Neurons	Hidden Layers	Activation
MNIST	6 × 100	FC	510	5	ReLU
	9 × 100		810	8	
	6 × 200		1010	5	
	9 × 200		1610	8	

Comparing DeepMIP to α-CROWN and PRIMA

Model	α-CROWN		PRIMA		DeepMIP (MiniMIP)	
	Solved	Time (seconds)	Solved	Time (seconds)	Solved	Time (seconds)
6 × 100	207	38	504	123	581	302
9 × 100	223	88	427	252	463	452
6 × 200	349	93	652	222	709	801
9 × 200	308	257	600	462	625	1121
Total	1087	476	2183	1059	2378	2676

Results

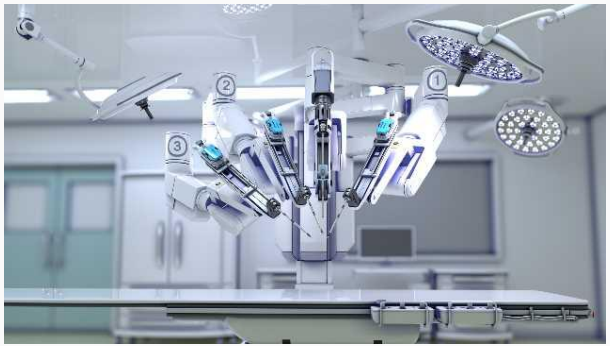
Our method solve more instances than the state-of-the-art on all benchmarks by generalizing back-substitution with error terms, allowing for tighter bounds to be computed.

Thank you!

questions?

A little background

- Neural networks are a method to design algorithms from data.
- They show state-of-the-art performance on many tasks.
- Quickly being adopted for use in a growing number of applications.
- Including safety critical applications.

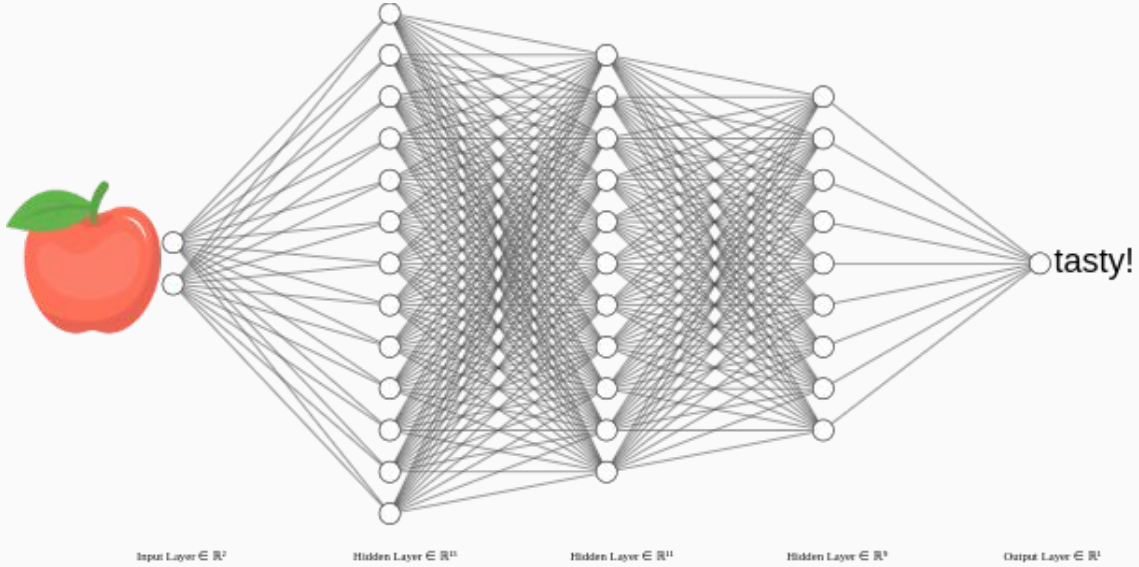


See the difference?

- Neural networks are inherently different from conventional algorithms
- They are challenging to reason about:

```
def is_apple_tasty(apple):
  if apple.color == "red":
    if apple.weight < 150:
      return True
    else:
      return False
  if apple.color == "green":
    if apple.weight > 100:
      return True
    else:
      return false
```

VS



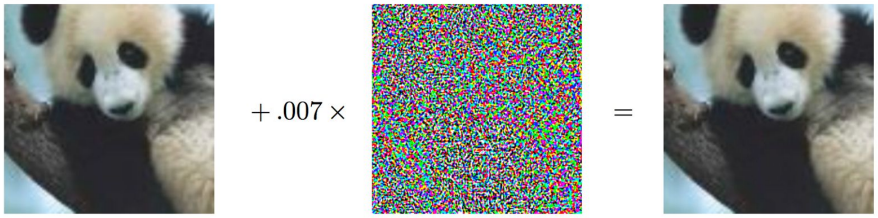
Why do we need to reason about neural networks

Neural networks can be tested, in fact - neural networks that are deployed in safety critical scenarios are **heavily** tested!

Question: should it even bother us that we can't reason about them? They are empirically safe!

Adversarial robustness

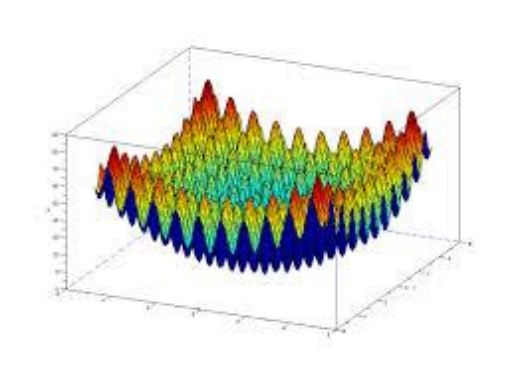
Answer: Yes! Neural networks could present behaviour that is not congruent with human expectations and common sense. Even safety critical and highly tested neural networks are not immune to breaking in surprising ways given specific inputs.

