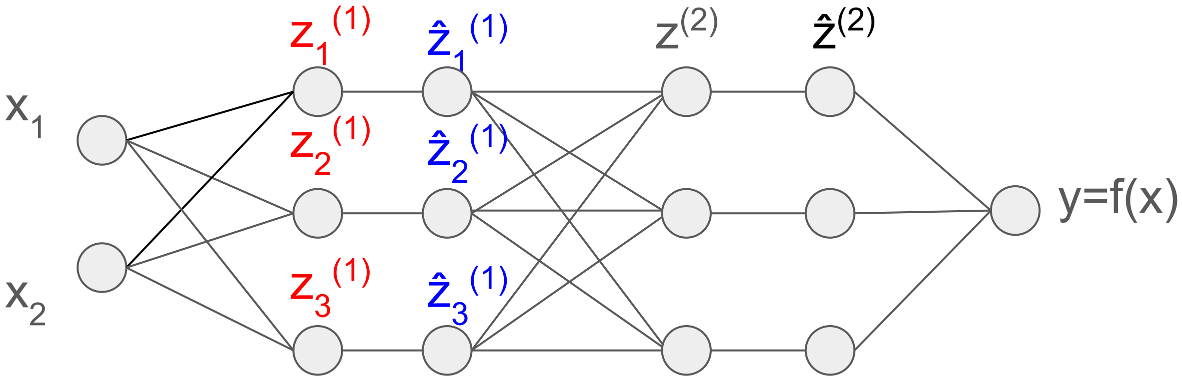
ECE598HZ: Advanced Topics in Machine Learning and Formal Methods

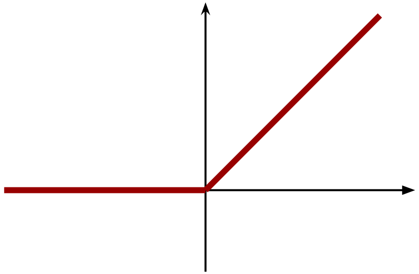
Lecture 10: Neural Network Verification with Bound Propagation Algorithms (Part I)

Prof. Huan Zhang

huan@huan-zhang.com

Review: Neural Networks (NNs)

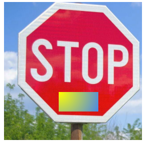


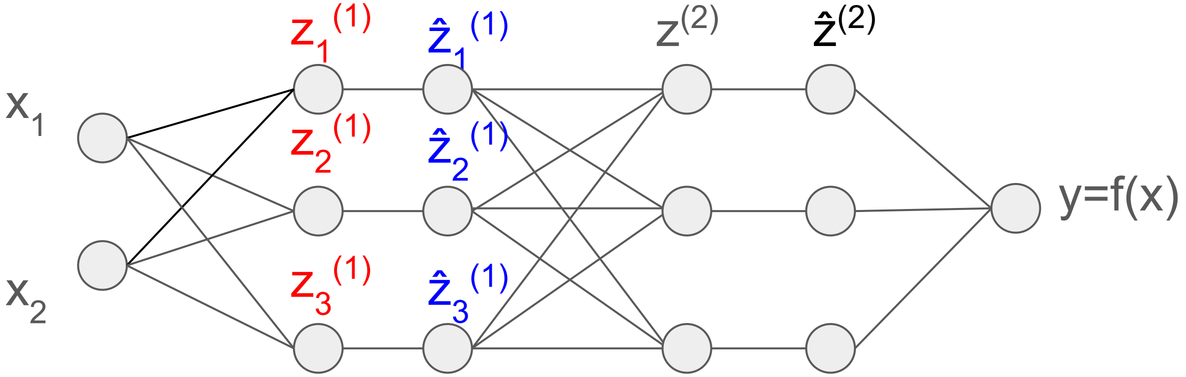


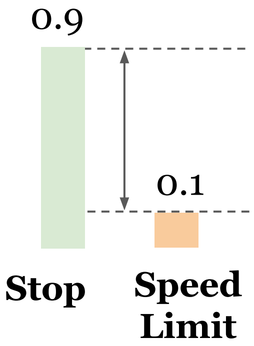
Linear layers: $z^{(1)} = W^{(1)} x$ $\quad z^{(2)} = W^{(2)} \hat{z}^{(1)}$ $\quad y = w^{(3)T} \hat{z}^{(2)}$

Nonlinear layers: $\hat{z}_j^{(i)} = \sigma(z_j^{(i)})$ (assume σ is ReLU for now)

Review: NN verification as an **optimization** problem







Does there $\exists$ x, s.t. $x \in S \wedge y \leq 0 \wedge y = f(x)$

$$y^* = \min_{x \in \mathcal{S}} f(x)$$

Input domain under consideration

Negation of the desired property

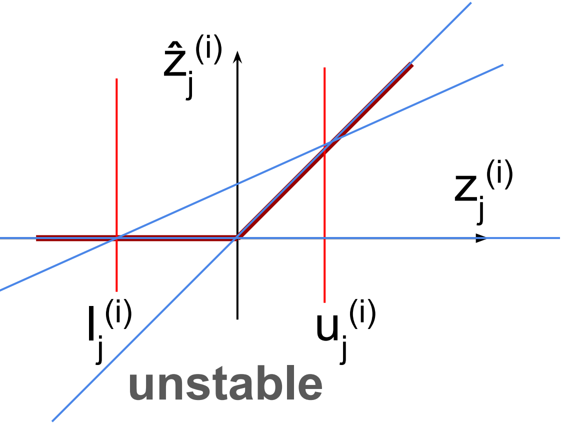
MILP and LP

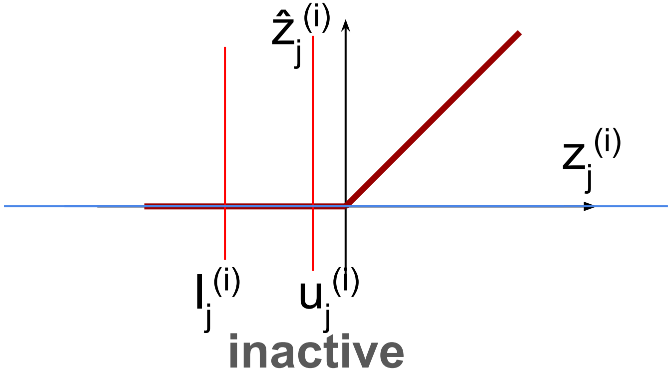
Review: stable vs. unstable neurons

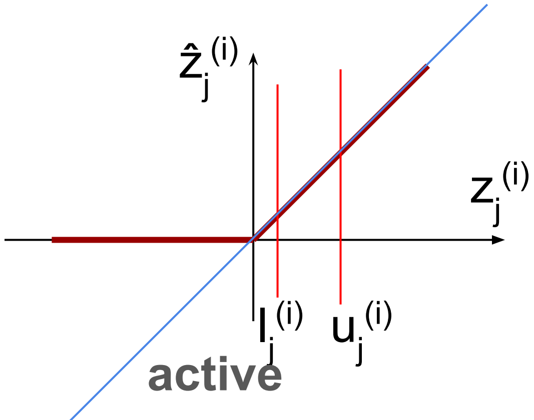
$$\hat{z}_j^{(i)} \leq z_j^{(i)} - l_j^{(i)}(1 - p_j^{(i)})$$
$$\hat{z}_j^{(i)} \leq u_j^{(i)} p_j^{(i)}$$
$$\hat{z}_j^{(i)} \geq z_j^{(i)}$$
$$\hat{z}_j^{(i)} \geq 0$$
$$0 \leq p_j^{(i)} \leq 1$$

