# Welcome to

BS6207

2022

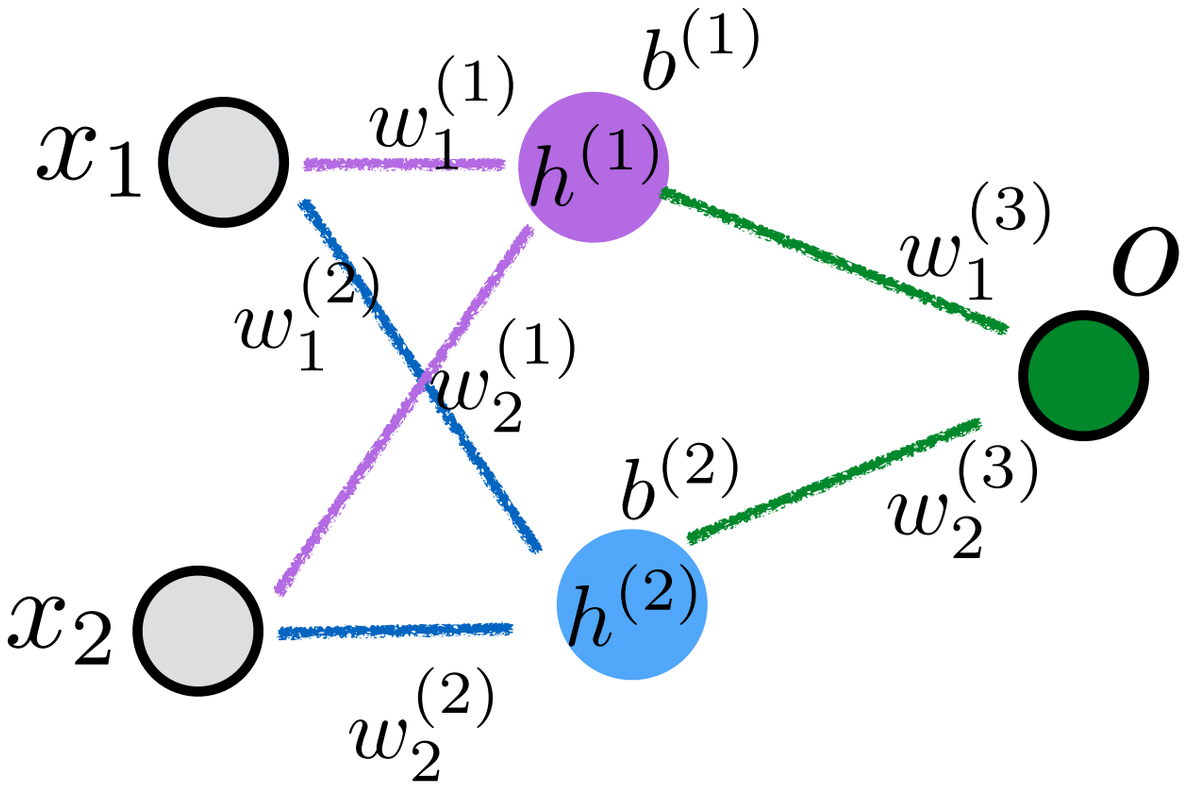
Lee Hwee Kuan

# Course schedule

Lecture 1: Nov 11
Lecture 2: Nov12
Lecture 3: Nov18
Lecture 4: Dec 09 6:30pm
Lecture 5: Dec16 6:30pm
Lecture 6: Jan 06 6:30pm
Lecture 7: Jan 07 7:00pm - Sat
Lecture 8:  Jan 13 6:30pm
Lecture 9: Jan 14 6:30pm - Sat
Lecture 10: Jan 20 6:30pm
Lecture 11: Jan 27 6:30pm
Lecture 12: Feb 03 6:30pm
Lecture 13: Feb 10 6:30pm

# Short review

# Basic construction of neural networks

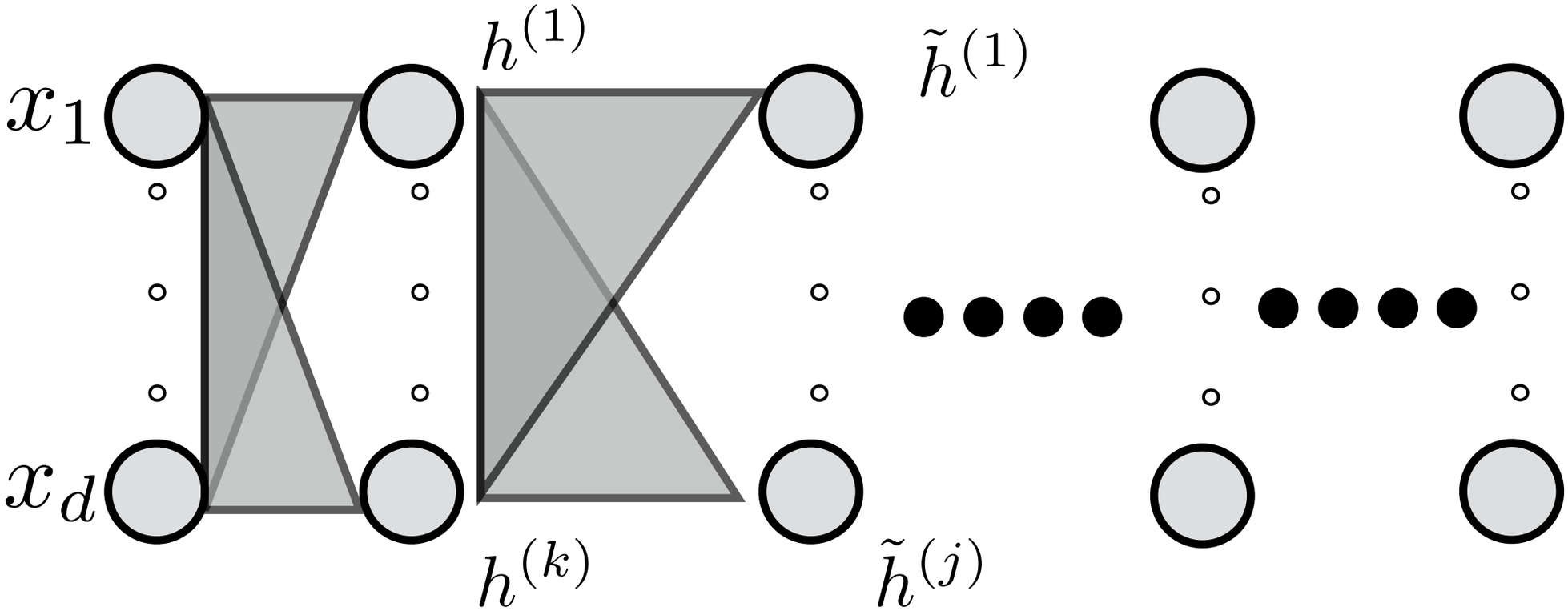



$$h^{(1)} = \sigma(w_1^{(1)} x_1 + w_2^{(1)} x_2 + b^{(1)})$$

$$h^{(2)} = \sigma(w_1^{(2)} x_1 + w_2^{(2)} x_2 + b^{(2)})$$

$$o = \sigma(w_1^{(3)} h^{(1)} + w_2^{(3)} h^{(2)} + b^{(3)})$$

# Basic construction of neural networks

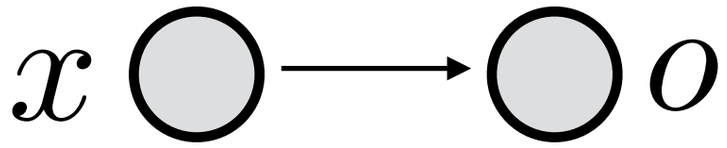


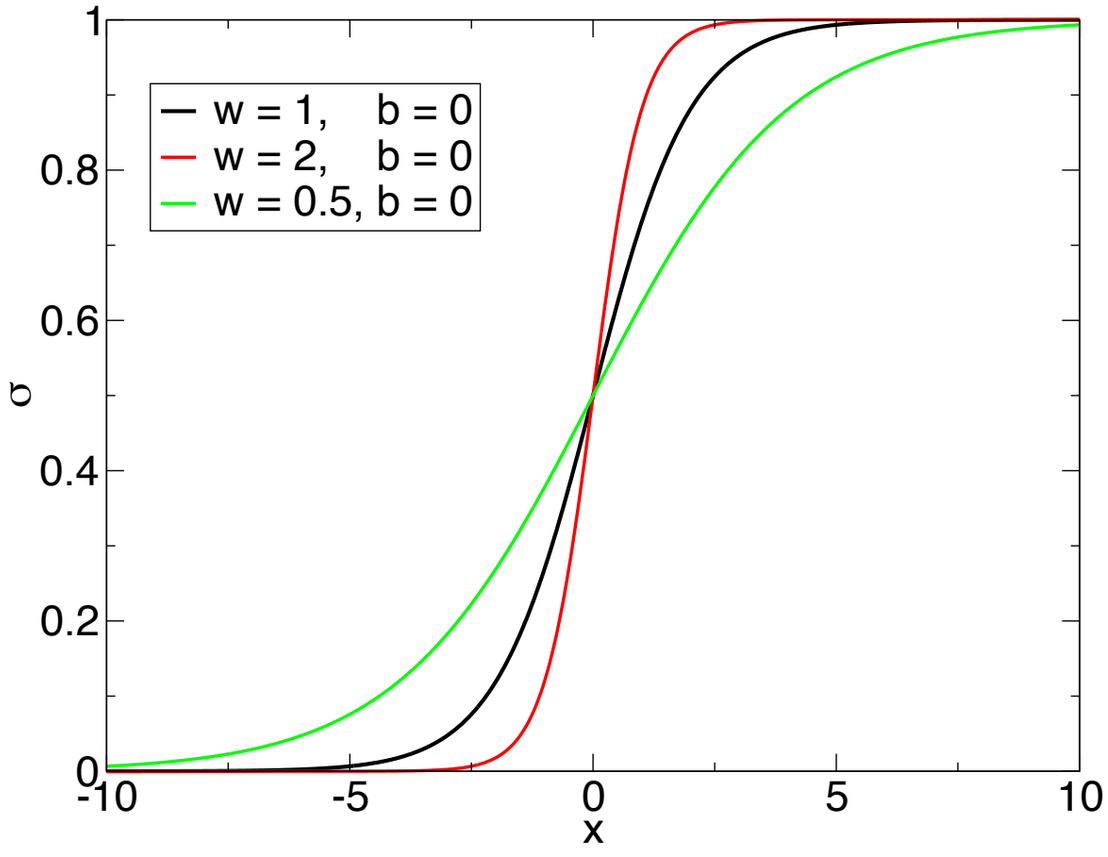

$$h^{(1)} = \sigma(\vec{w}_1 \cdot \vec{x} + b_1)$$
$$h^{(k)} = \sigma(\vec{w}_k \cdot \vec{x} + b_k)$$
$$\tilde{h}^{(j)} = \sigma(\vec{w}_j \cdot \vec{h} + b_j)$$

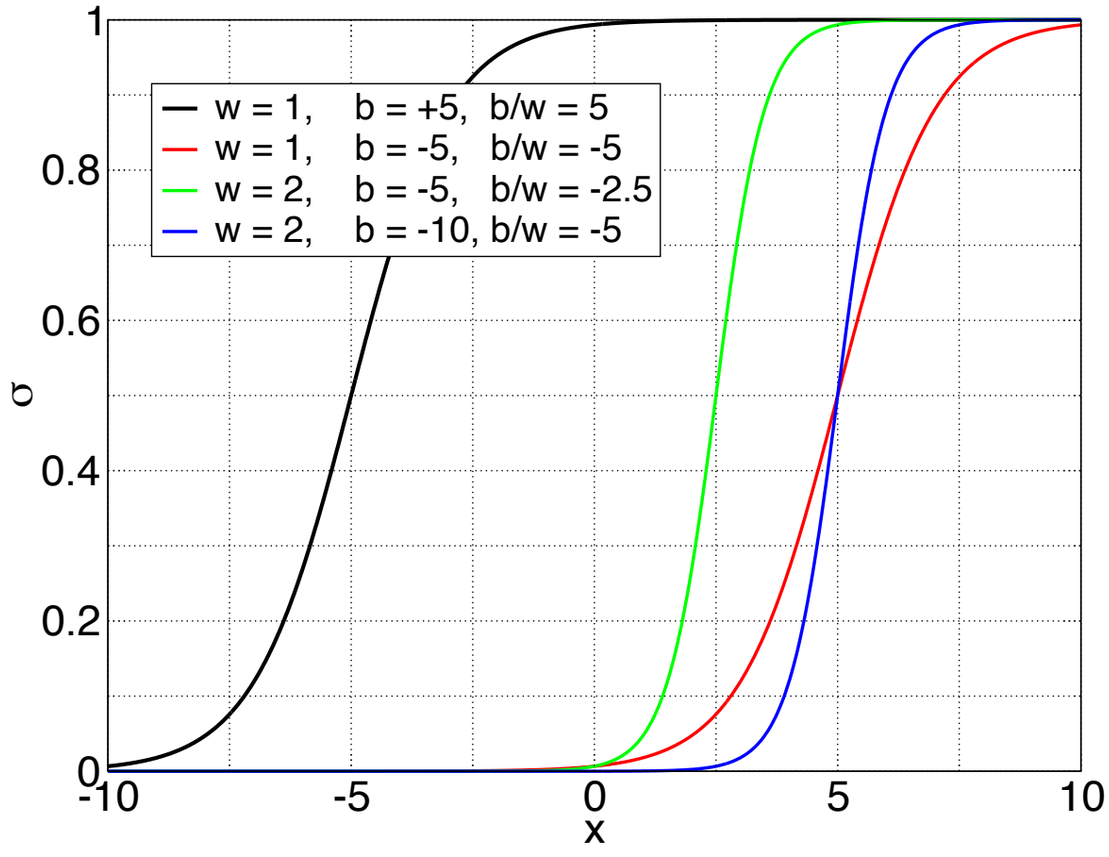
# Role of parameters $w$ and $b$

$$x \in \mathbb{R}$$

$$o = \frac{1}{1 + \exp(-wx - b)}$$



# Role of parameters $w$ and $b$

Do it yourself : plot ReLU graph for various $w$ $b$

Desmos

# Mathematical Preliminaries

# Functions

# Functions

Given 3 numbers, I manipulate them and return you one number

# Functions

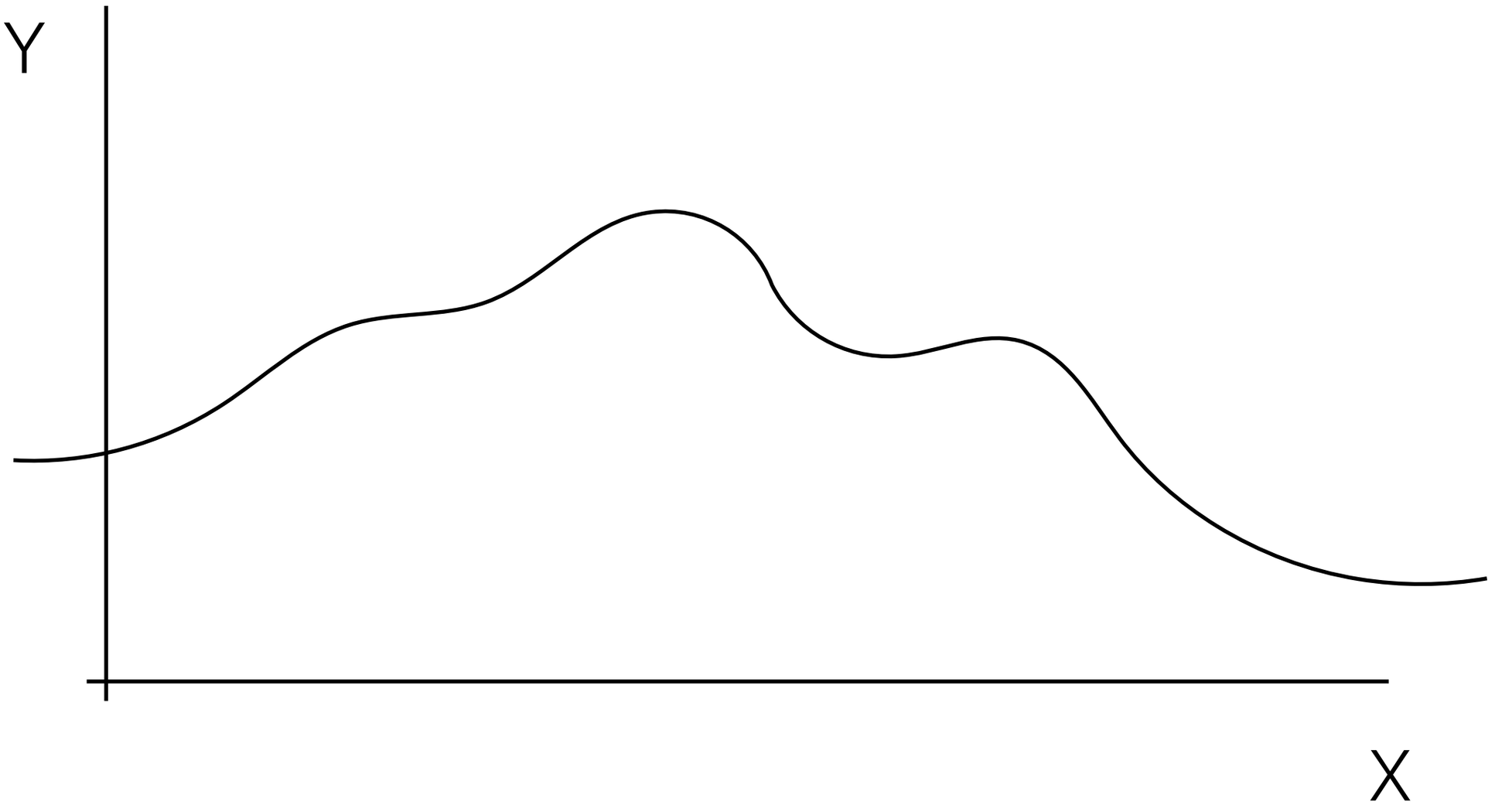
Given 3 numbers, I manipulate them and return you one number