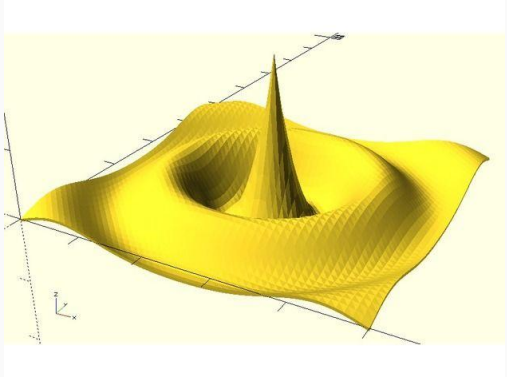


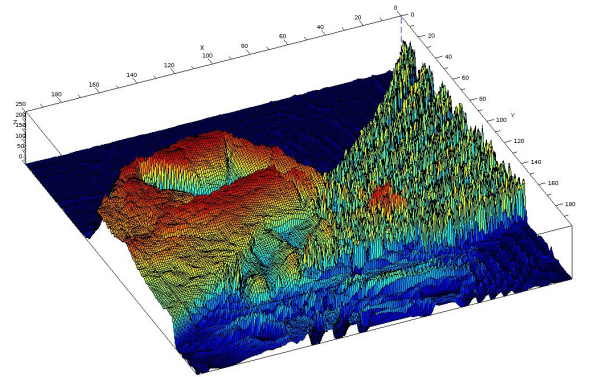
Adversarial robustness

Remember, trained neural networks are just functions, they takes an input and map it to some output.

They have no common sense and even when they behave well on test samples, there is no guarantee that they will not “flip out” on new, unseen input.







Verification of neural networks

Given a neural network for autonomous driving, can we guarantee that a funny rock at the side of an empty highway won't cause it to make a hard right turn while driving at 120 km/h?

Or in other words:

Given a neural network **NN: $x \longrightarrow y$**,
an input region **P(x)** and an output region **Q(y)**,
does there exists an input **$x_0 \in P(x)$** such that **$NN(x_0) \in Q(y)$?**

Existing solutions

In the general case, this kind of verification is NP-Complete

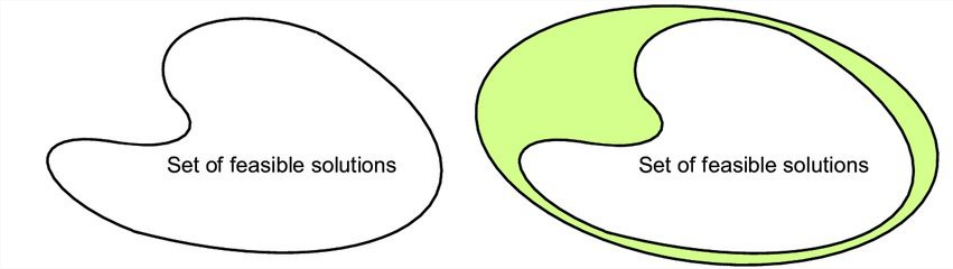
However there are methods capable of verifying many useful properties on many real-world neural networks without exhibiting worst case behaviour.

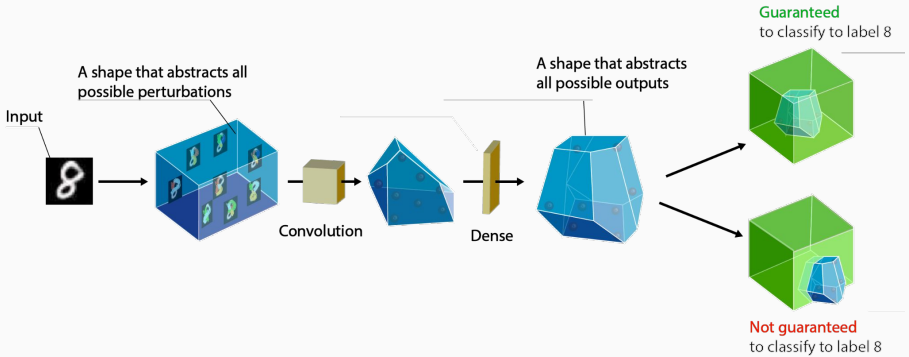
These methods can roughly be split into two types:

- Exact - might take exponential time.
- Relaxed - efficient but incomplete.

Existing solutions

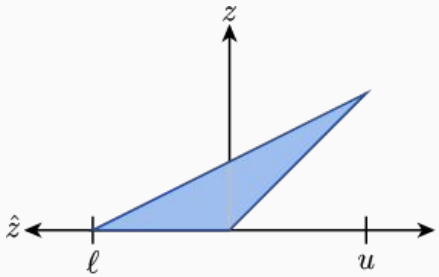
Among the relaxed methods: *convex-relaxation* methods are the most dominant, methods in this category are incomplete and attempt to relax the exact problem into a convex problem (for example, one that can be solved by an LP-solver).





The convex barrier

For scalability reasons, almost all approximations of non-linear activations operate separately on each neuron, For a single neuron convex approximation, the triangle relaxation:



Convex ReLU set

Is considered optimal since it is the convex hull of the ReLU function.

However in:

H. Salman, G. Yang, H. Zhang, C.-J. Hsieh, and P. Zhang, "A convex relaxation barrier to tight robustness verification of neural networks," in Advances in Neural Information Processing Systems, 2019, pp. 9835–9846.

It was shown that the even the optimal (but expensive to calculate) neuron relaxation for the ReLU activation function still achieves poor results compared to the actual robust-error of the neural network.