$$\hat{z}_j^{(i)} = 0$$

$$\hat{z}_j^{(i)} = z_j^{(i)}$$







Review: triangle relaxation for unstable ReLU neurons

Each ReLU is represented by

$$\hat{z}_j^{(i)} \leq z_j^{(i)} - l_j^{(i)}(1 - p_j^{(i)})$$

$$\hat{z}_j^{(i)} \leq u_j^{(i)} p_j^{(i)}$$

$$\hat{z}_j^{(i)} \geq z_j^{(i)}$$

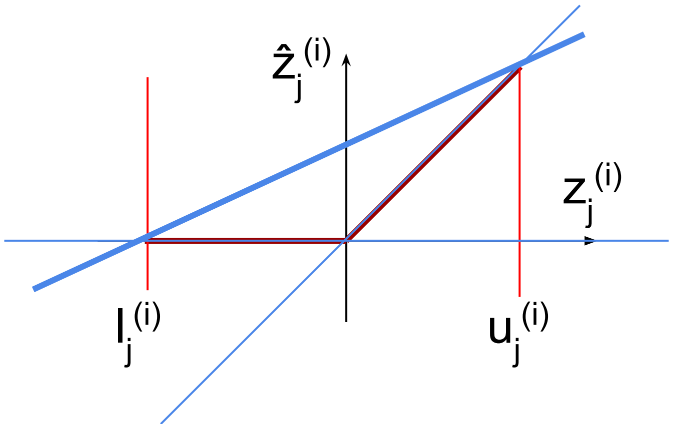
$$\hat{z}_j^{(i)} \geq 0$$

$$\hat{z}_j^{(i)} \leq \frac{u_j^{(i)}}{u_j^{(i)} - l_j^{(i)}} z_j^{(i)} - \frac{u_j^{(i)} l_j^{(i)}}{u_j^{(i)} - l_j^{(i)}}$$

$$\hat{z}_j^{(i)} \geq z_j^{(i)}$$

$$\hat{z}_j^{(i)} \geq 0$$

“Triangle” relaxation



Today: more efficient algorithms for NN verification

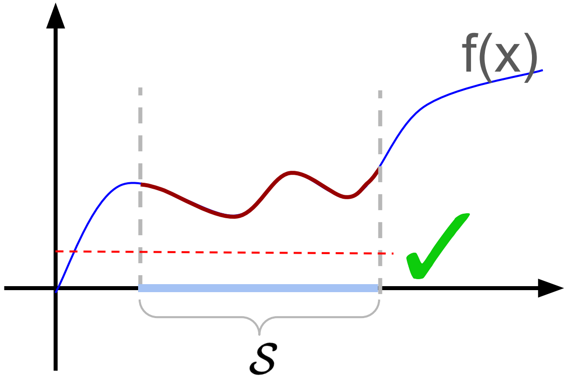
Solving neural network verification using SMT solvers

Solving neural network verification using optimization (MIP/LP)

Solving neural network verification using **bound propagation (this lecture!)**

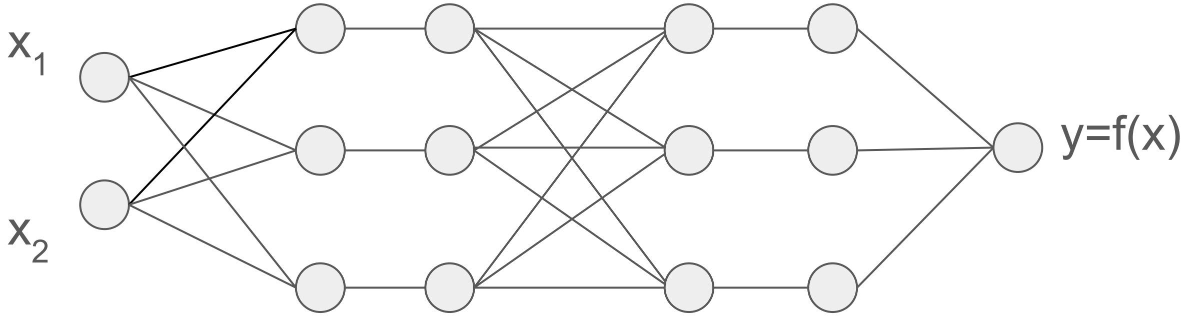
- Interval bound propagation (IBP)
- Linear (symbolic) bound propagation (CROWN)

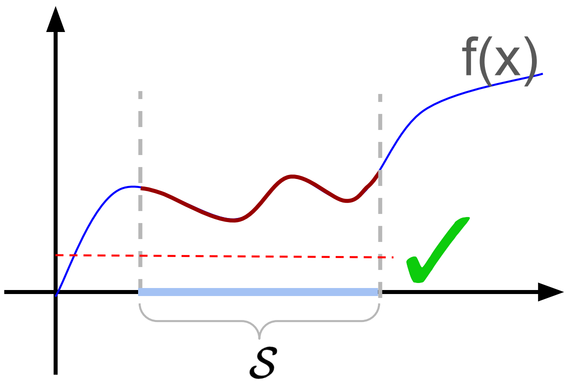
Efficient methods are typically incomplete (solving a lower bound, as tight as possible)



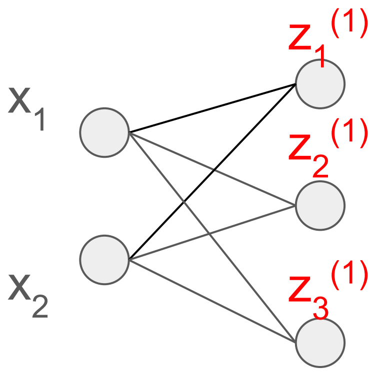
$$y^* = \min_{x \in \mathcal{S}} f(x)$$

Any faster ways to calculate the bounds on f(x)?





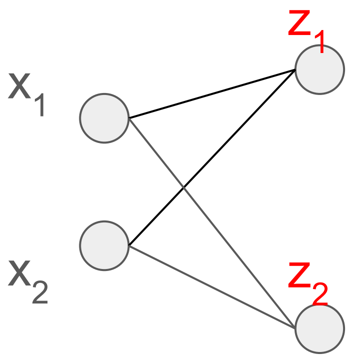
Let's look at one layer first



Given bounds on x, can we calculate the bounds on z?

$$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$$

Let's look at one layer first



Given bounds on x, can we calculate the bounds on z?

$$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$$

As an illustration, suppose we have

$$z_1 = x_1 - x_2$$
$$z_2 = 2x_1 - x_2$$

Can you infer bounds on z given bounds on x?

Interval Bound Propagation (IBP)

$$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$$

$$z_1 = x_1 - x_2$$
$$z_2 = 2x_1 - x_2$$

Lower bounds

$$\underline{z}_1 = -1 - 1 = -2$$

$$\underline{z}_2 = -1 \times 2 - 1 = -3$$

Upper bounds

$$\overline{z}_1 = 2 - (-2) = 4$$

$$\overline{z}_2 = 2 \times 2 - (-2) = 6$$

Interval Bound Propagation (IBP)

$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$ $\quad z_1 = x_1 - x_2 \quad z_2 = 2x_1 - x_2$

$$\underline{z}_1 = -1 - 1 = -2 \qquad \overline{z}_1 = 2 - (-2) = 4$$

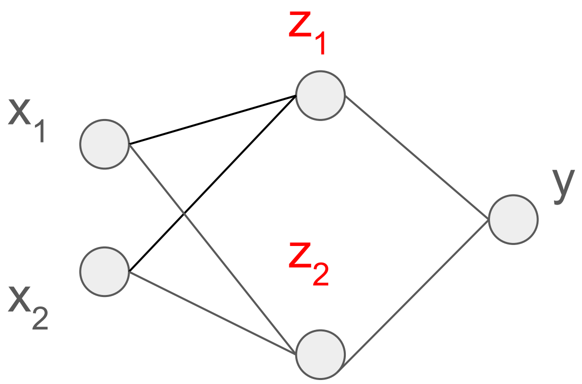
$$\underline{z}_2 = -1 \times 2 - 1 = -3 \qquad \overline{z}_2 = 2 \times 2 - (-2) = 6$$

In general:

$$\sum_{i \in \{i | w_i \geq 0\}} w_i l_i + \sum_{i \in \{i | w_i < 0\}} w_i u_i \leq \sum_i w_i x_i \leq \sum_{i \in \{i | w_i \geq 0\}} w_i u_i + \sum_{i \in \{i | w_i < 0\}} w_i l_i$$

Elements lower and upper bounds of x

Interval Bound Propagation: continue to the next layer



Let's say $y = z_1 - z_2$

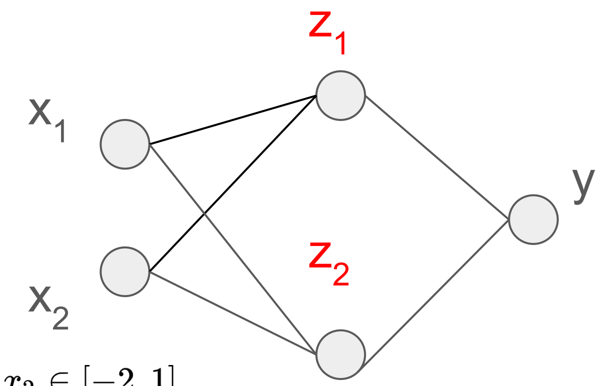
We also know that:

$$z_1 \in [-\mathbf{2}, \mathbf{4}] \qquad z_2 \in [-\mathbf{3}, \mathbf{6}]$$

The what can we conclude about y?

$$y \in [-\mathbf{8}, \mathbf{7}]$$

Interval Bound Propagation: limitations



$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$

Apply IBP we obtain $y \in [-8, 7]$ for this simple linear network.