$$f(x1, x2, x3) = y$$

# Functions

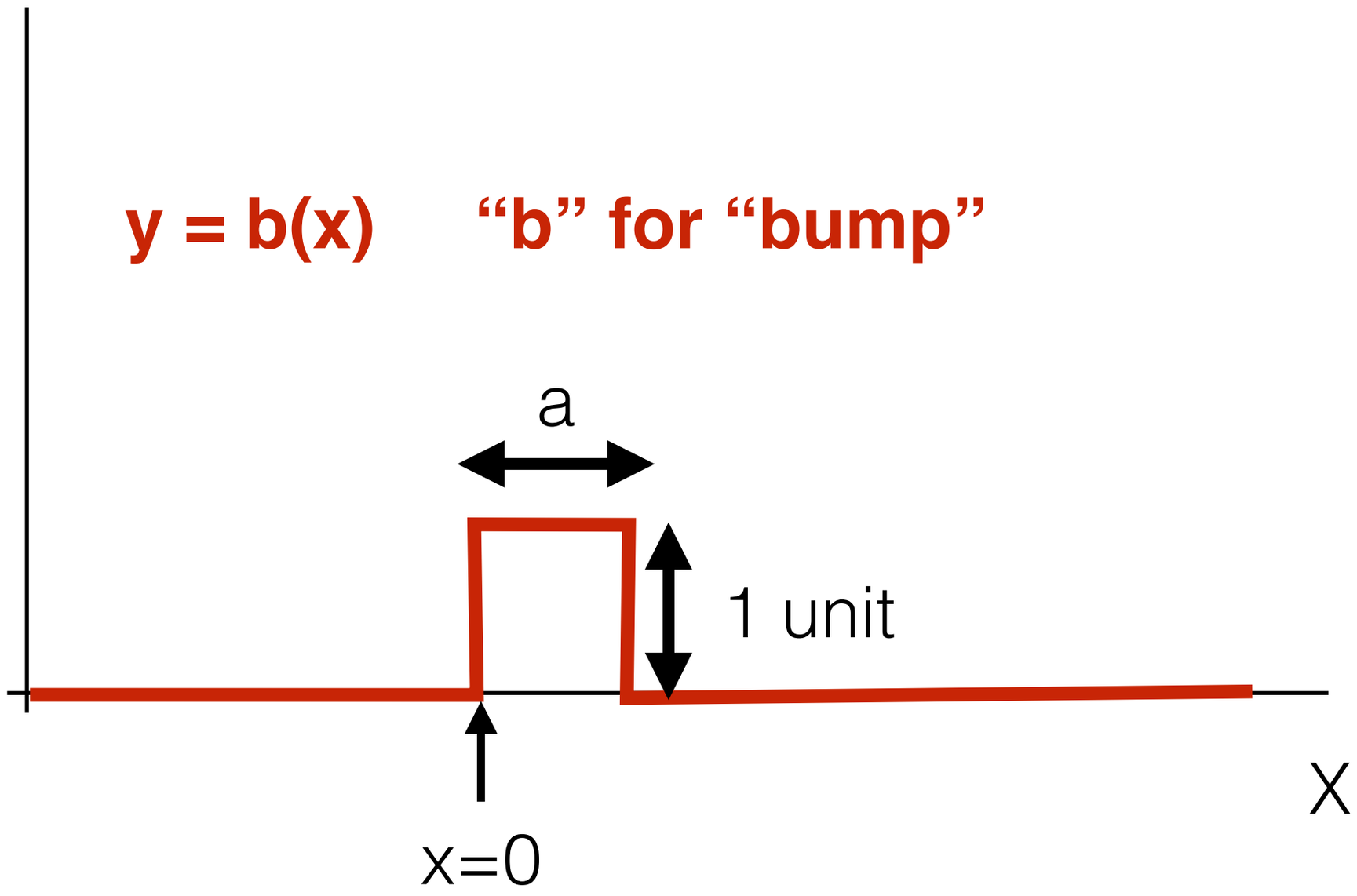
Given 1 number, I manipulate it and return you one number