However observe that

$$z_1 = x_1 - x_2$$

$$z_2 = 2x_1 - x_2$$

$$y = z_1 - z_2$$

$$y = x_1 - x_2 - (2x_1 - x_2) = -x_1$$

The actual bounds is [-2, 1], much tighter than [-8, 7]

A Better Idea: Keep the correlations between x and z

$$z_1 = x_1 - x_2$$
$$z_2 = 2x_1 - x_2$$
$$y = z_1 - z_2$$
$$y = x_1 - x_2 - (2x_1 - x_2) = -x_1$$

The actual bounds is [-2, 1], much tighter than [-8, 7]

It is important to keep the correlations between z and x to obtain this tighter result!

We treat z as a **symbolic function of x**, rather than intervals

A Better Idea: linear bound propagation

$$z_1 = x_1 - x_2$$
$$z_2 = 2x_1 - x_2$$
$$y = z_1 - z_2$$
$$y = x_1 - x_2 - (2x_1 - x_2) = -x_1$$

The actual bounds is [-2, 1], much tighter than [-8, 7]

It is important to keep the correlations between z and x to obtain this tighter result!

We treat z as a **linear function of x**, rather than intervals

A Better Idea: linear bound propagation

$$y = z_1 - z_2 \Longrightarrow y = x_1 - x_2 - (2x_1 - x_2) = -x_1$$

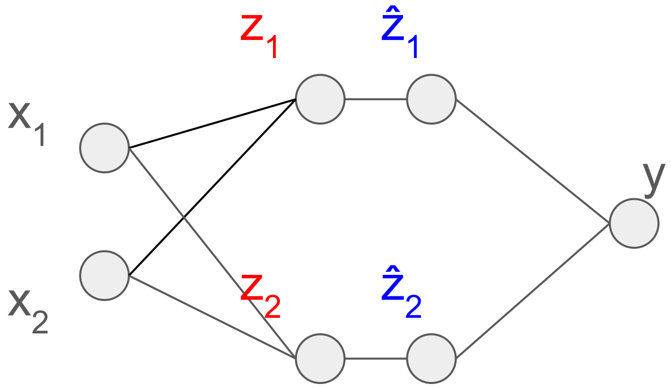
Plug in

$$z_1 = x_1 - x_2$$
$$z_2 = 2x_1 - x_2$$

We treat z as a **linear function of x**, rather than concrete intervals.

After we plug in linear functions (z w.r.t. x), we still get a linear function (y w.r.t. x)

Bound propagation: how about nonlinear functions?



Instead of $y = z_1 - z_2$

Now we have $y = \text{ReLU}(z_1) - \text{ReLU}(z_2)$

From IBP we already know that

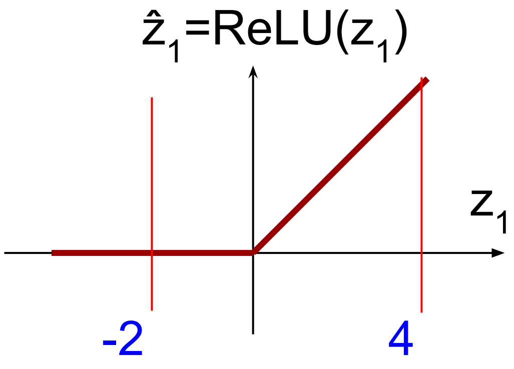
$z_1 \in [-2, 4]$, $z_2 \in [-3, 6]$,

$\text{ReLU}(z_1) \in [0, 4]$, $\text{ReLU}(z_1) \in [0, 6]$

$y \in [-6, 4]$

Can we improve IBP using symbolic linear bounds?

Linear bound propagation for ReLU function (CROWN)



Instead of $y = z_1 - z_2$

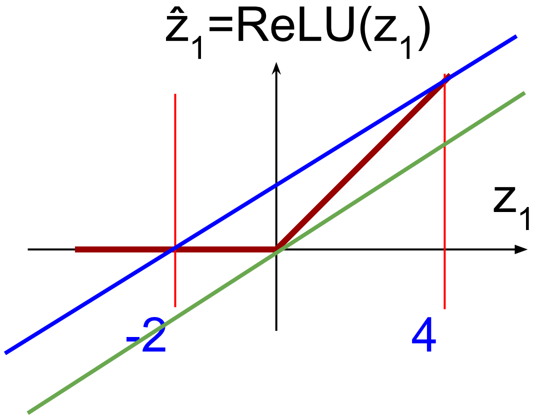
Now we have $y = \text{ReLU}(z_1) - \text{ReLU}(z_2)$

We already know that

$z_1 \in [-2, 4]$, $z_2 \in [-3, 6]$,

(Preactivation bounds)

Linear bound propagation for ReLU function (CROWN)

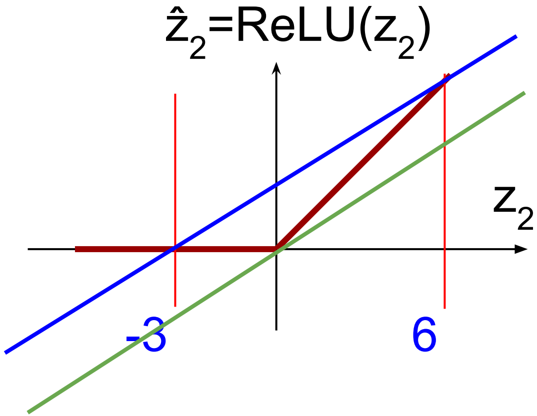


Linear upper bound (same as the one of triangle relaxation in LP)

Linear lower bound (actually not unique)

$$\boxed{\frac{2}{3}z_1} \leq \mathrm{ReLU}(z_1) \leq \boxed{\frac{2}{3}z_1 + \frac{4}{3}}$$

Linear bound propagation for ReLU function (CROWN)



ReLU(z_2) can be bounded using linear functions similarly.

Now let's consider y = ReLU(z_1) - ReLU(z_2). How to bound it using linear functions of z_1 and z_2?

$$\frac{2}{3}z_1 \leq \mathrm{ReLU}(z_1) \leq \frac{2}{3}z_1 + \frac{4}{3}$$

$$\frac{2}{3}z_2 \leq \mathrm{ReLU}(z_2) \leq \frac{2}{3}z_2 + 2$$

Linear bound propagation for ReLU function (CROWN)

$$\boxed{\frac{2}{3}z_1} \le \mathrm{ReLU}(z_1) \le \frac{2}{3}z_1 + \frac{4}{3} \qquad \frac{2}{3}z_2 \le \mathrm{ReLU}(z_2) \le \boxed{\frac{2}{3}z_2 + 2}$$

Negative coefficient, take upper bound

$$\boxed{\frac{2}{3}z_1} - \boxed{\left(\frac{2}{3}z_2 + 2\right)} \le$$

positive coefficient, take lower bound

$$y = \mathrm{ReLU}(z_1) - \mathrm{ReLU}(z_2)$$

$$\le \left(\frac{2}{3}z_1 + \frac{4}{3}\right) - \frac{2}{3}z_2$$

Linear bound propagation for ReLU function (CROWN)

$$\frac{2}{3}z_1 - (\frac{2}{3}z_2 + 2) \leq y \leq (\frac{2}{3}z_1 + \frac{4}{3}) - \frac{2}{3}z_2$$

Now we have linear inequalities for y w.r.t. z!

Next step we can simply plug in, as in the linear ($y=z_1-z_2$) case.

Linear bound propagation for ReLU function (CROWN)

$$\frac{2}{3}z_1 - (\frac{2}{3}z_2 + 2) \le y \le (\frac{2}{3}z_1 + \frac{4}{3}) - \frac{2}{3}z_2$$

$$z_1 = x_1 - x_2$$
$$z_2 = 2x_1 - x_2$$

Plug in

$$-\frac{2}{3}x_1 - 2 \le y \le -\frac{2}{3}x_1 + \frac{4}{3}$$

Linear bound propagation for ReLU function (CROWN)

We now have symbolic linear bounds for y w.r.t. x

$$-\frac{2}{3}x_1 - 2 \leq y \leq -\frac{2}{3}x_1 + \frac{4}{3}$$

$$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$$

Take lower bound given x

Take upper bound given x

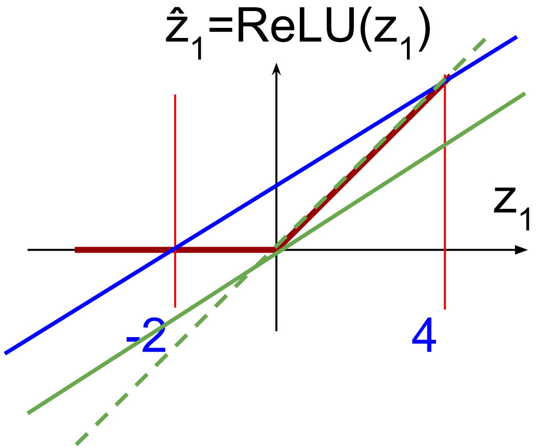
$$y \in [-\frac{10}{3}, 2]$$

Concrete interval bounds

A lot more tighter than IBP bounds y ∈ [-6, 4]

Can we do even better?

Let's recall that when we linearly bound the ReLU function, there are some flexibilities



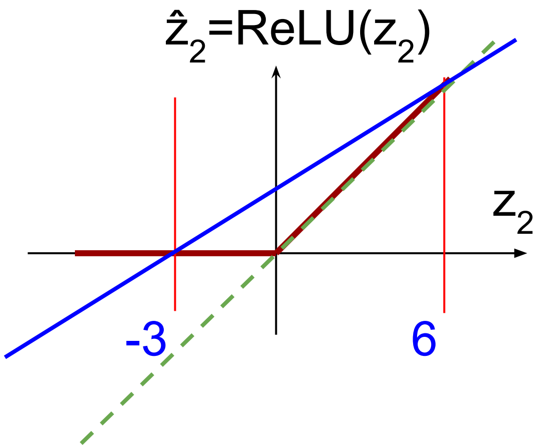
Linear upper bound (same as the one of triangle relaxation in LP)

Linear lower bound (actually not unique)

$$\frac{2}{3} z_1 \leq \mathrm{ReLU}(z_1) \leq \frac{2}{3} z_1 + \frac{4}{3}$$

Also valid: $\boxed{z_1} \leq \mathrm{ReLU}(z_1) \leq \frac{2}{3} z_1 + \frac{4}{3}$

Choosing different linear bounds (α-CROWN)



Now what are the linear bounds of
y = ReLU(z_1) - ReLU(z_2)?

$$z_1 - \left(\frac{2}{3}z_2 + 2\right) \leq y \leq \left(\frac{2}{3}z_1 + \frac{4}{3}\right) - z_2$$

$$z_1 \leq \mathrm{ReLU}(z_1) \leq \frac{2}{3}z_1 + \frac{4}{3}$$

$$z_2 \leq \mathrm{ReLU}(z_2) \leq \frac{2}{3}z_2 + 2$$

Choosing different linear bounds (α-CROWN)

$$z_1 - (\tfrac{2}{3}z_2 + 2) \leq y \leq (\tfrac{2}{3}z_1 + \tfrac{4}{3}) - z_2$$

Plug in

$$z_1 = x_1 - x_2$$

$$z_2 = 2x_1 - x_2$$

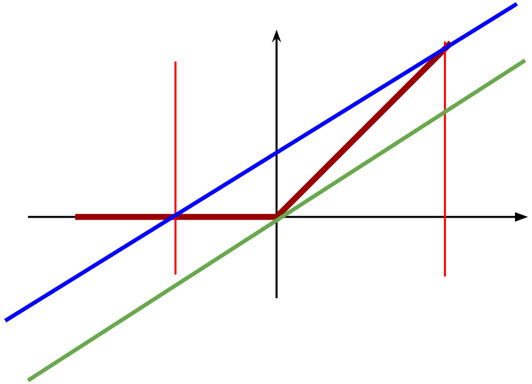
$$-\tfrac{1}{3}x_1 - \tfrac{1}{3}x_2 - 2 \leq y \leq -\tfrac{4}{3}x_1 + \tfrac{1}{3}x_2 + \tfrac{4}{3}$$

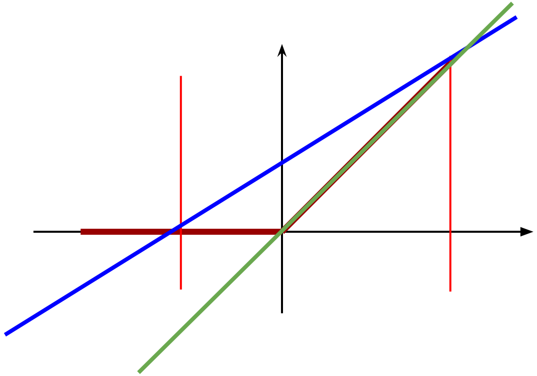
Concretize

$$x_1 \in [-1, 2],\ x_2 \in [-2, 1]$$

$$y \in [-3, 3]$$

Linear lower bounds for ReLU function matters!





$$\frac{2}{3}z_1 \leq \text{ReLU}(z_1) \leq \frac{2}{3}z_1 + \frac{4}{3}$$

$$\frac{2}{3}z_2 \leq \text{ReLU}(z_2) \leq \frac{2}{3}z_2 + 2$$

$$z_1 \leq \text{ReLU}(z_1) \leq \frac{2}{3}z_1 + \frac{4}{3}$$

$$z_2 \leq \text{ReLU}(z_2) \leq \frac{2}{3}z_2 + 2$$

$$y \in [-\frac{10}{3}, 2]$$

Which one is correct?

$$y \in [-3, 3]$$

Linear lower bounds for ReLU function matters!

Both results are correct! But we want the bounds to be as tight as possible! So best result is $\mathbf{y \in [-3, 2]}$

In general, the slope of the linear lower bound for every ReLU neuron can be optimized to find the best result.

Linear lower bounds for ReLU function matters!

In general, the slope of the linear lower bound for every ReLU neuron can be optimized to find the best result.

$$\alpha_1 z_2 \leq \mathrm{ReLU}(z_2) \leq \frac{2}{3} z_2 + \frac{4}{3}$$

$$\alpha_2 z_2 \leq \mathrm{ReLU}(z_2) \leq \frac{2}{3} z_2 + 2$$

For optimal lower bound of y, set $\alpha_1=1$, $\alpha_2=1$

For optimal upper bound of y, set $\alpha_1=2/3$, $\alpha_2=2/3$

(note that the optimal α_1 and α_2 do not equal in general)

Linear bound propagation method (CROWN)

1. Obtain all pre-activation bounds (can be done via CROWN recursively)
2. Start from the output layer, form the initial linear (in)equality y = y
3. Recursively propagate linear inequality y <= $a^T z$ + b through each layer:
   a. For a linear layer, z=Wz', directly plug in $a^T z$ + b to get a linear bound of z'
   b. For a non-linear layer (e.g., z=ReLU(z')), we first form the linear inequalities to bound the nonlinear layer itself. Then multiply either the lower or upper bound based on the sign of element in a
4. When the linear inequality propagates to the input layer, we can concretize the linear bound using bounds on input layer.