$$f(x) = y$$

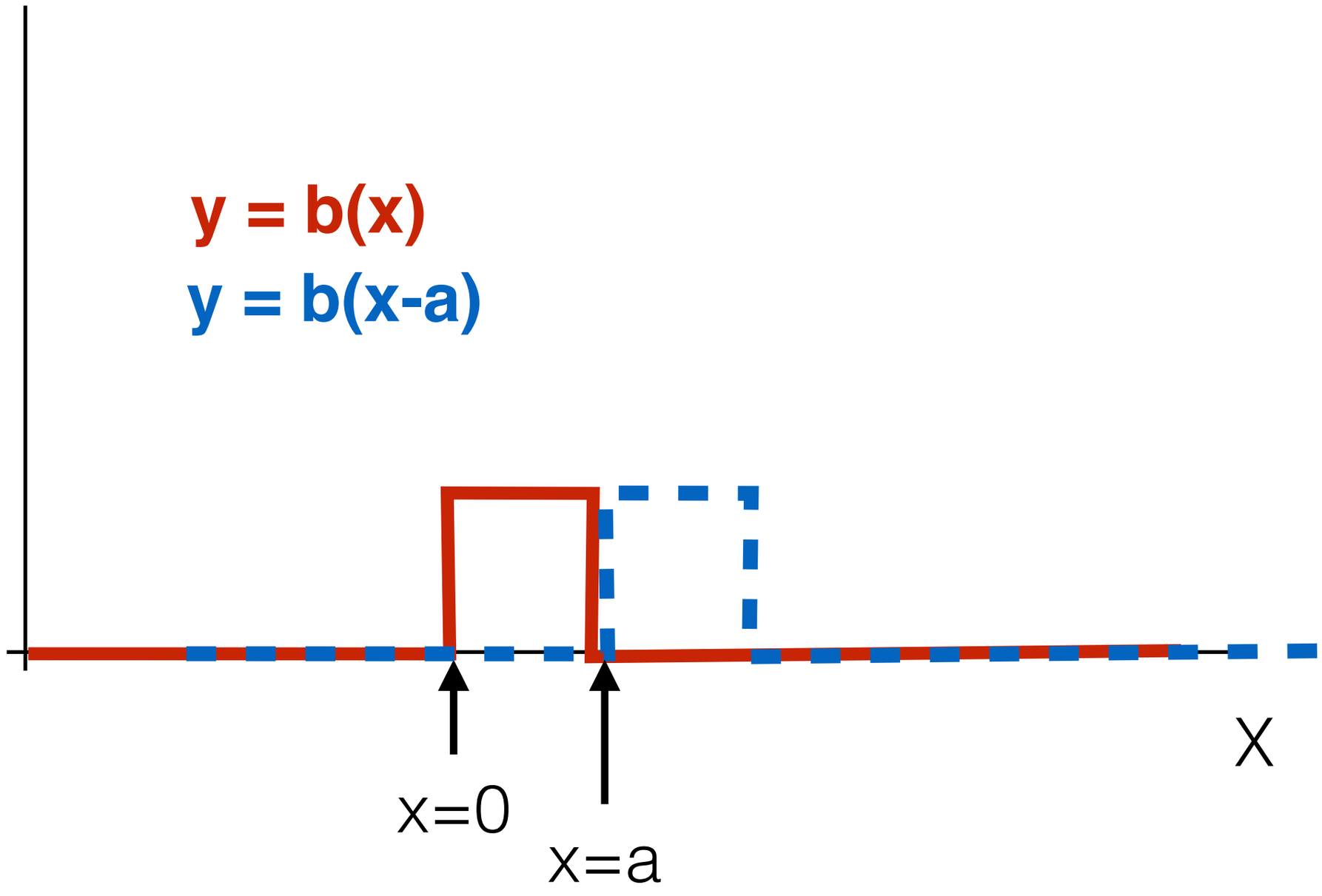


# Building a function

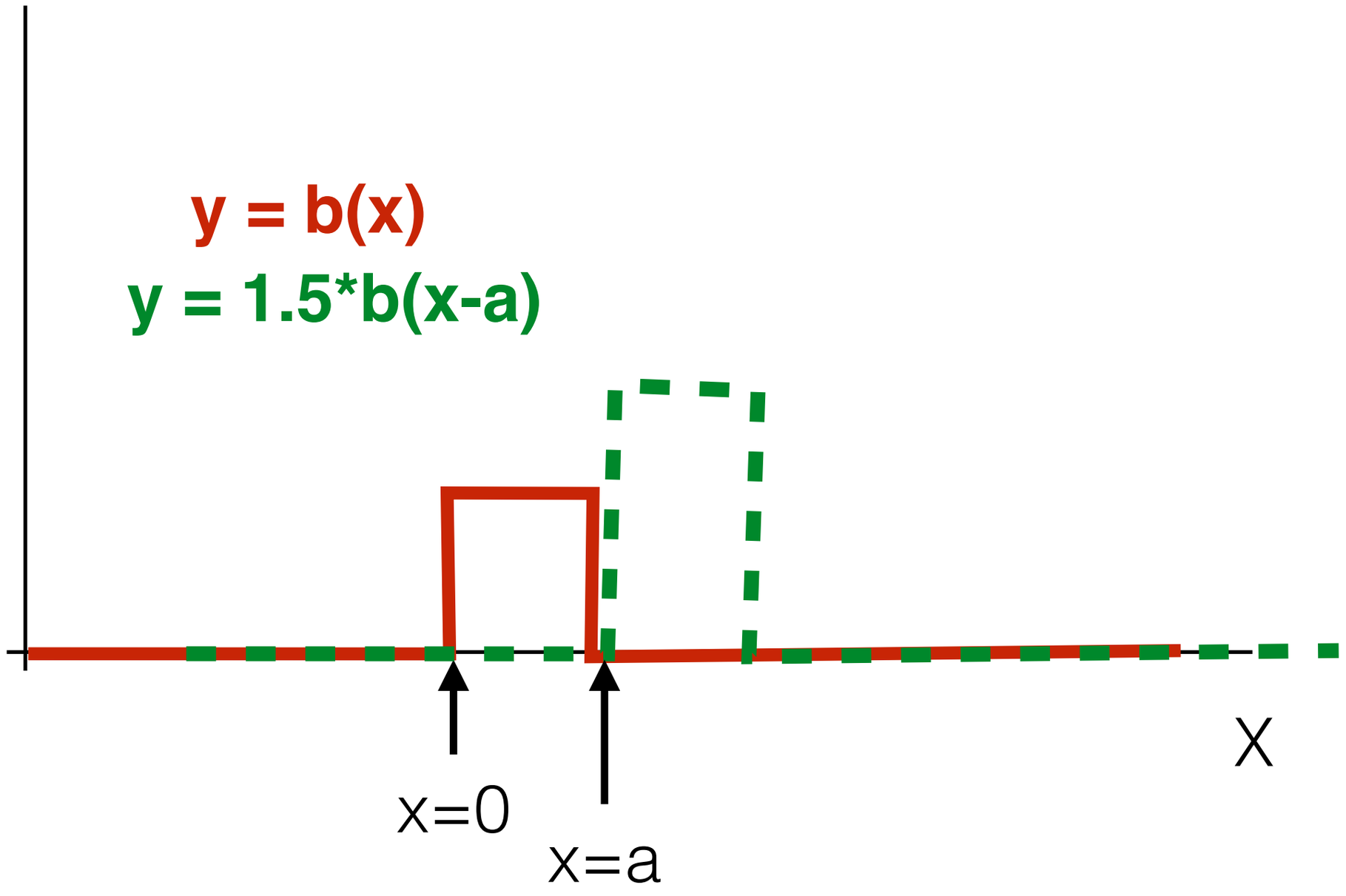
# Building a function



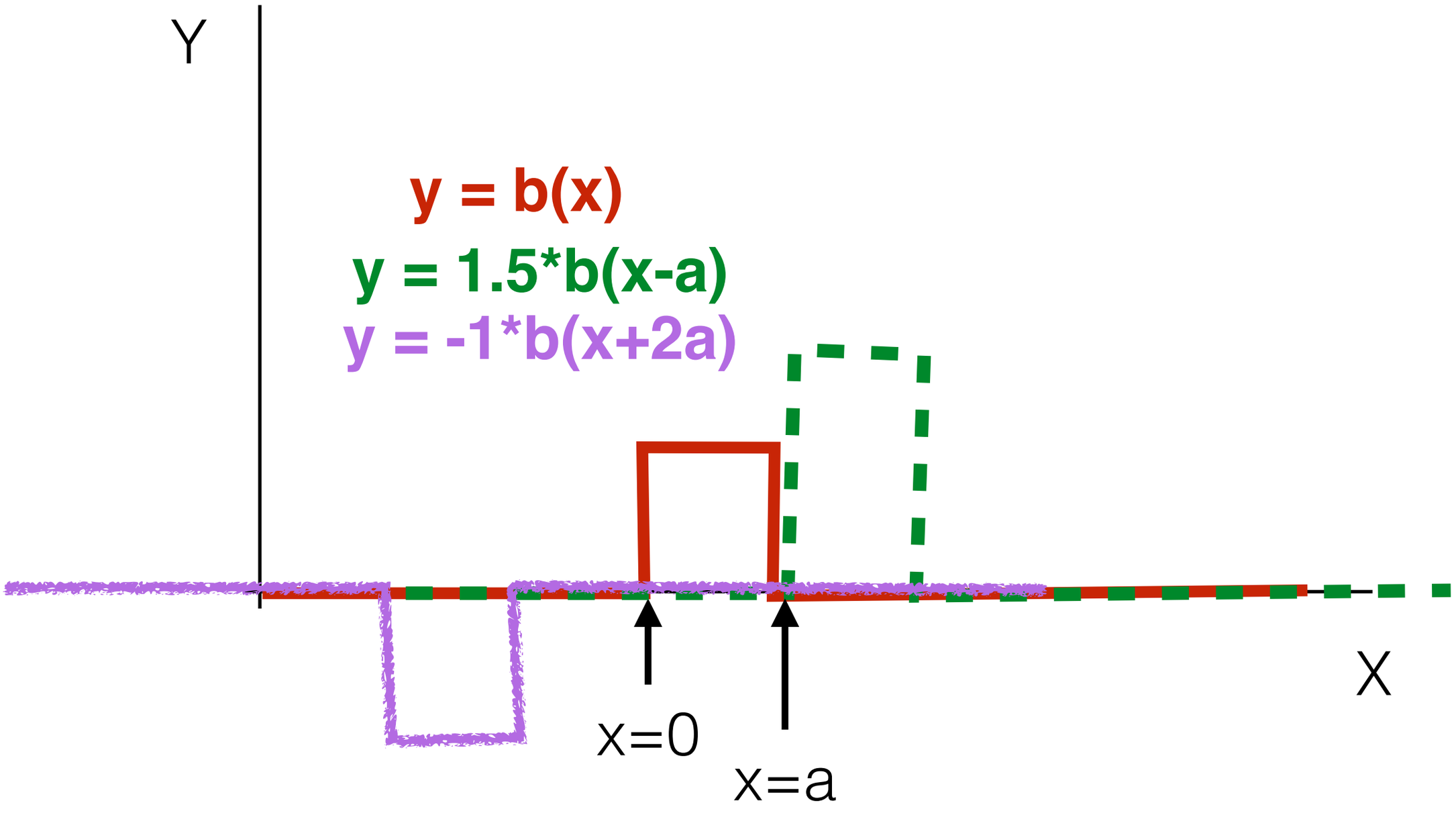
# Building a function



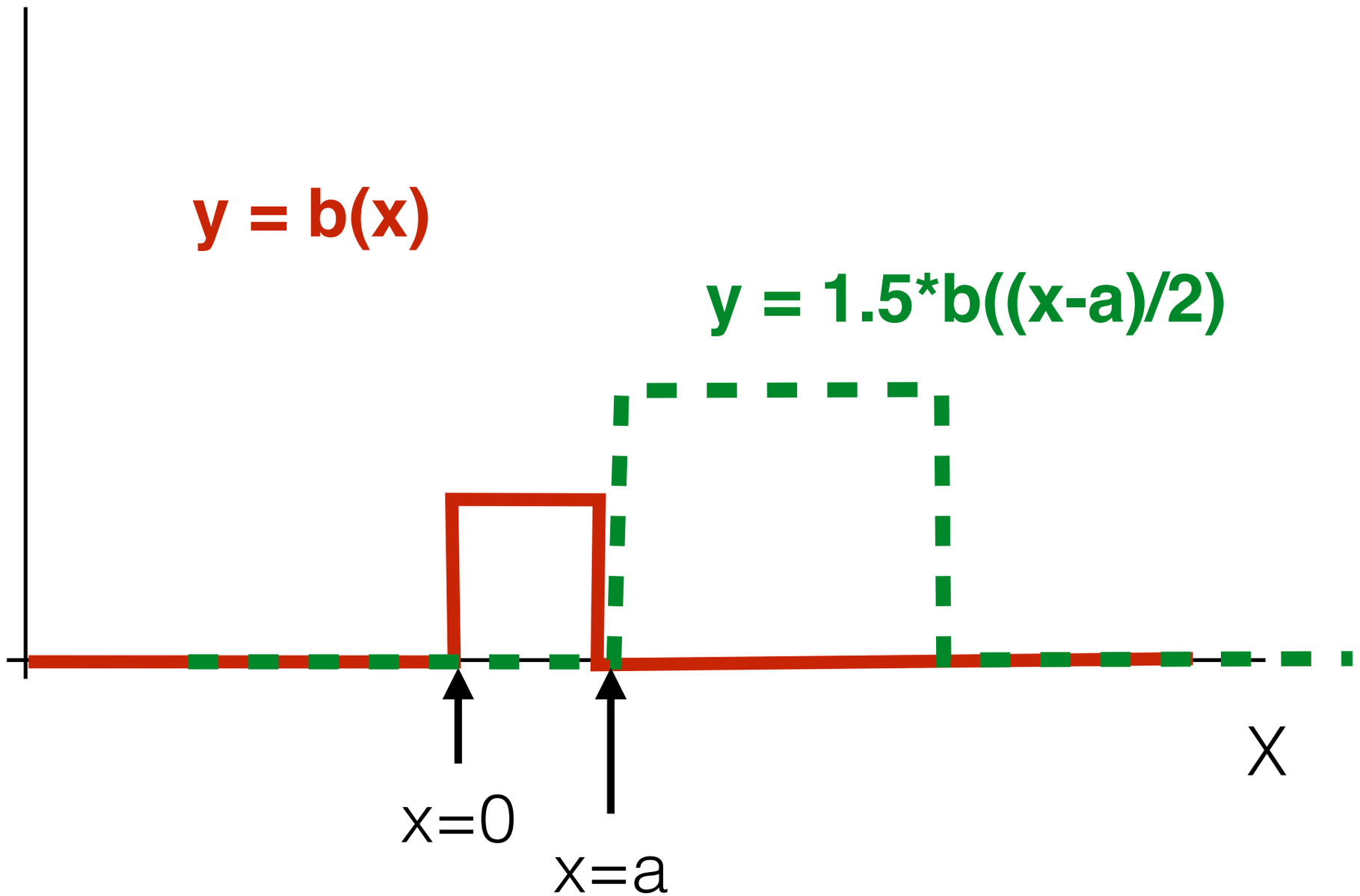
# Building a function



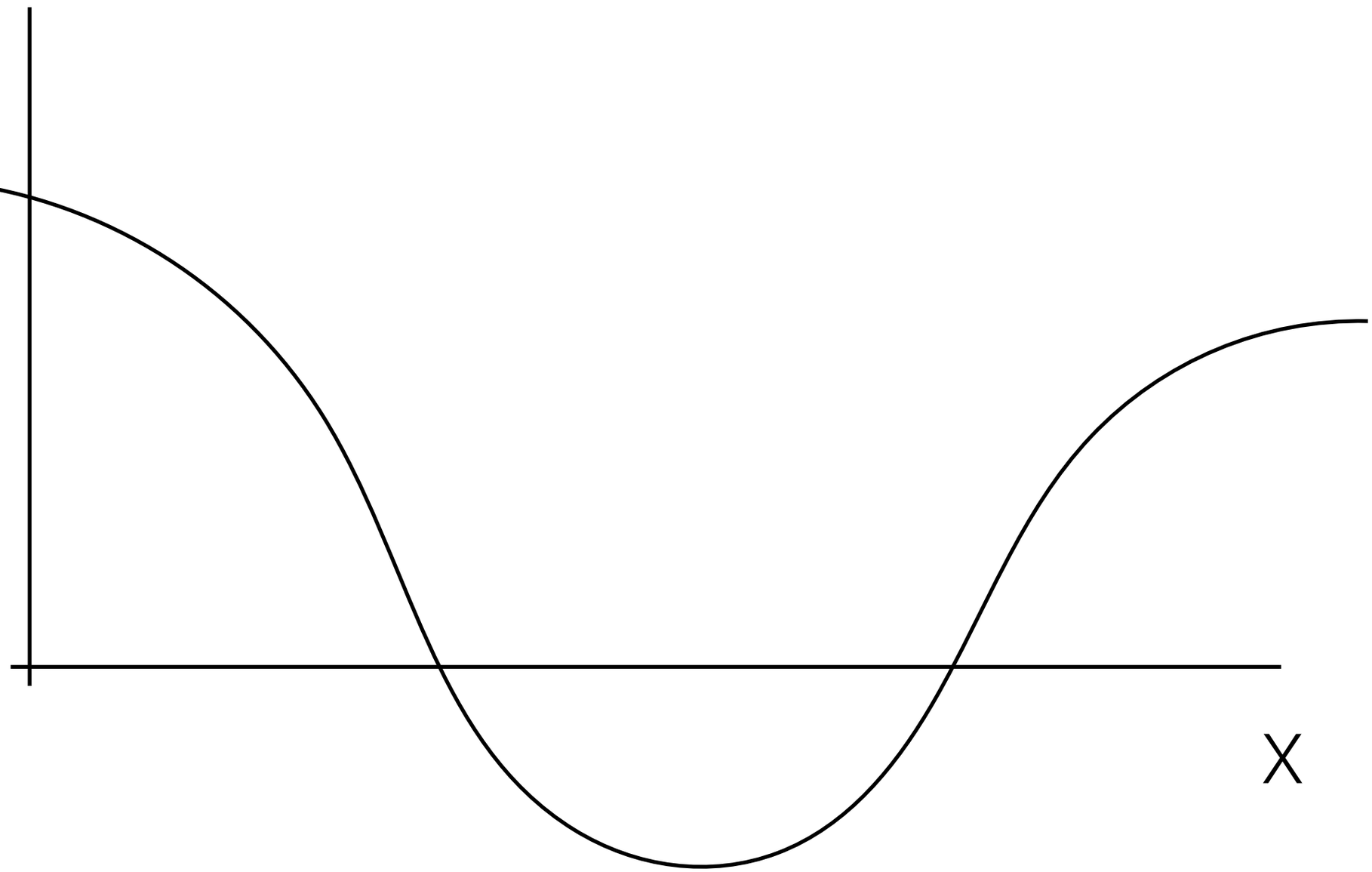
# Building a function



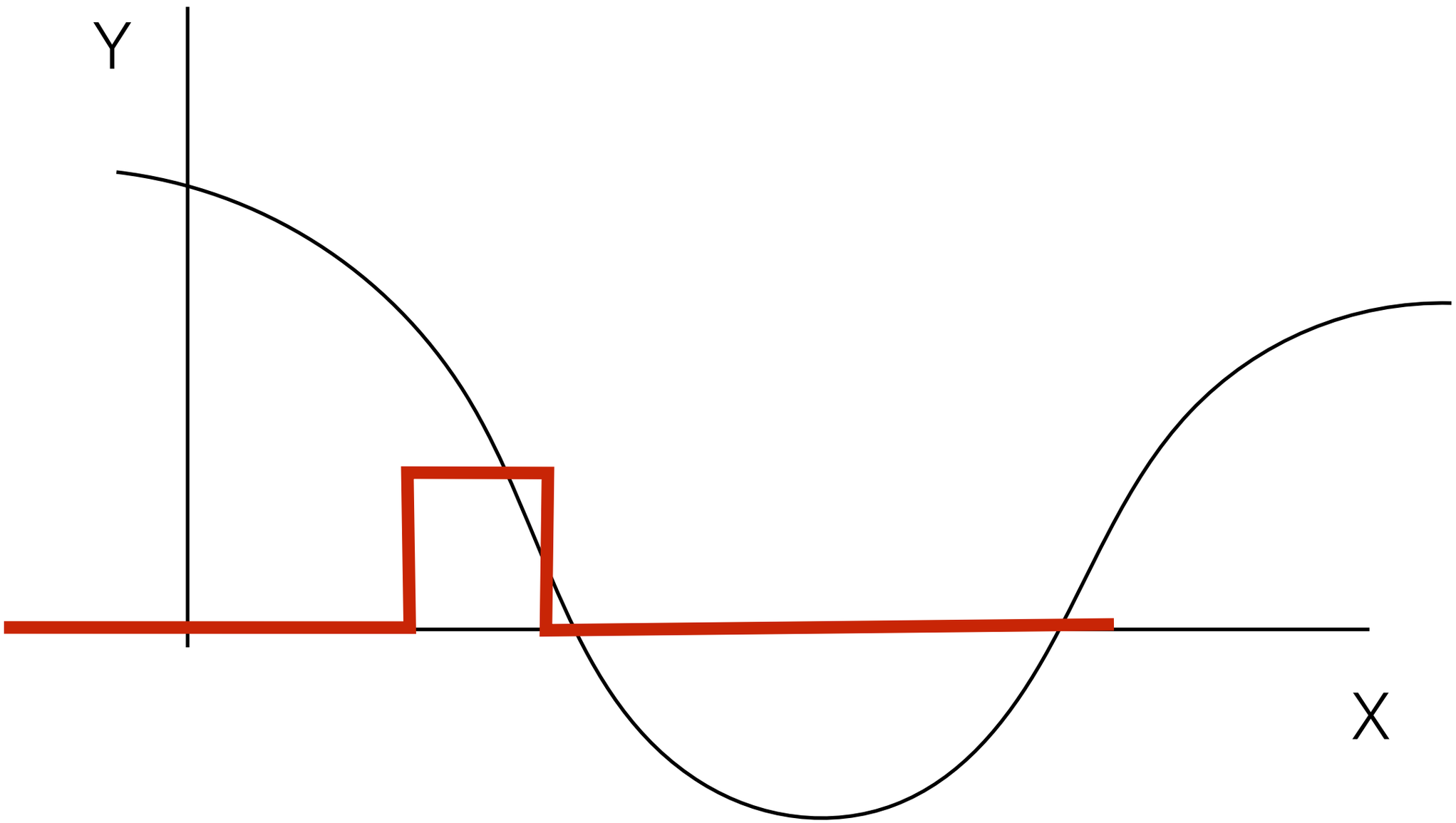
# Building a function



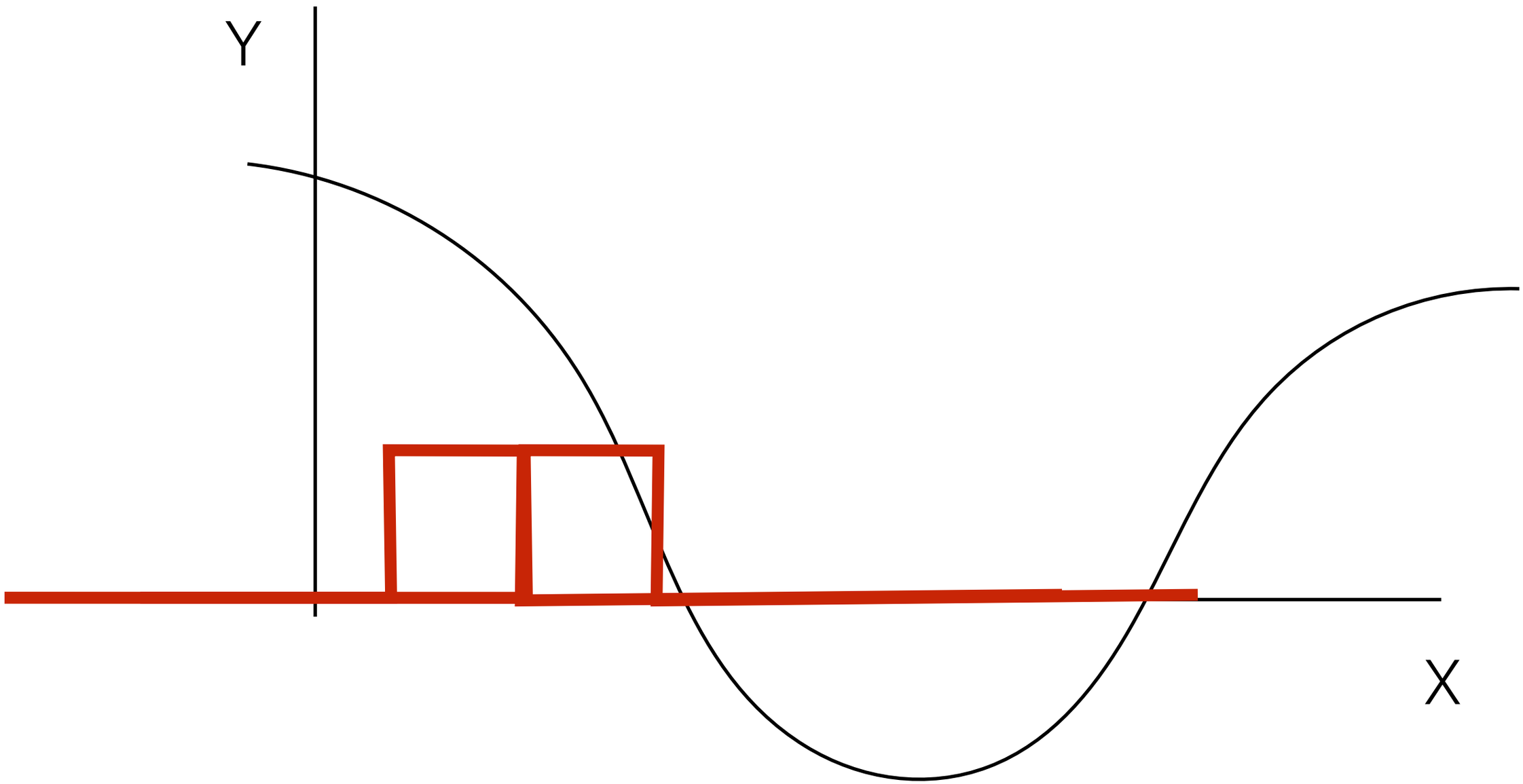
# How to build this function using b(x)?