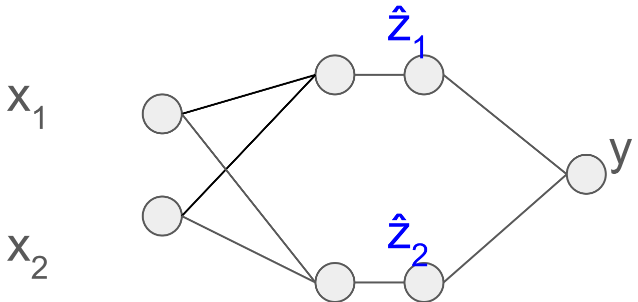
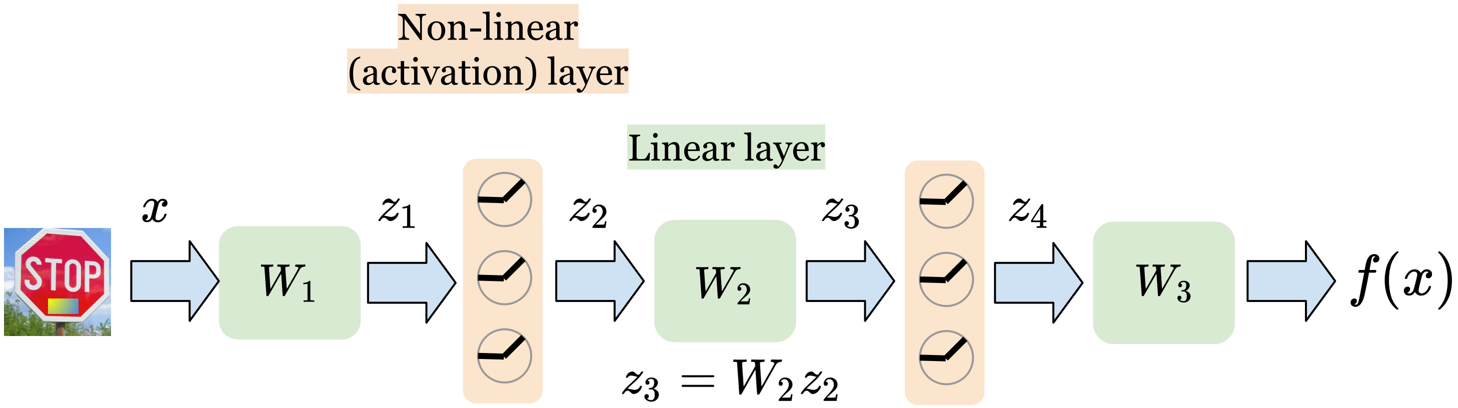
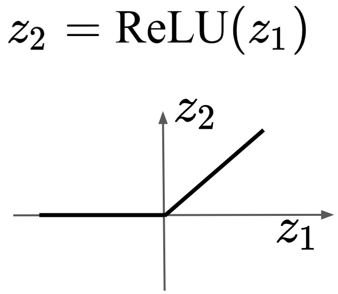


Illustration: Linear bound propagation process



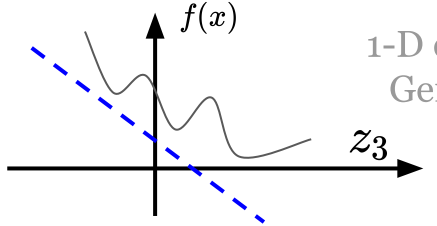


Steps:

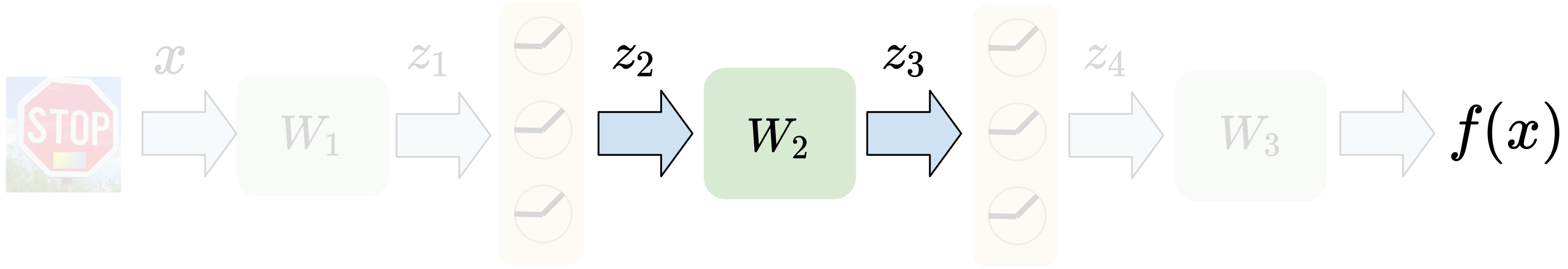
- Propagate bounds through linear layers
- Propagate bounds through non-linear layers