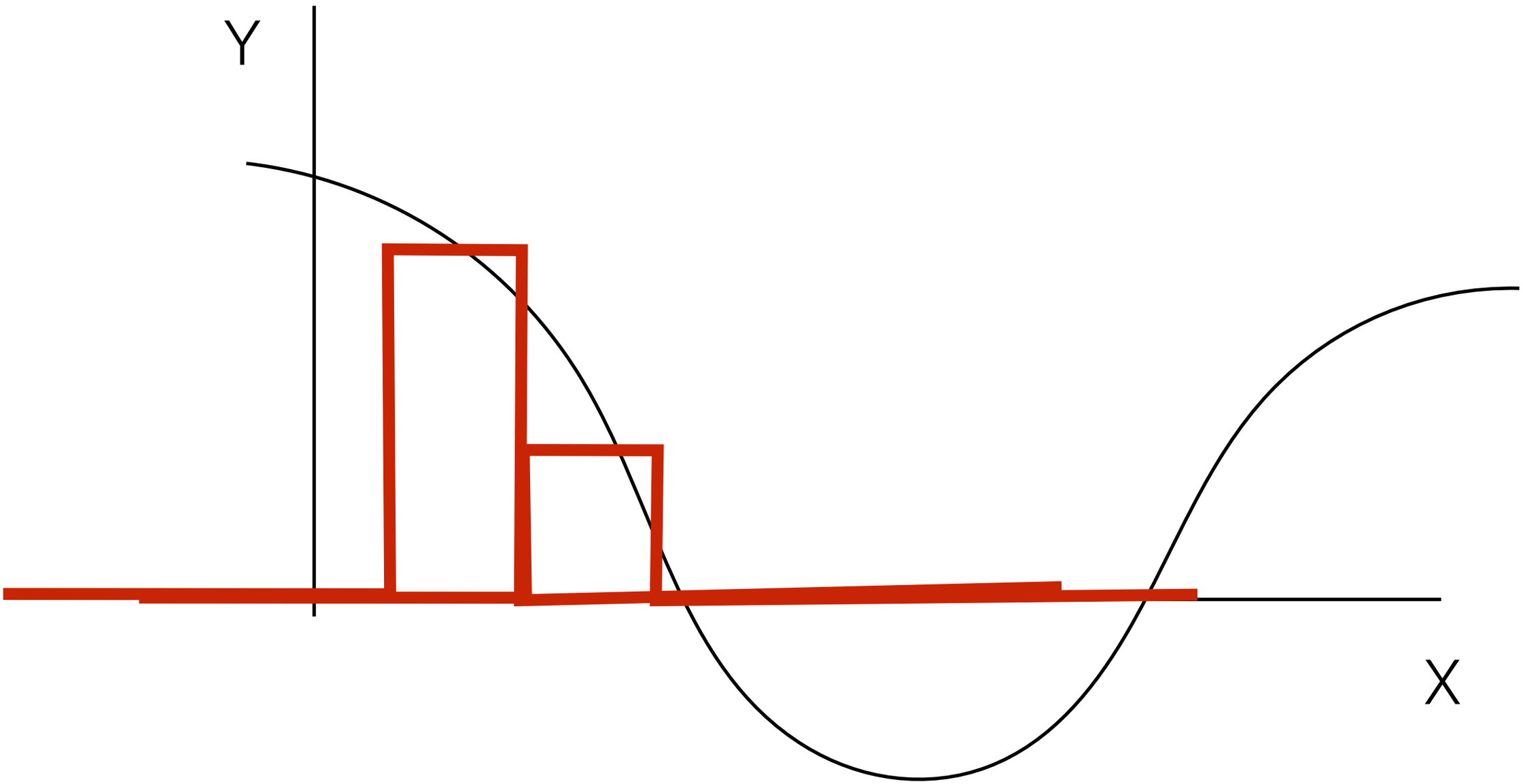
# How to build this function using b(x)?



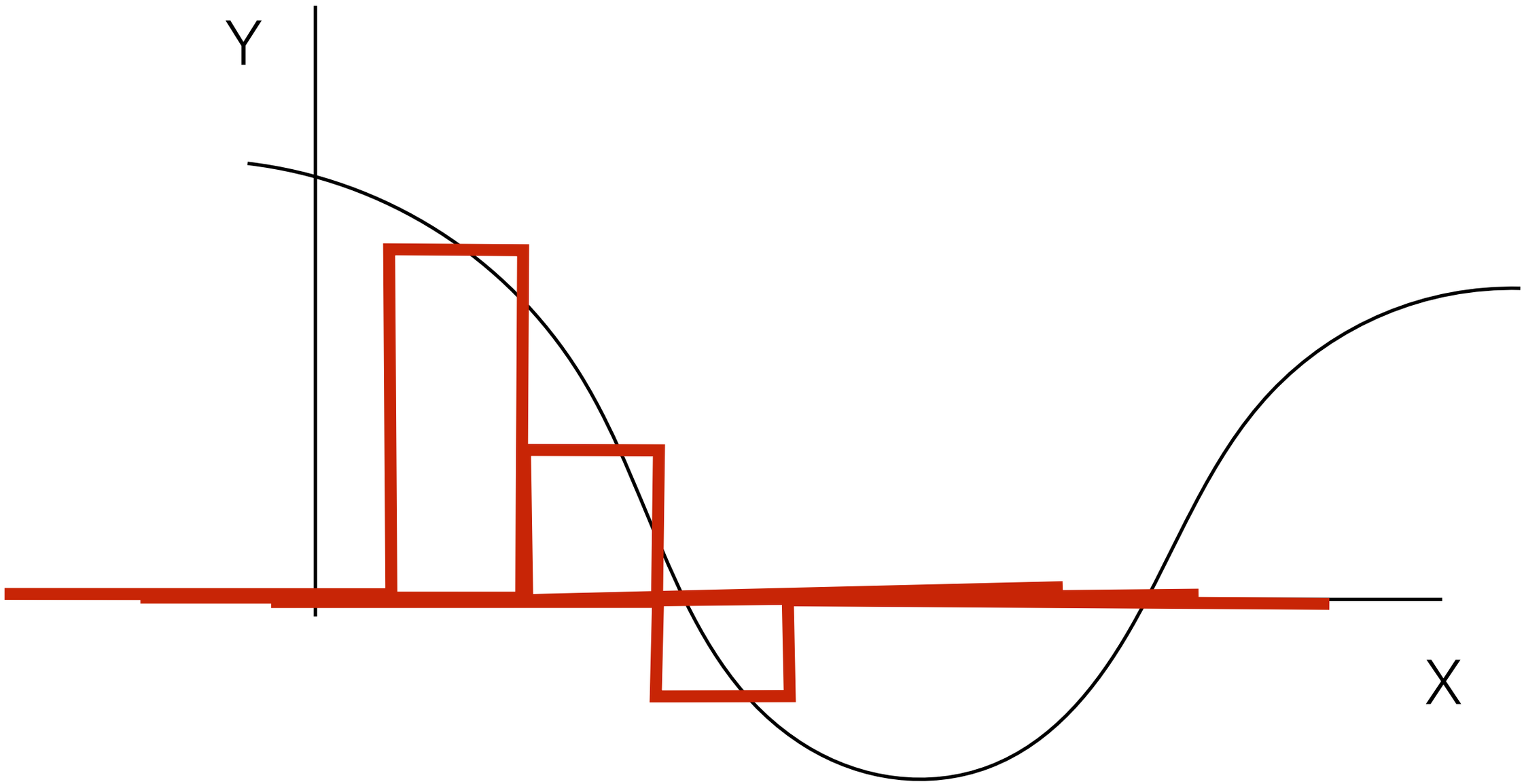
# How to build this function using b(x)?



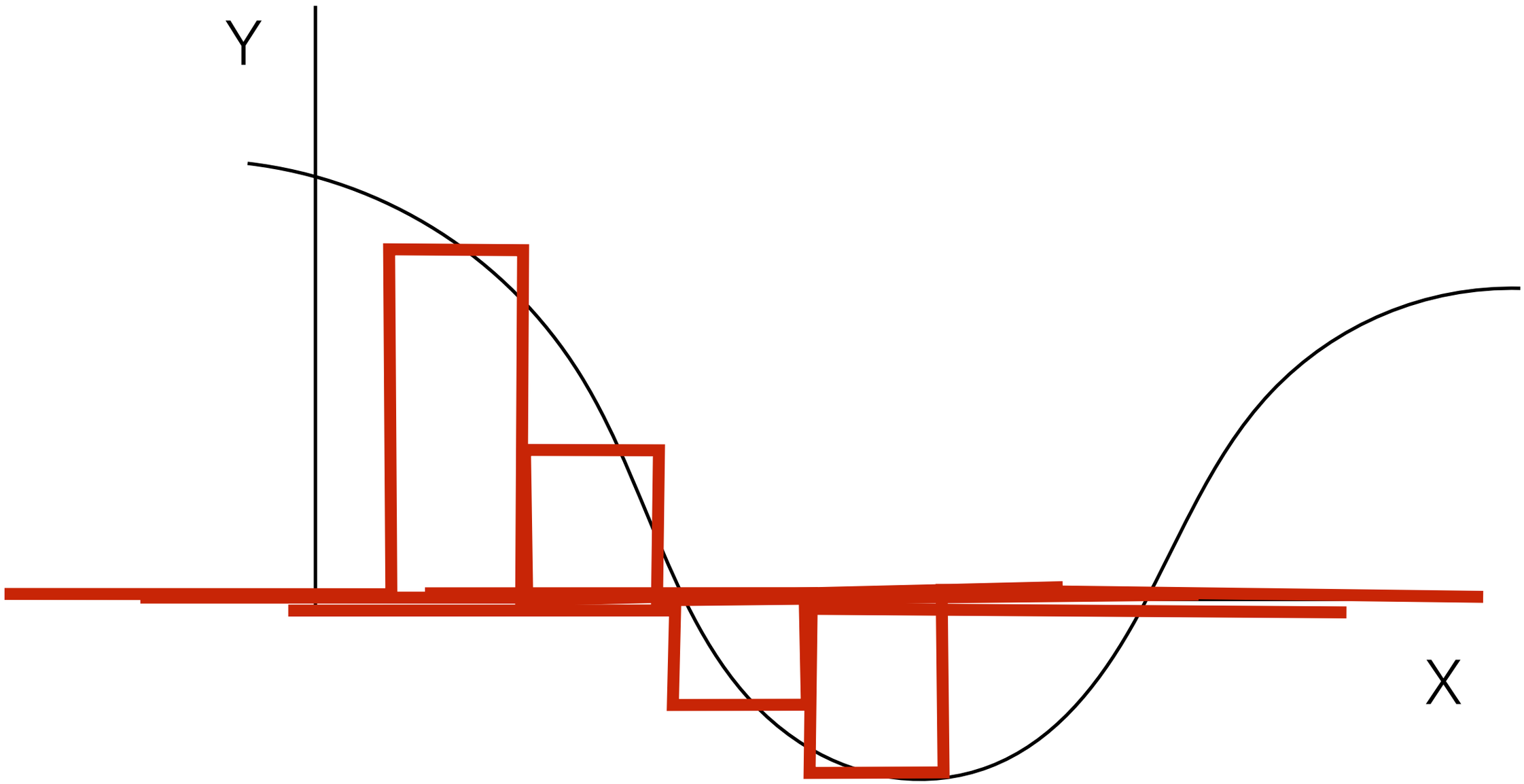
# How to build this function using b(x)?



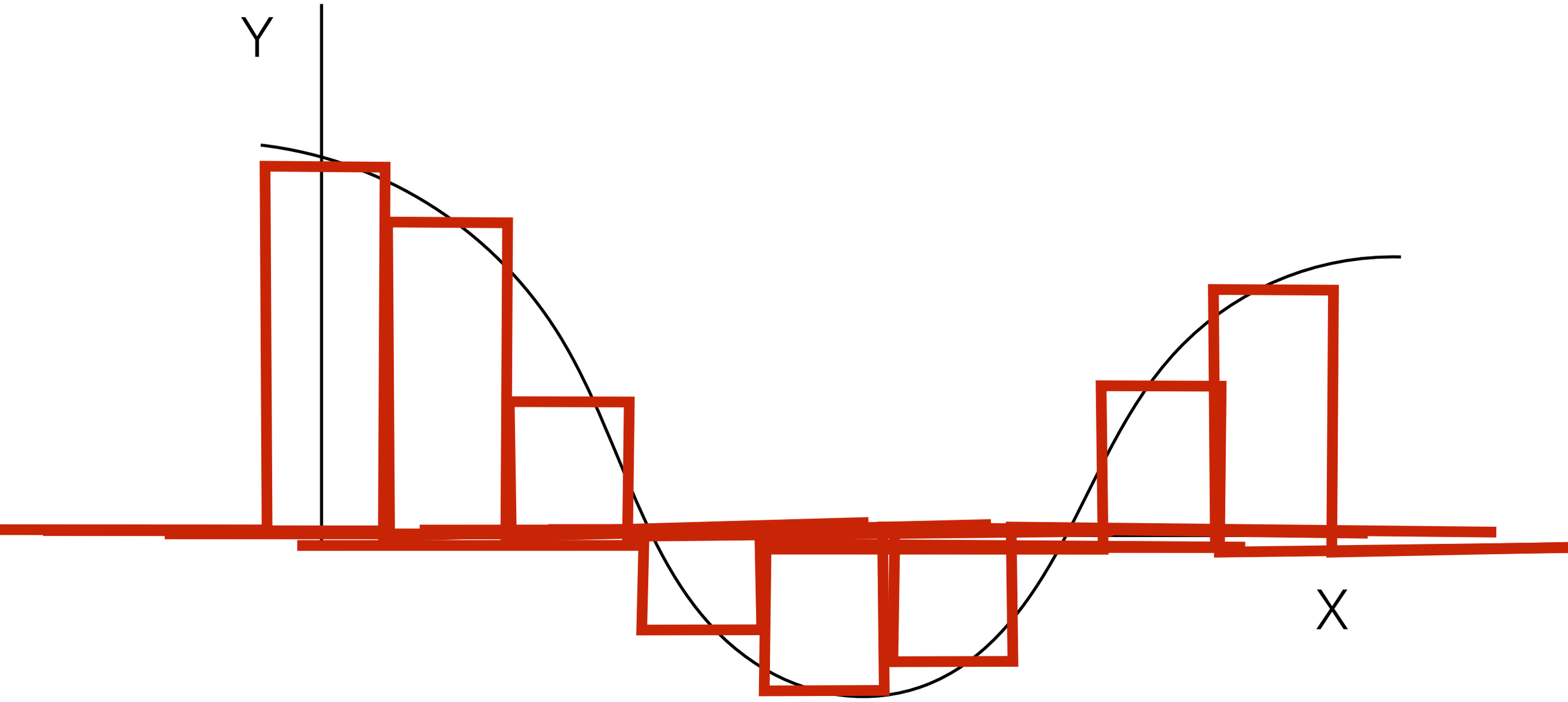
# How to build this function using b(x)?



# How to build this function using b(x)?

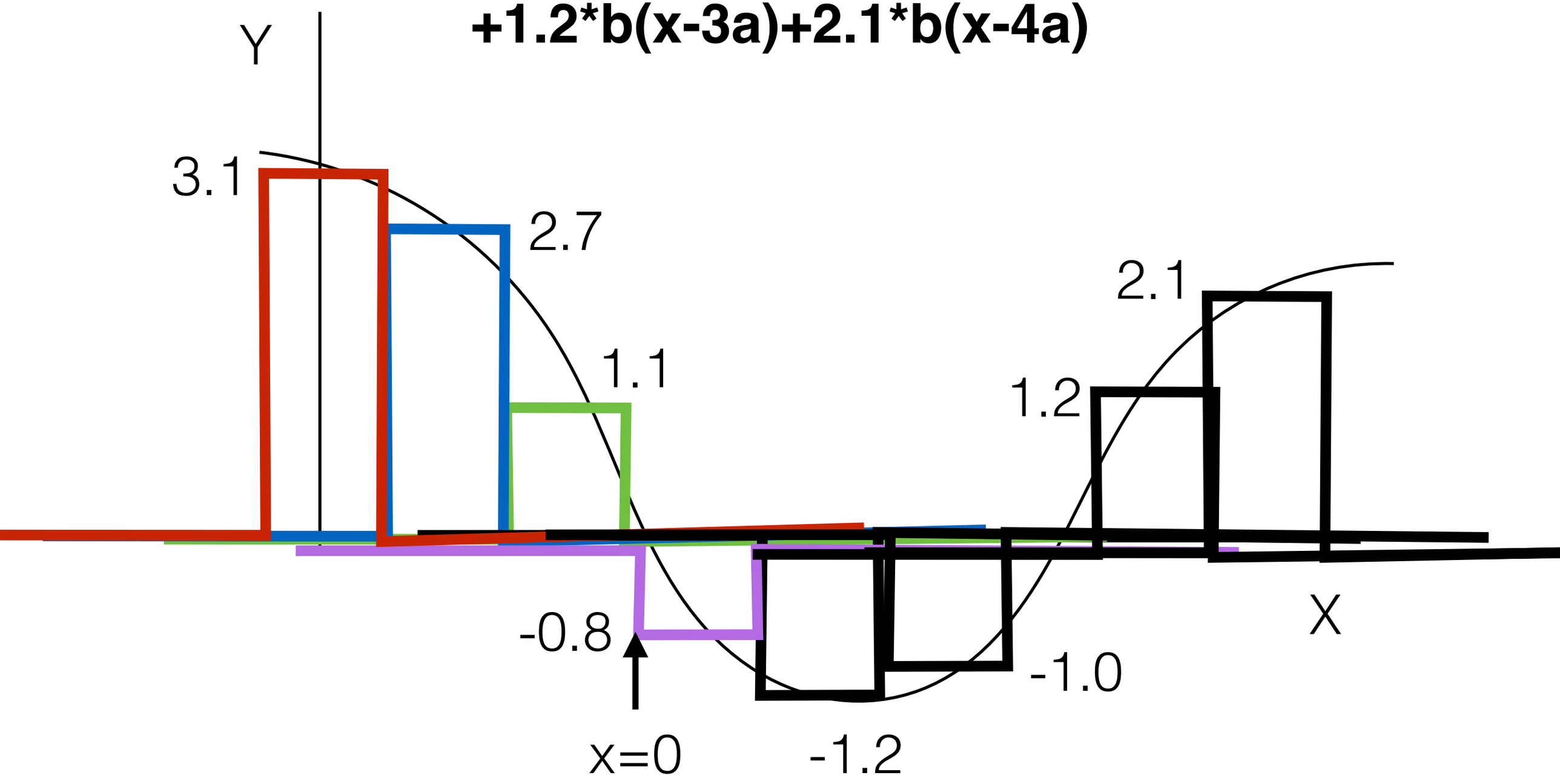


# How to build this function using b(x)?

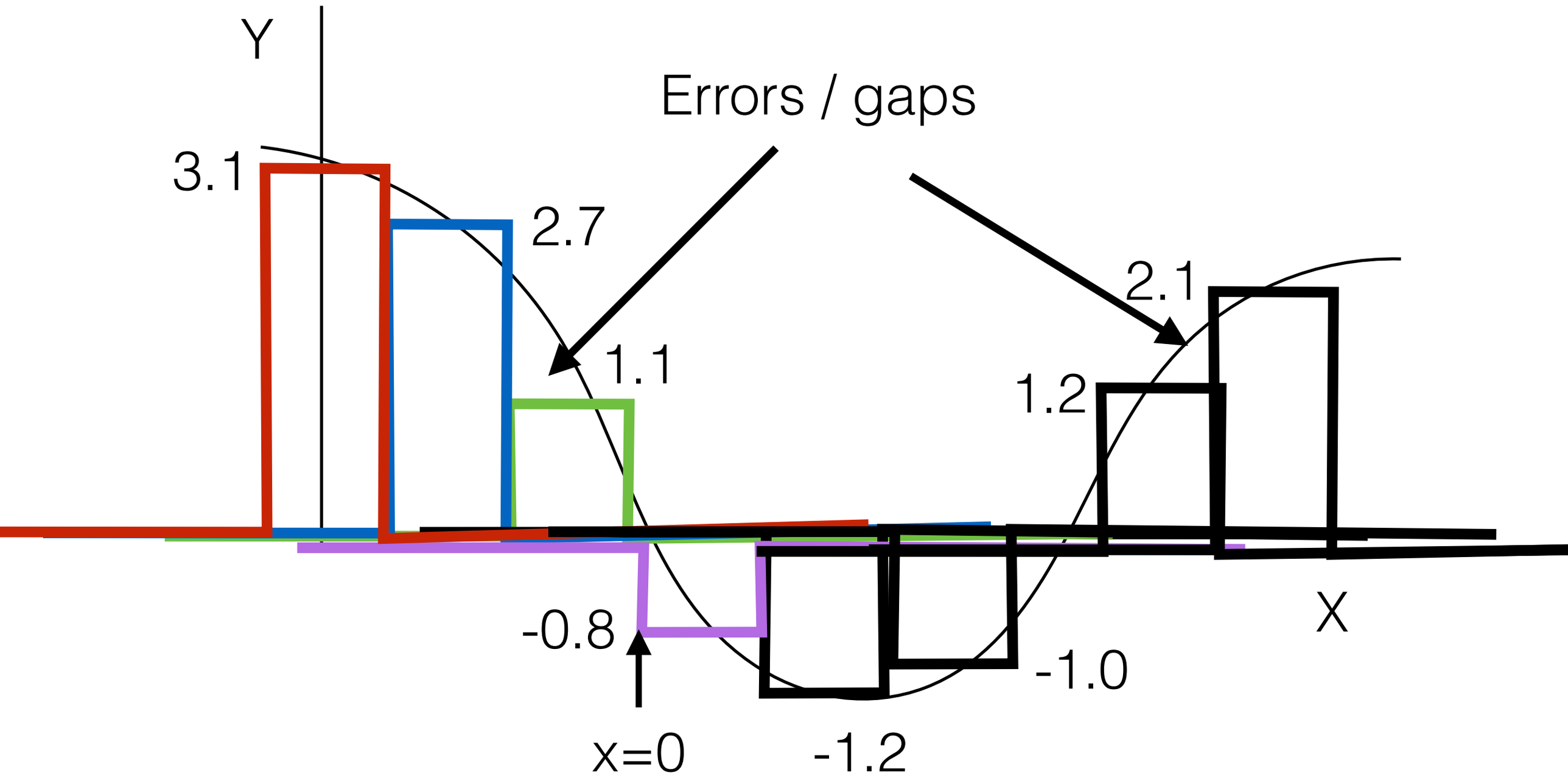


How to build this function using b(x)?

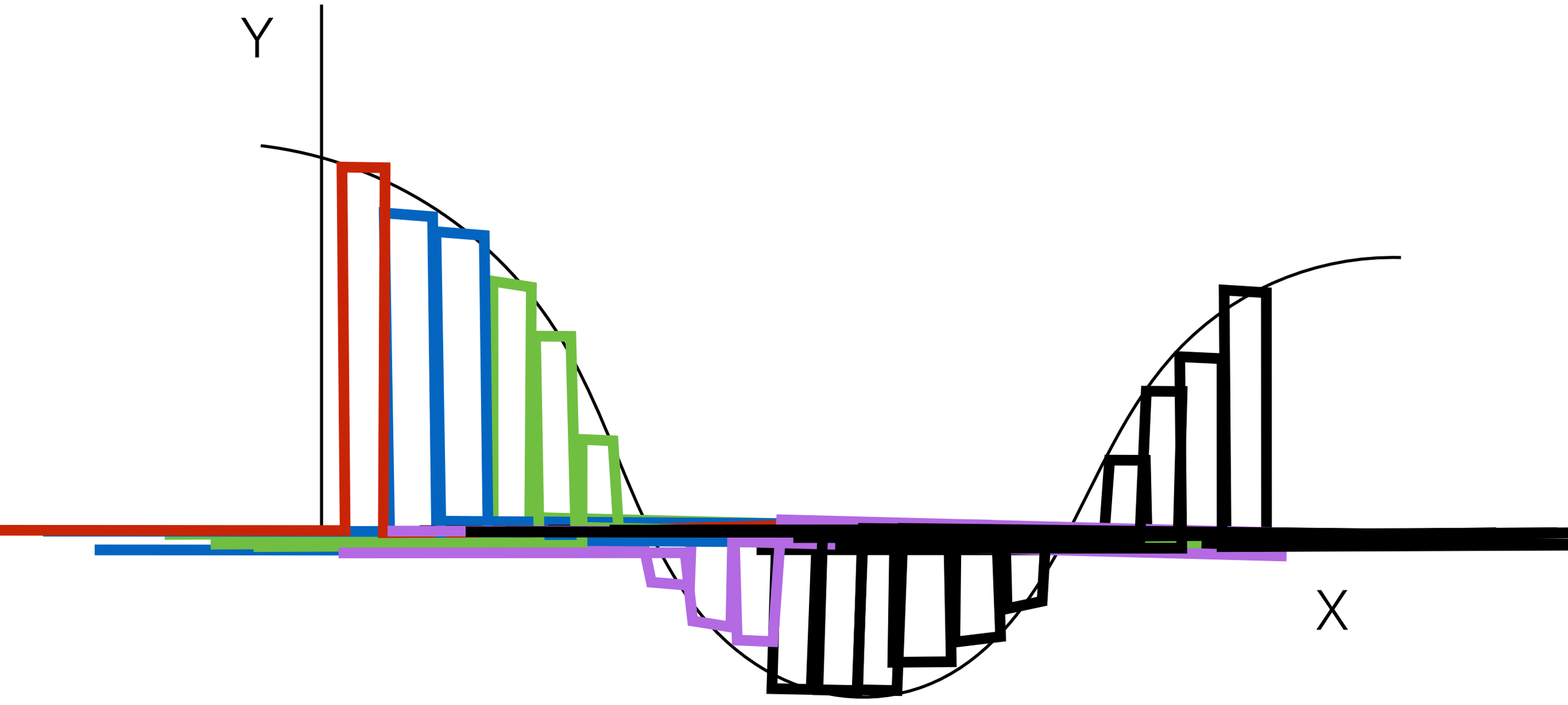
$$y = 3.1*b(x+3a) + 2.7*b(x+2a) + 1.1*b(x+a) - 0.8*b(x) - 1.2*b(x-a) - 1.0*b(x-2a) + 1.2*b(x-3a) + 2.1*b(x-4a)$$



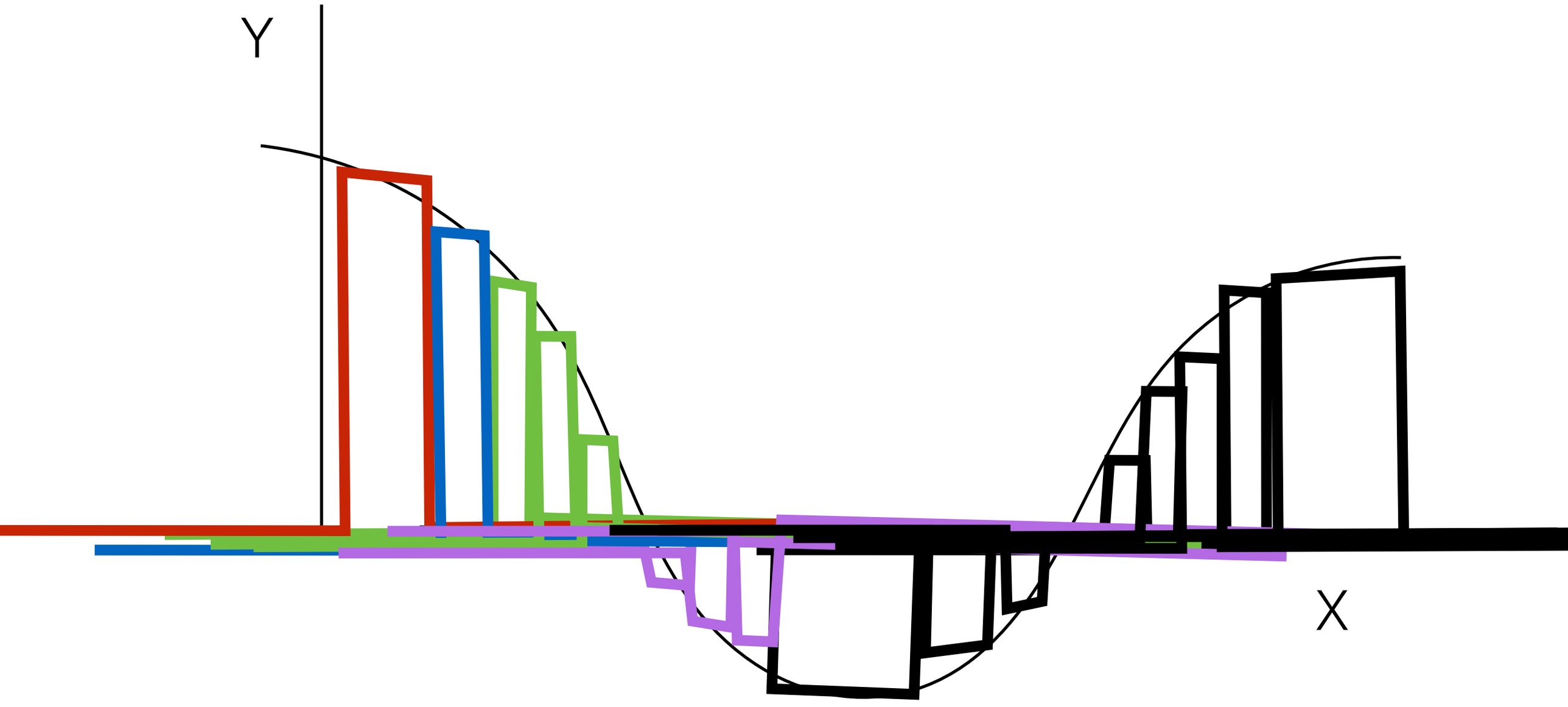
# How to build this function using b(x)?