Illustration: Linear bound propagation process

A linear lower bound for an intermediate layer

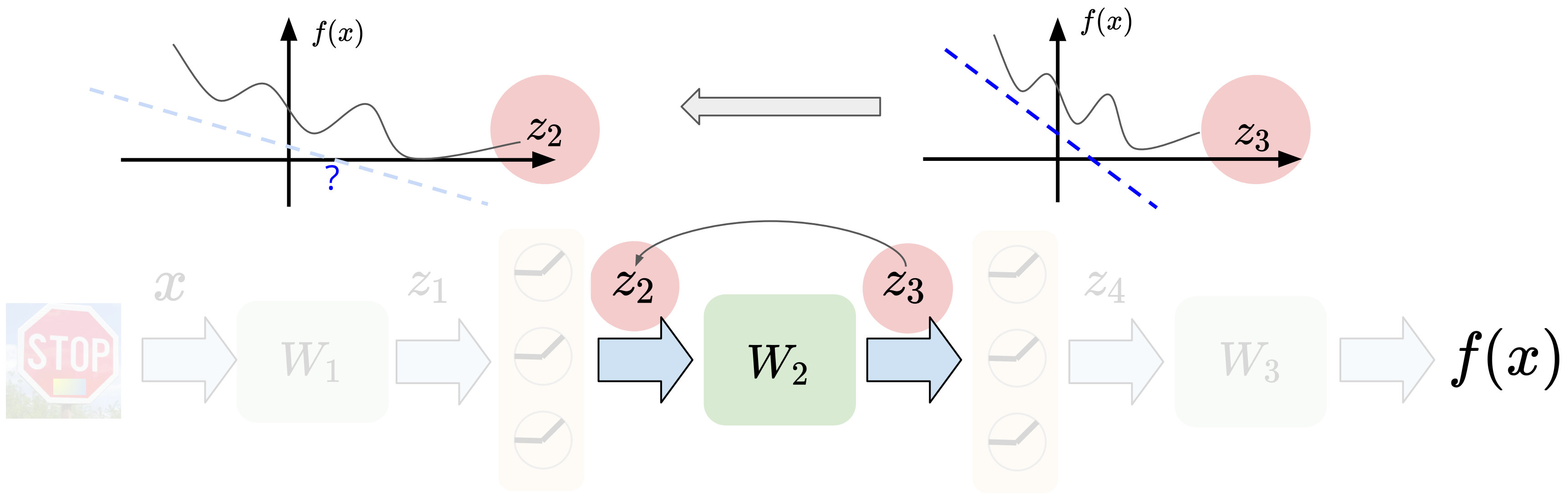


1-D case for illustration. Generally it's a linear hyperplane



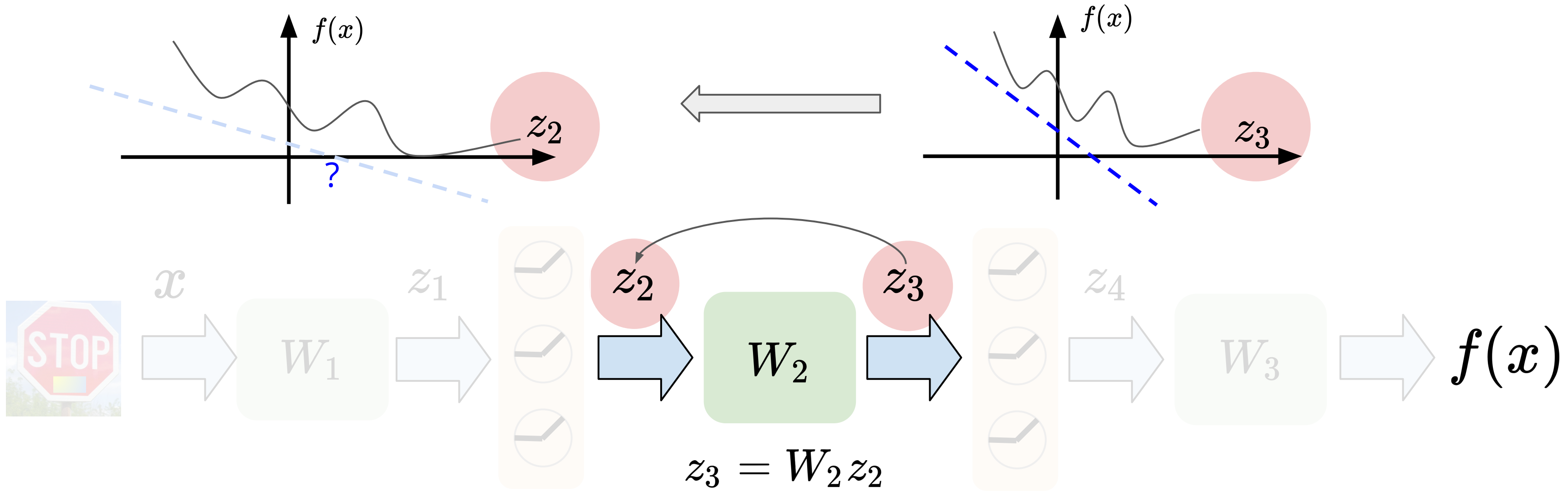
$$f(x) \geq a^\top z_3 + b$$

Illustration: Linear bound propagation process



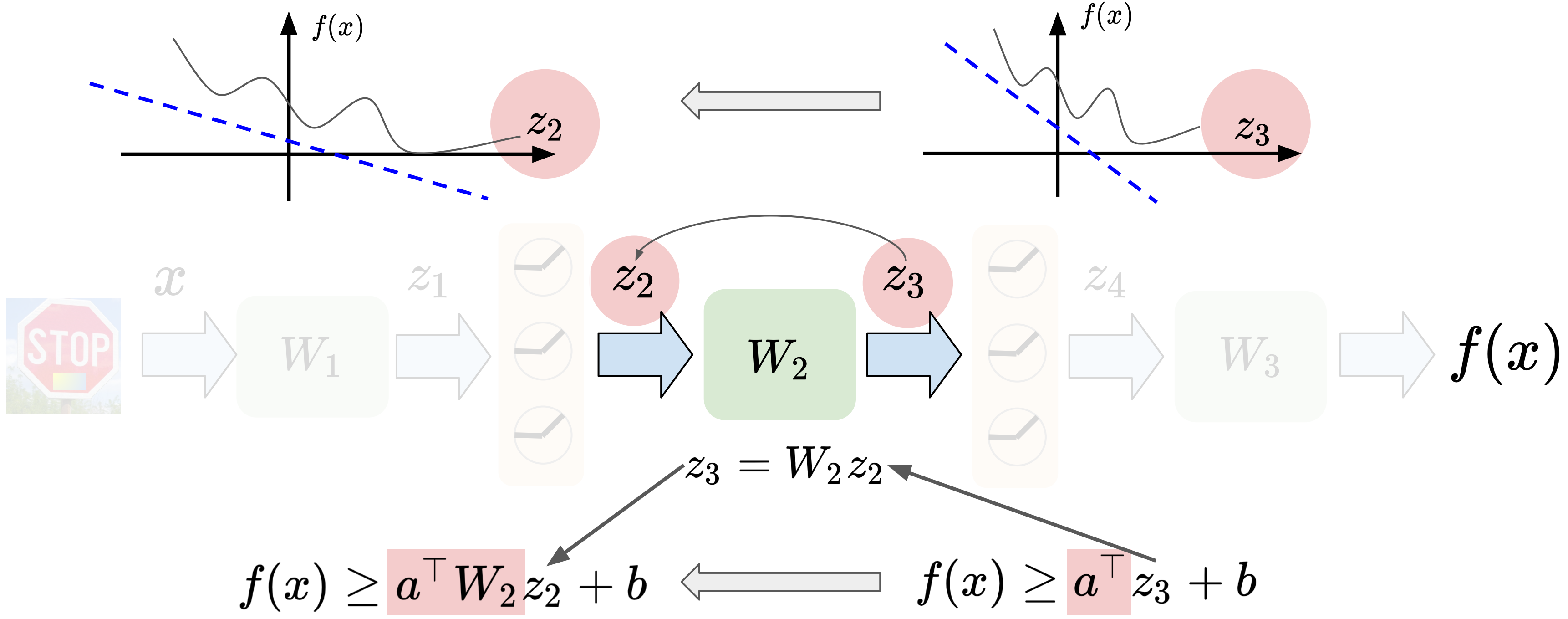
Propagate it to one layer before,
while keeping the lower bound valid