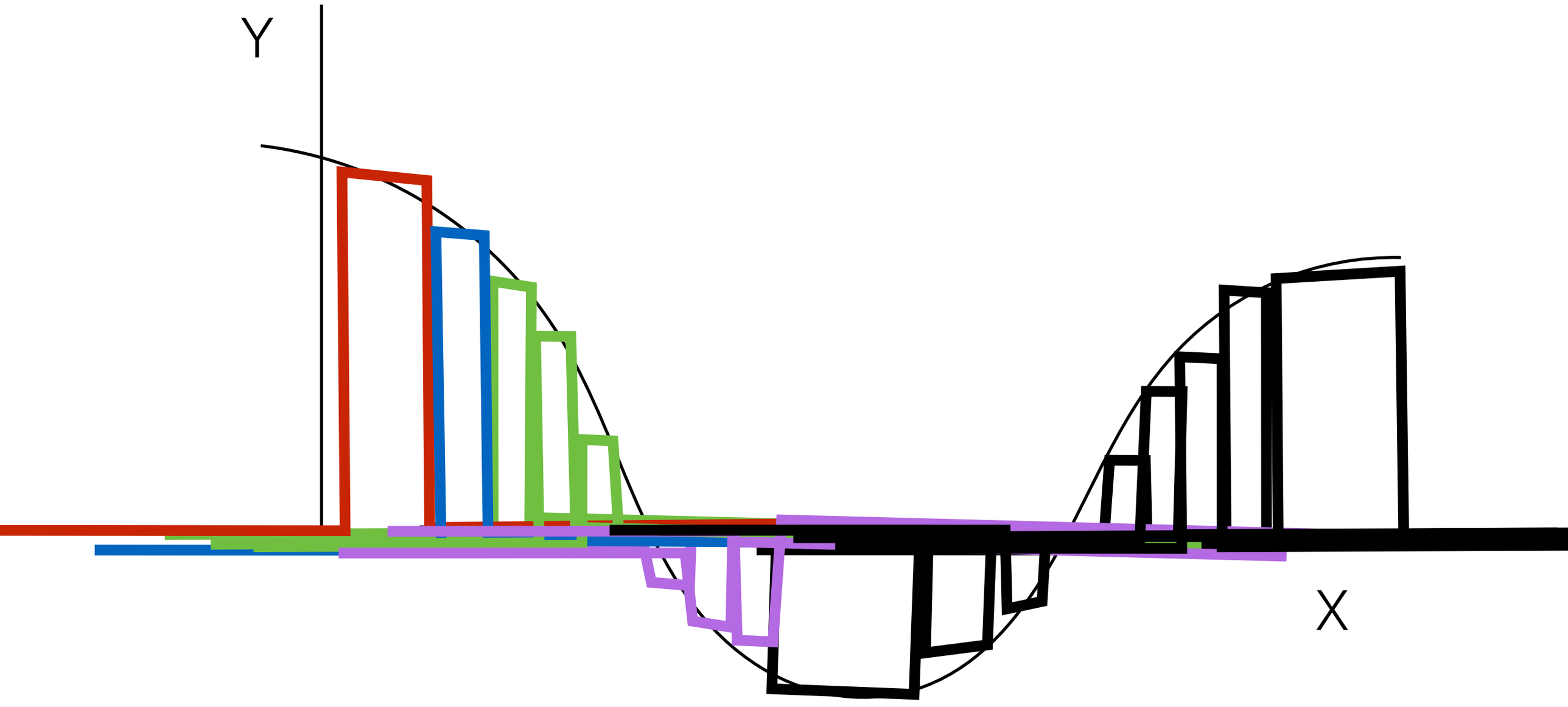
# Smaller “a” (width)



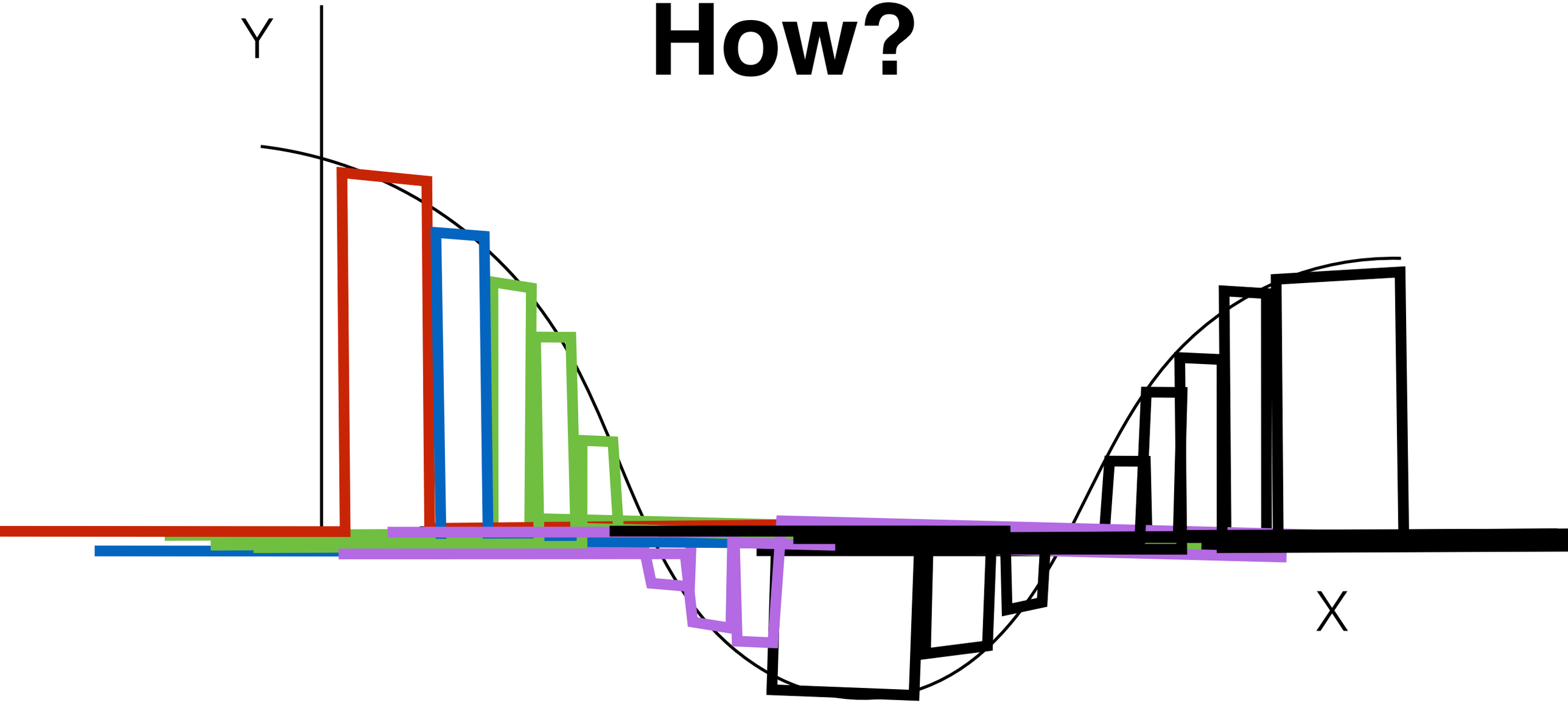
# Bumps with different widths



# Relationship with deep learning — Neural Nets build functions by adding functions



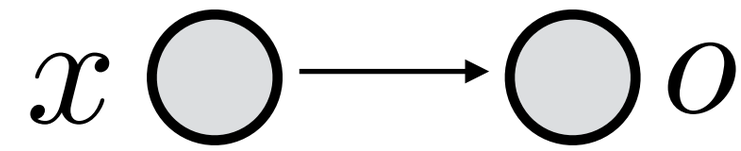
Relationship with deep learning — Neural Nets build functions by adding functions

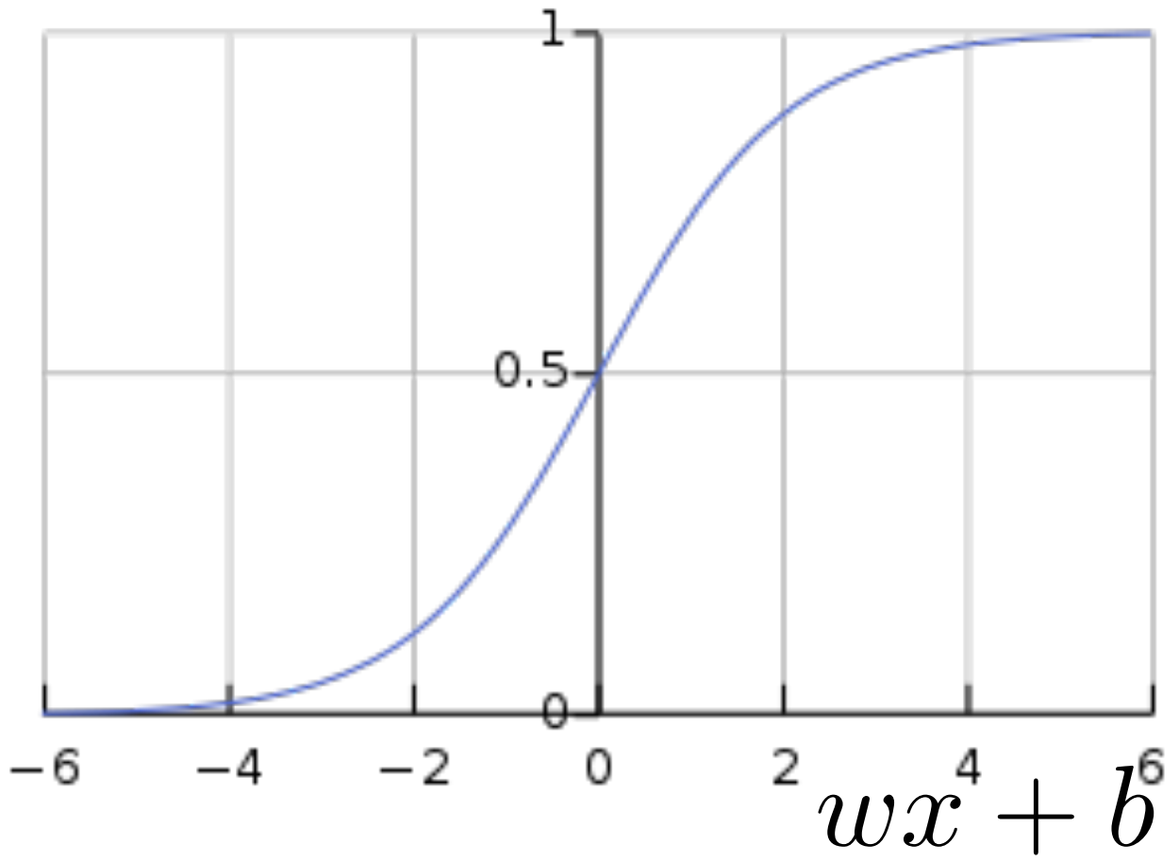



**How?**

# Simplest perceptron - sigmoid activation function

$$o = \frac{1}{1 + \exp(-wx - b)}$$

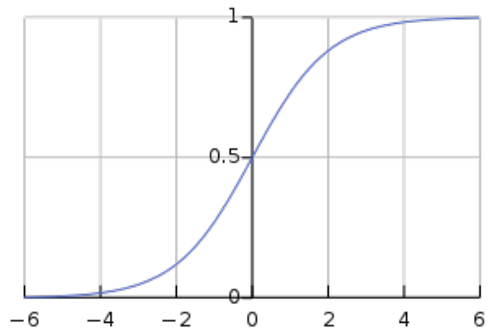


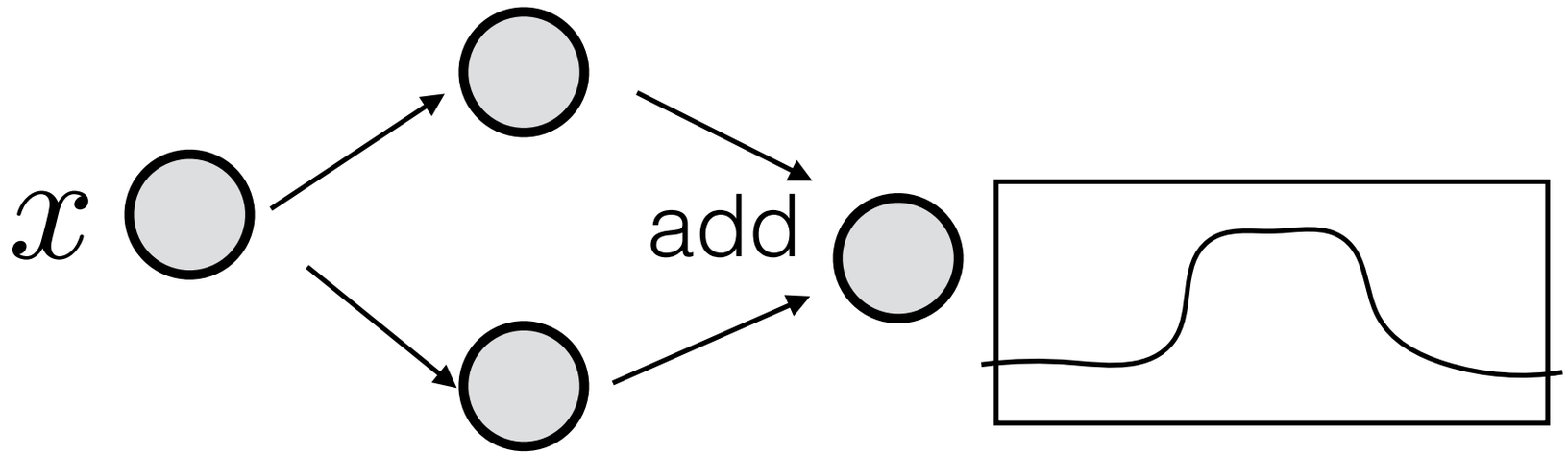



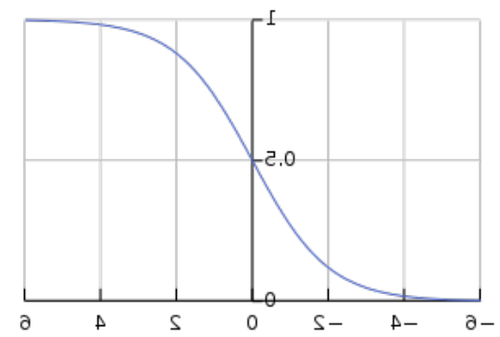

For more activation functions, check out

https://en.wikipedia.org/wiki/Activation_function

# Neural network make a bump function

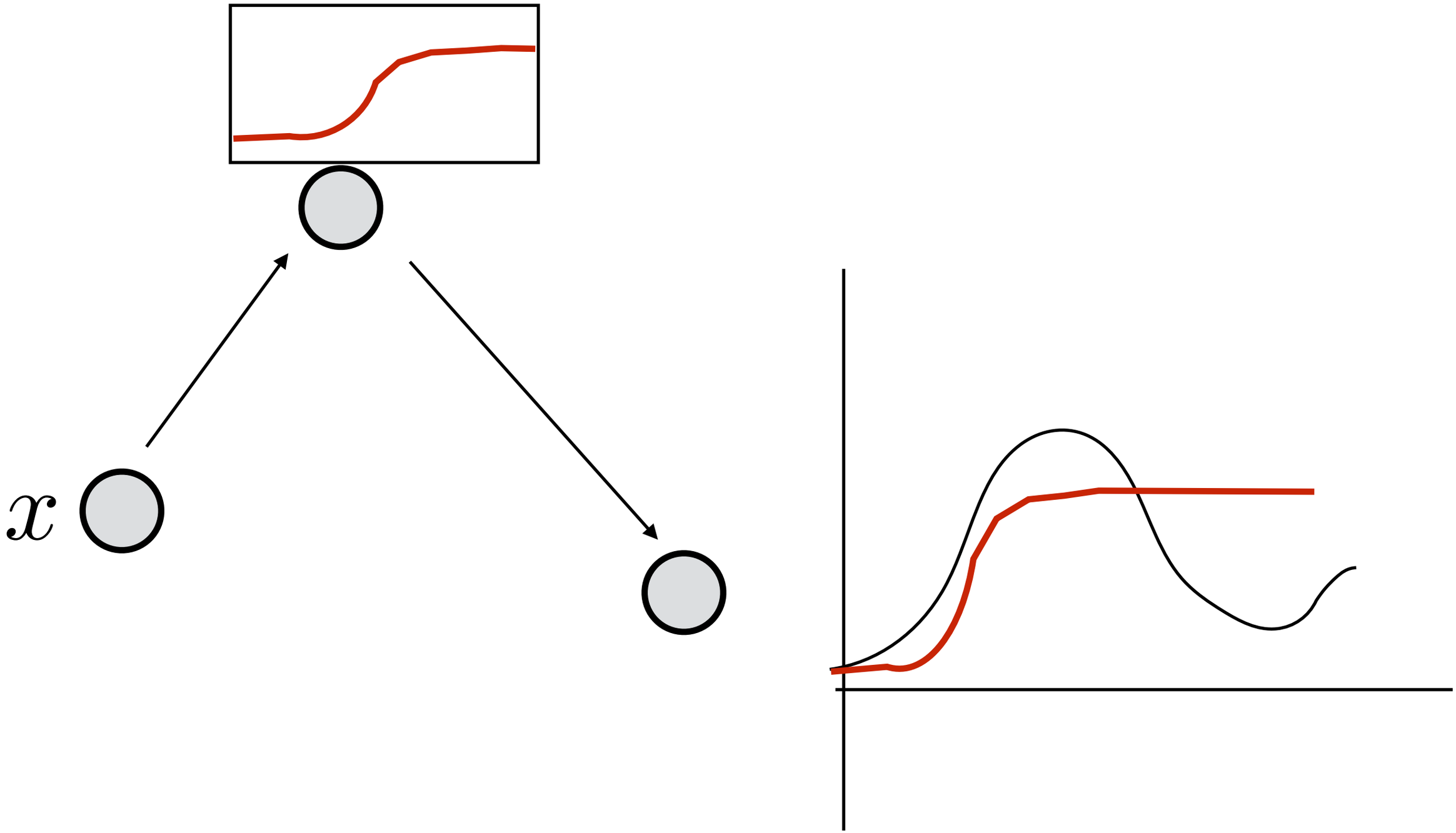


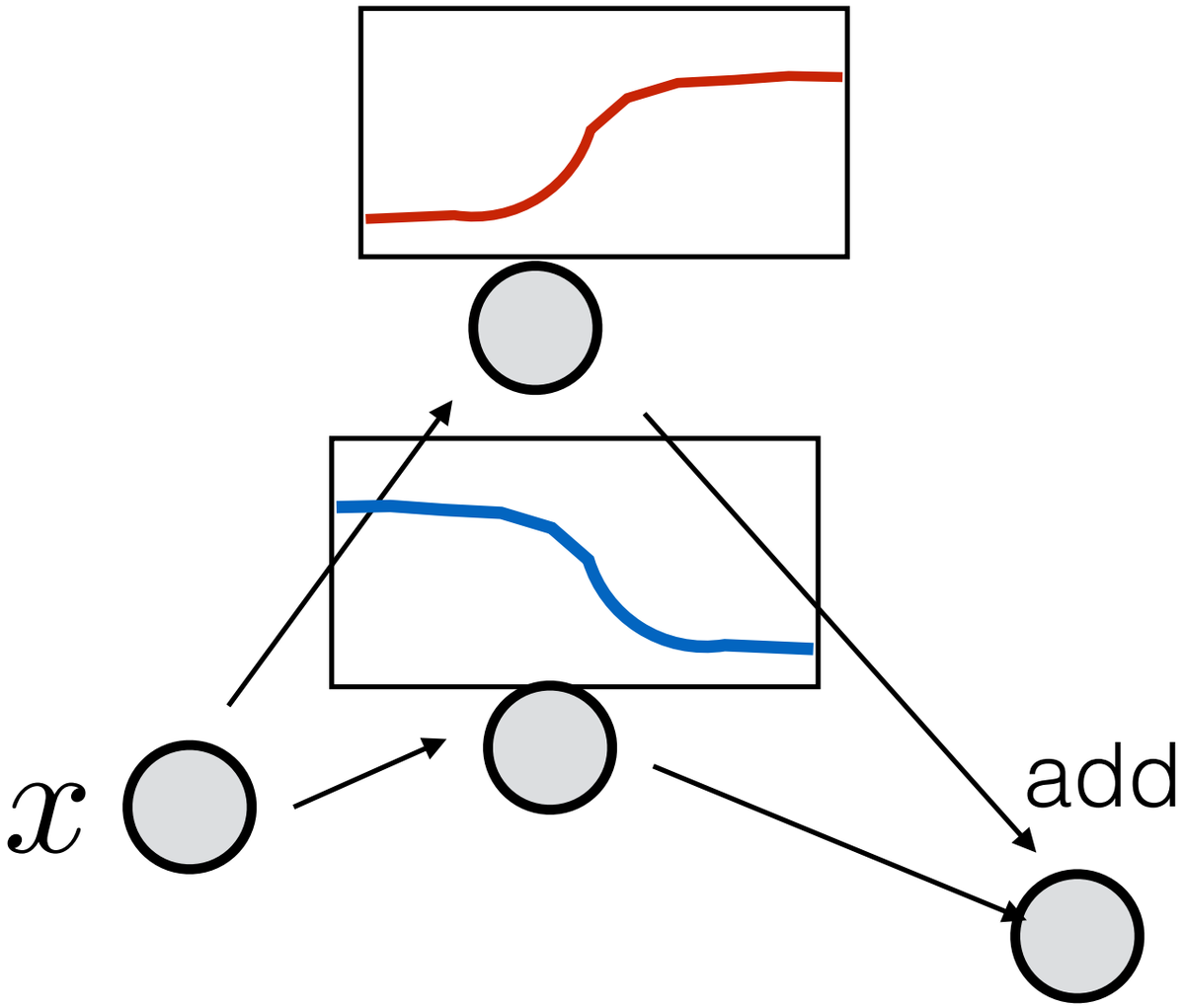



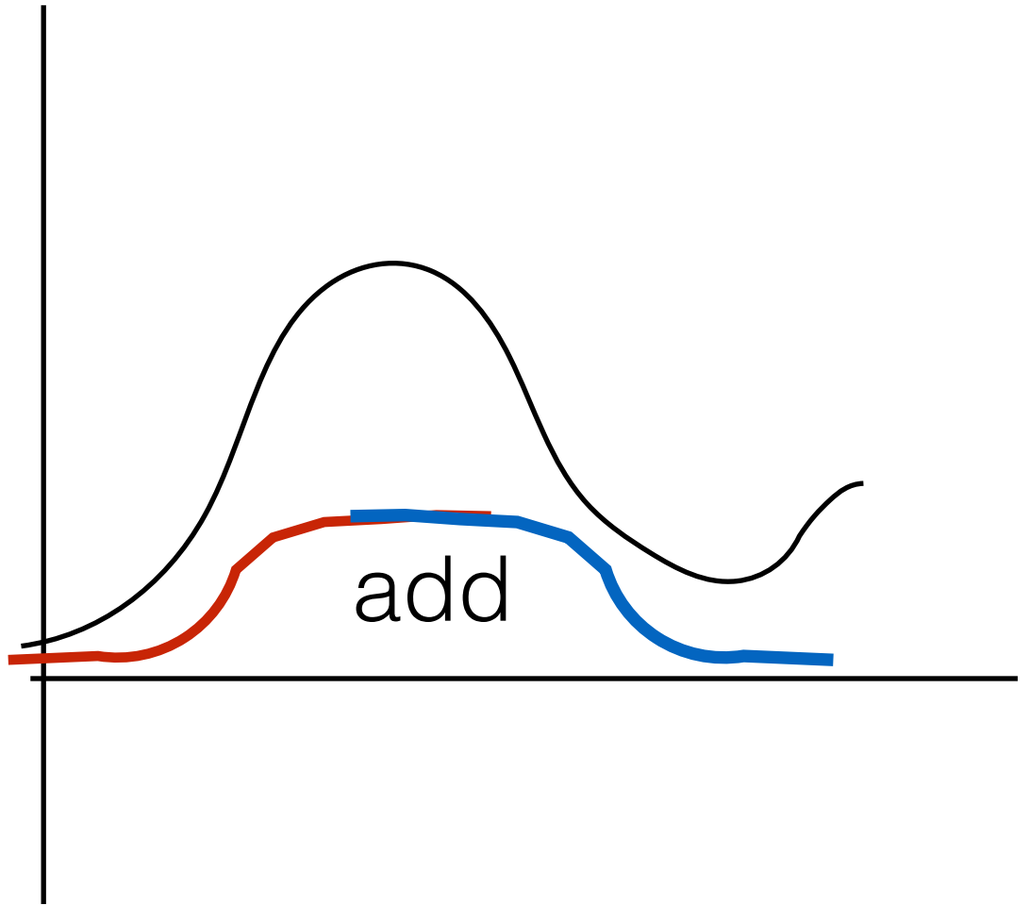



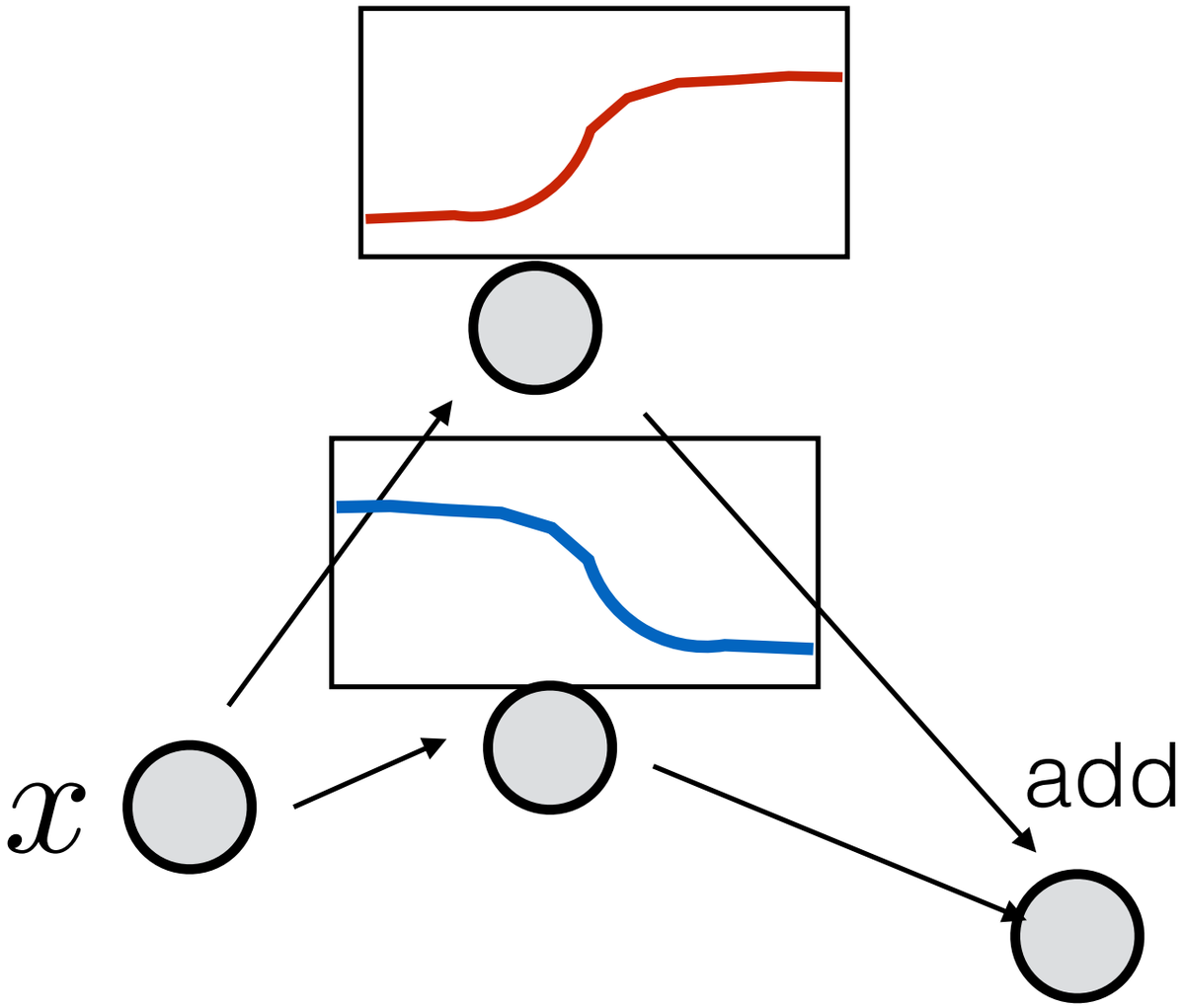
For more activation functions, check out

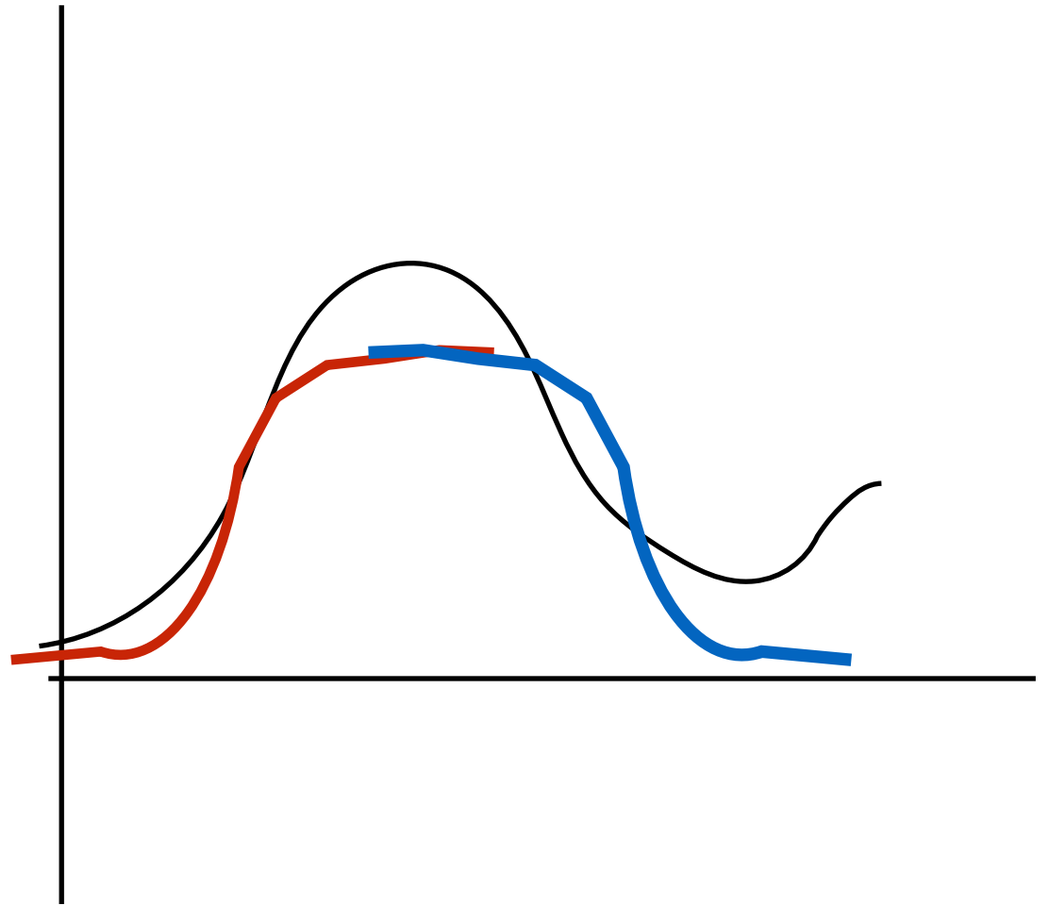
https://en.wikipedia.org/wiki/Activation_function