Illustration: Linear bound propagation process



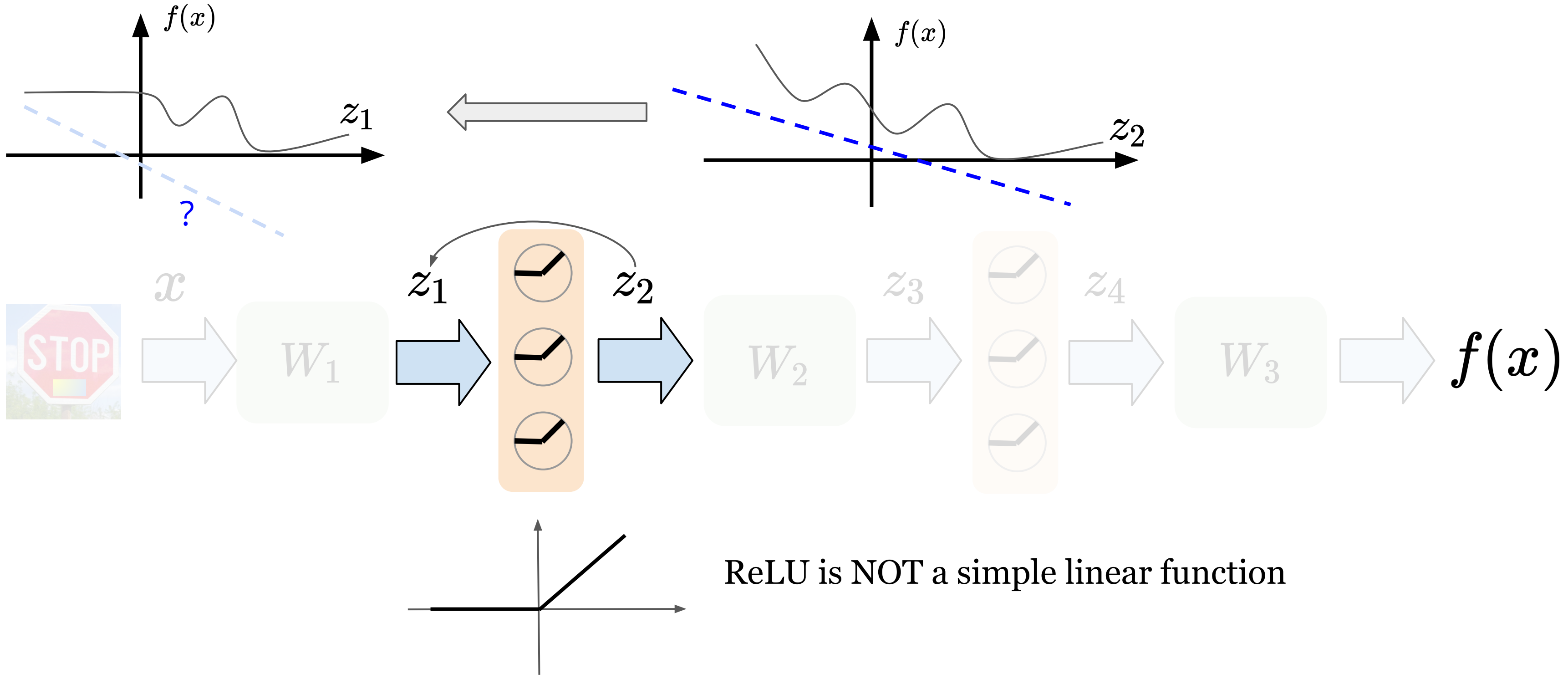
$$f(x) \geq a^\top z_3 + b$$

Illustration: Linear bound propagation process



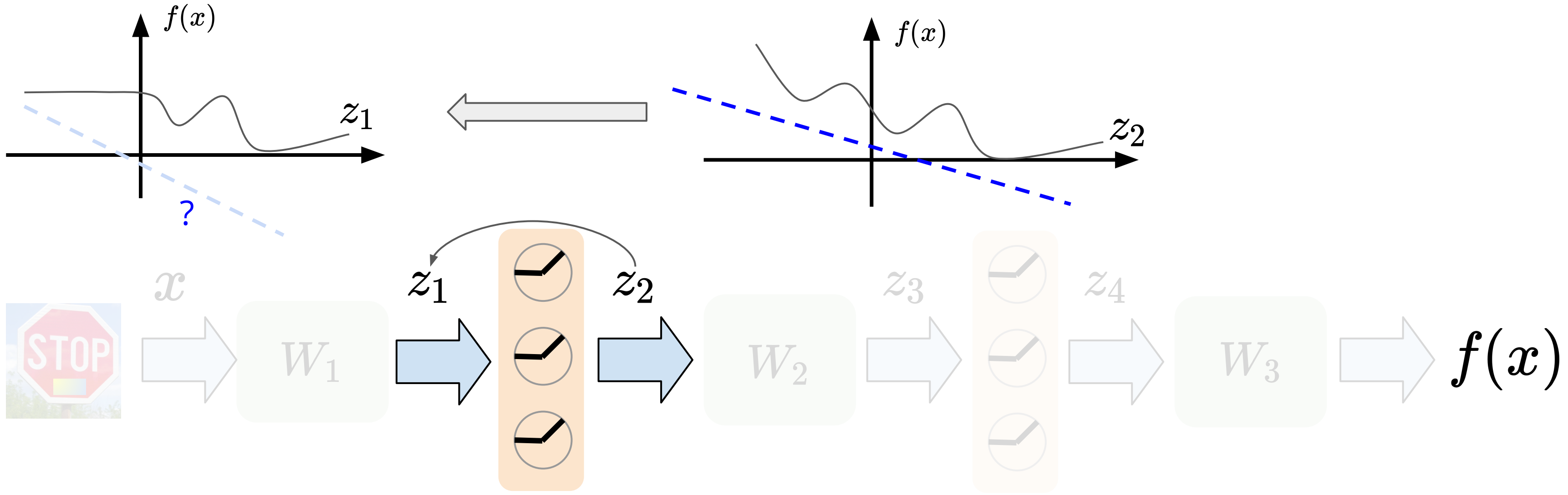
Inequality updated after propagation

Illustration: Linear bound propagation process



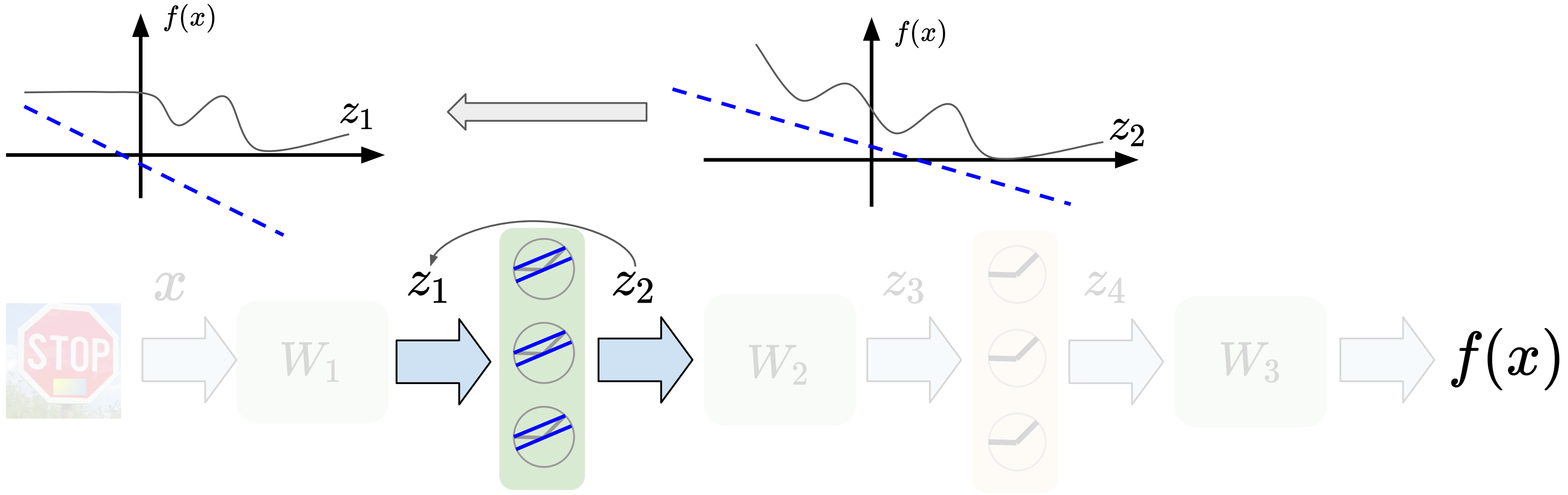
ReLU is NOT a simple linear function

Illustration: Linear bound propagation process



$$f(x) \geq a^\top W_2 z_2 + b \quad \forall x \in \mathcal{S}$$

Illustration: Linear bound propagation process



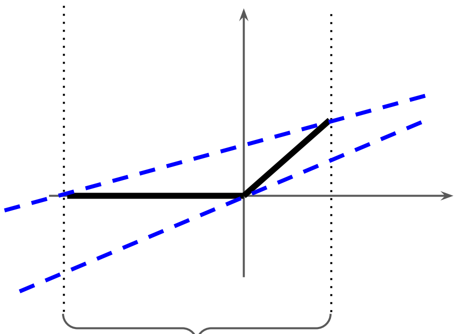
Theorem (informal): we can efficiently find $D,\ b'$ such that:

$$f(x) \geq a^\top W_2 D z_1 + b' \Longleftarrow f(x) \geq a^\top W_2 z_2 + b \quad \forall x \in \mathcal{S}$$

[**Z***W*CHD NeurIPS 2018]

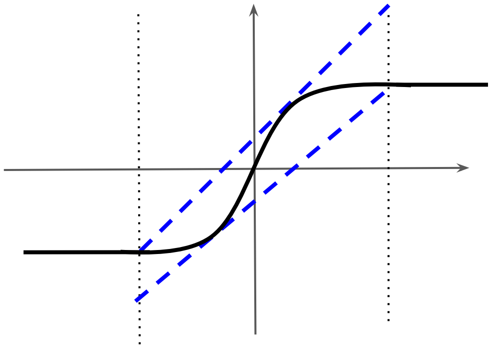
Illustration: Linear bound propagation process

Proof sketch: conservatively use linear bounds to replace a non-linear function.



Pre-activation bounds

(can be pre-computed using CROWN)



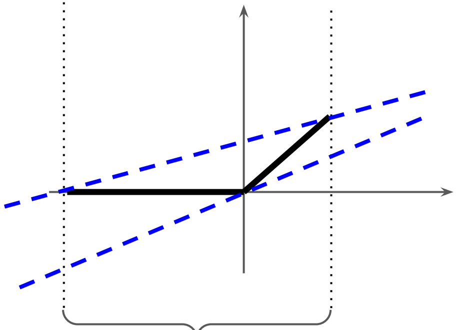
Theorem (informal): we can efficiently find $D,\ b'$ such that:

$$f(x) \geq a^\top W_2 D z_1 + b' \Longleftarrow f(x) \geq a^\top W_2 z_2 + b \quad \forall x \in \mathcal{S}$$

[Z*W*CHD NeurIPS 2018]

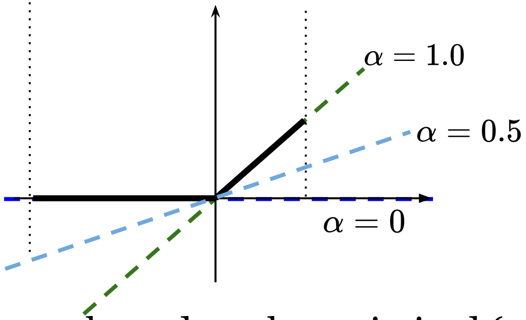
Illustration: Linear bound propagation process

Proof sketch: conservatively use linear bounds to replace a non-linear function.



Pre-activation bounds

(can be pre-computed using CROWN)



ReLU's lower bound can be optimized ($\boldsymbol{\alpha}$-CROWN)

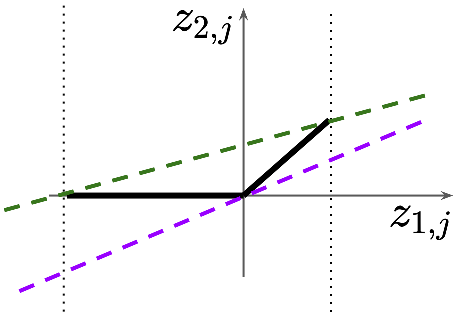
Theorem (informal): we can efficiently find $D,\ b'$ such that:

$$f(x) \geq \mathbf{a}^\top W_2 D z_1 + b' \Longleftarrow f(x) \geq \mathbf{a}^\top W_2 z_2 + b \quad \forall x \in \mathcal{S}$$

[**Z***W*CHD NeurIPS 2018]

Illustration: Linear bound propagation process

Proof sketch: conservatively use linear bounds to replace a non-linear function.



$$f(x) \geq a^\top W_2 z_2 + b$$

$$f(x) \geq \sum_j \left[(a^\top W_2)_j \cdot z_{2,j} \right] + b$$

$(a^\top W_2)_j \geq 0$ Choose lower bound

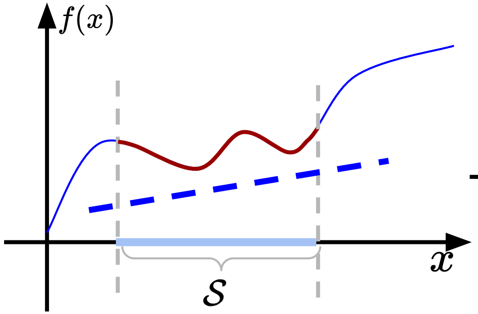
$(a^\top W_2)_j < 0$ Choose upper bound

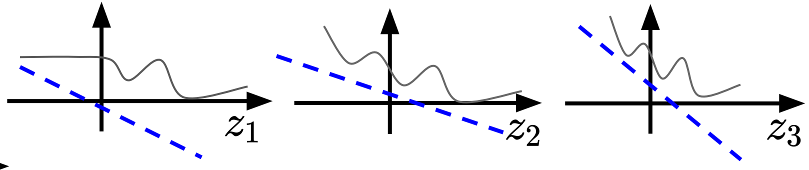
Theorem (informal): we can efficiently find $D,\ b'$ such that:

$$f(x) \geq a^\top W_2 D z_1 + b' \Longleftarrow f(x) \geq a^\top W_2 z_2 + b \quad \forall x \in \mathcal{S}$$

[**Z***W*CHD NeurIPS 2018]

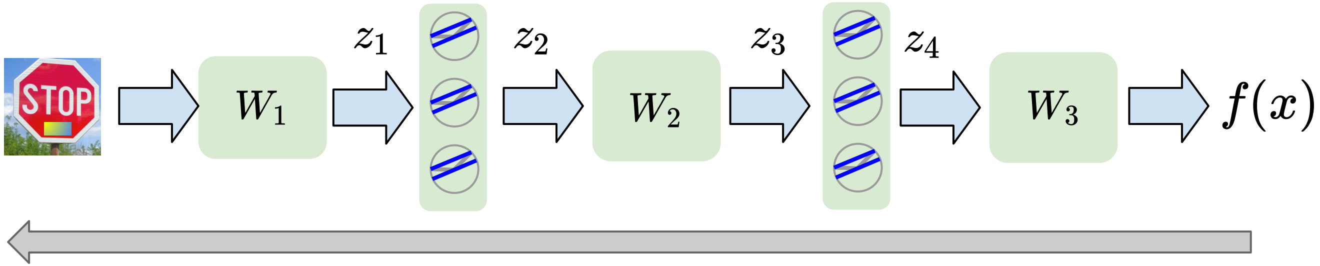
Illustration: Linear bound propagation process





Initial (in)equality:

$$f(x) = W_3 z_4$$

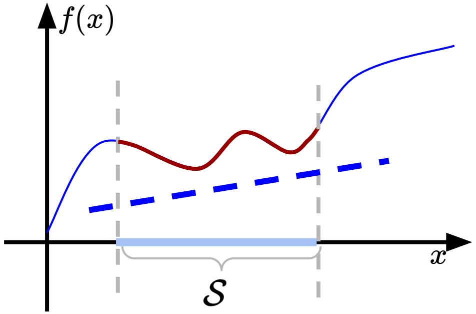


Always keep valid lower bounds

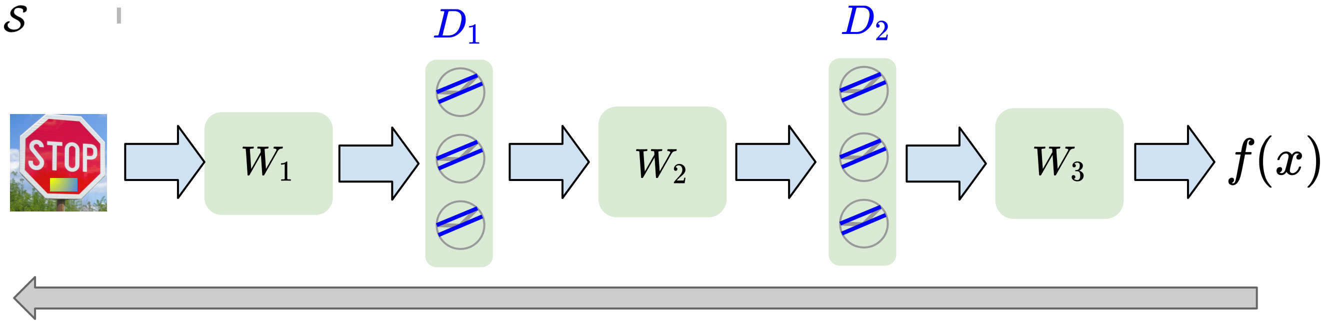
CROWN main theorem (simplified): $f(x) \geq a_{\text{CROWN}}^{\top} x + b_{\text{CROWN}} \quad \forall x \in \mathcal{S}$

[Z*W*CHD NeurIPS 2018]

Illustration: Linear bound propagation process



Bounds propagated through simple matrix multiplations! Fast and GPU-friendly



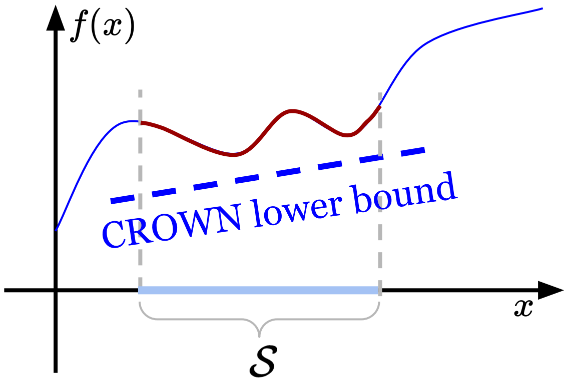
Always keep valid lower bounds

CROWN main theorem (simplified): $f(x) \geq a_{\text{CROWN}}^\top x + b_{\text{CROWN}} \quad \forall x \in \mathcal{S}$

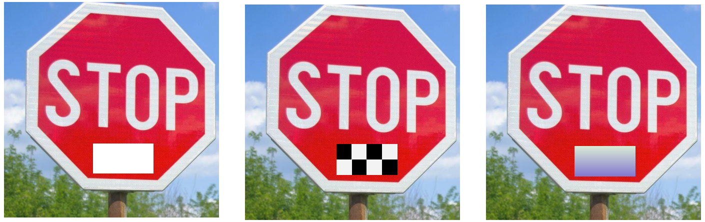
$$a_{\text{CROWN}} = W_3 D_2 W_2 D_1 W_1$$

[**Z***W*CHD NeurIPS 2018]

Use Linear Bounds to Prove Robustness



Prove: $\forall x \in \mathcal{S},\ f(x) > 0$



$x_1 \in \mathcal{S}$ $x_2 \in \mathcal{S}$ $x_3 \in \mathcal{S}$ …

Lower bound > 0 ⇨ $f(x) > 0$ ⇨ verified (always a stop sign)

auto_LiRPA: Verification Library for General Computation Graphs

Colab Demo:

http://PaperCode.cc/AutoLiRPA-Demo

The auto_LiRPA library on GitHub:

http://PaperCode.cc/AutoLiRPA

MILP/LP vs Bound Propagation

Bound propagation:
- Scalable and fast propagation
- GPU friendly
- Incomplete verification (will be extended in the next lecture)
- Bounds are looser compared to LP; much looser compared to MILP

MILP/LP:
- Tighter solution
- Does no scale (MILP ~10k neurons, LP ~100k neurons)
- Much slower; cannot utilize GPU