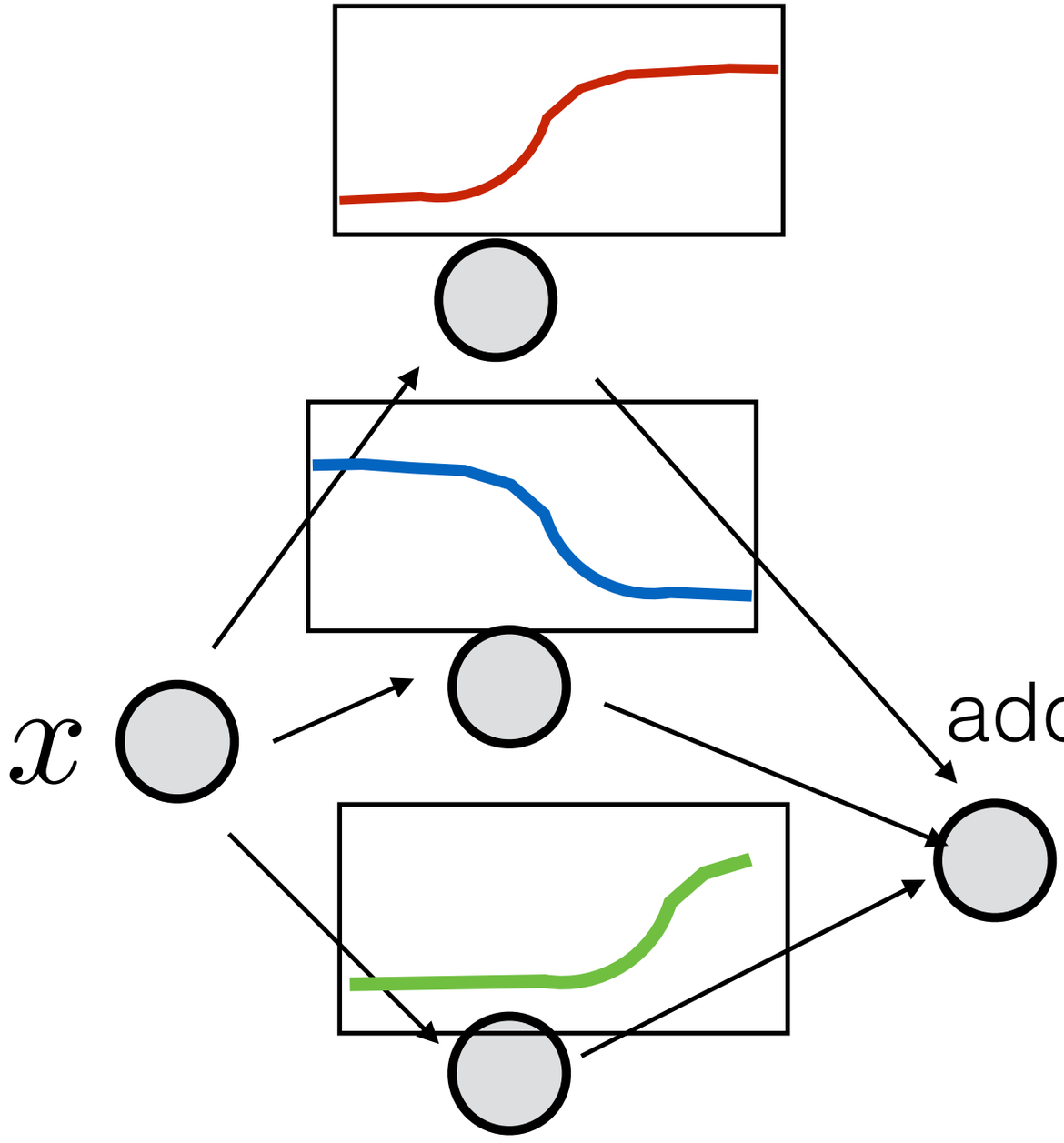

$x$

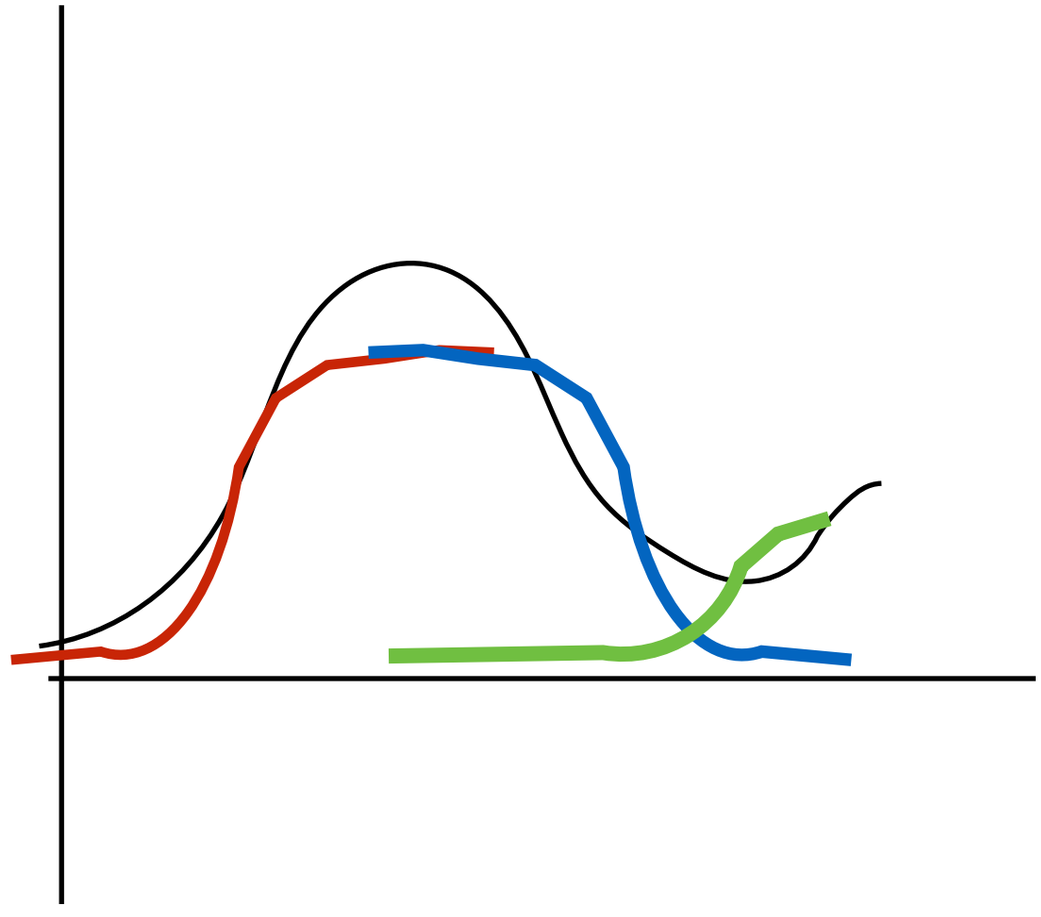

$x$
add

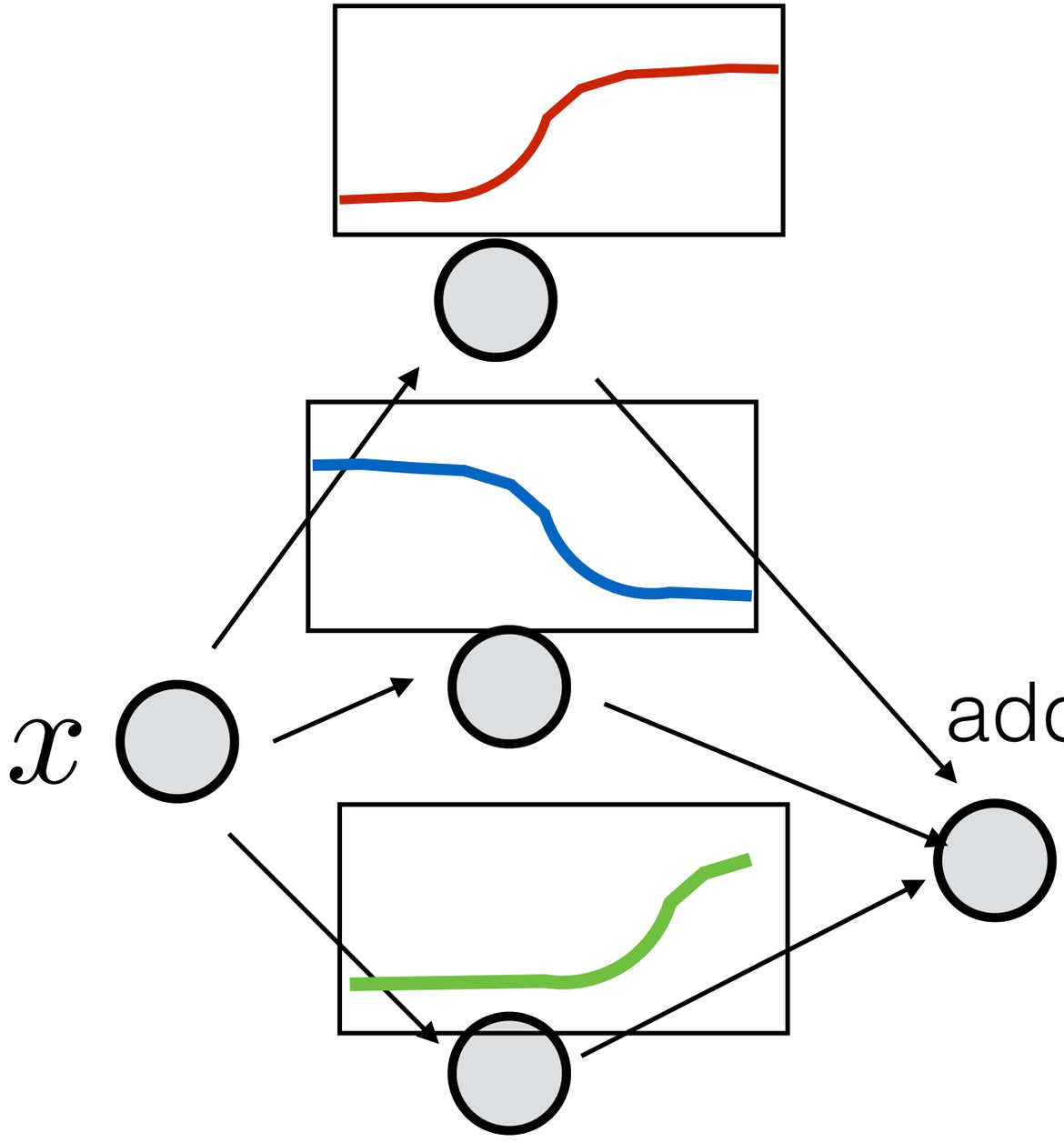

add

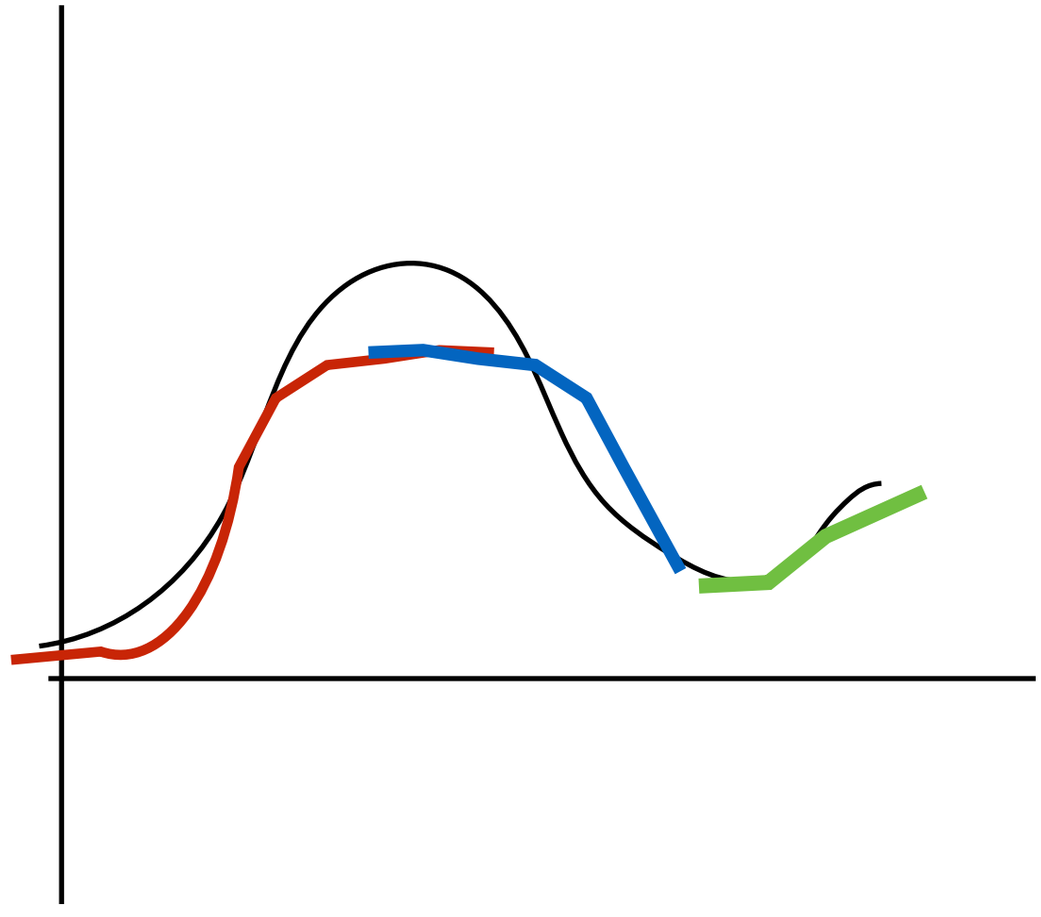

$x$
add

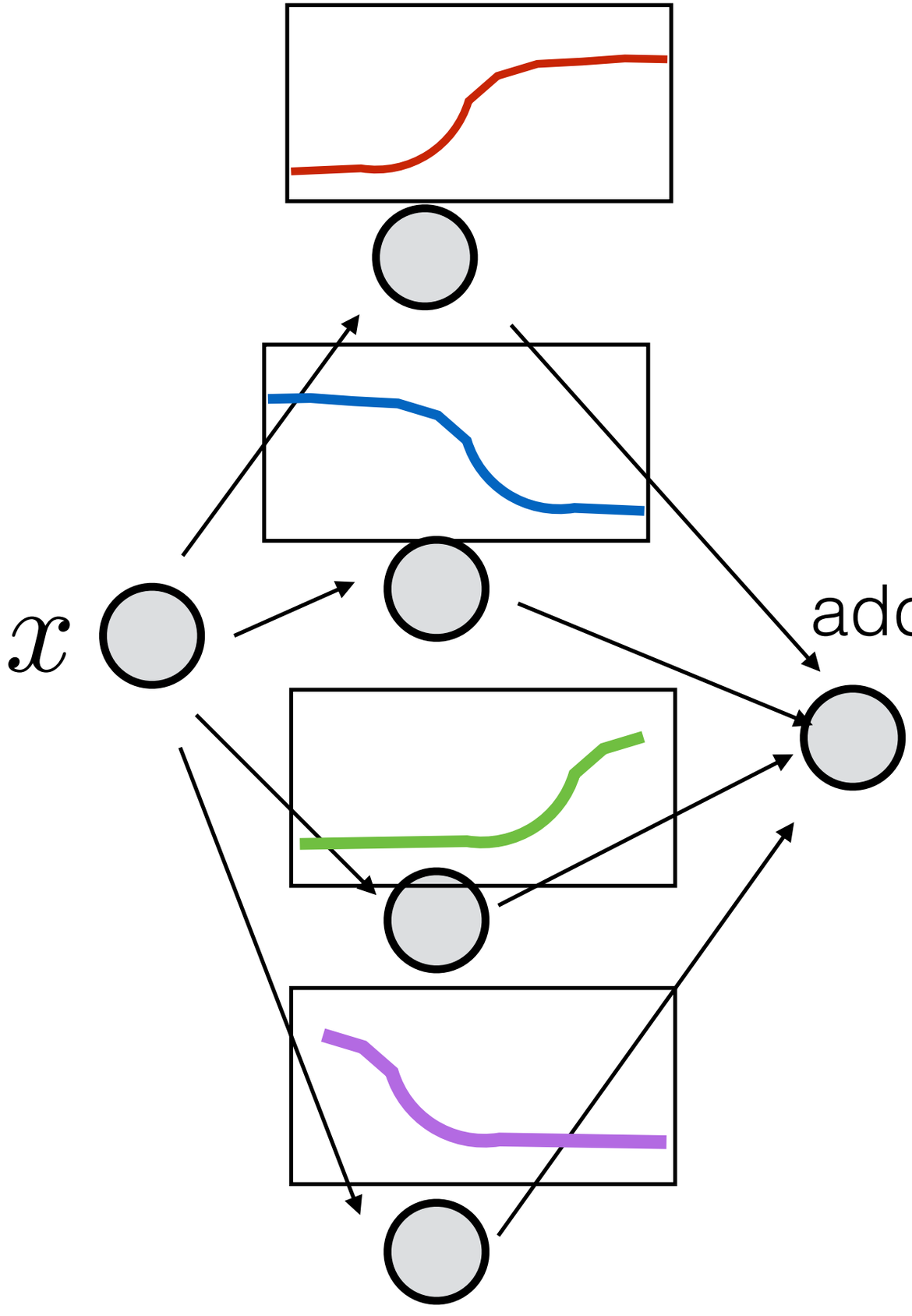


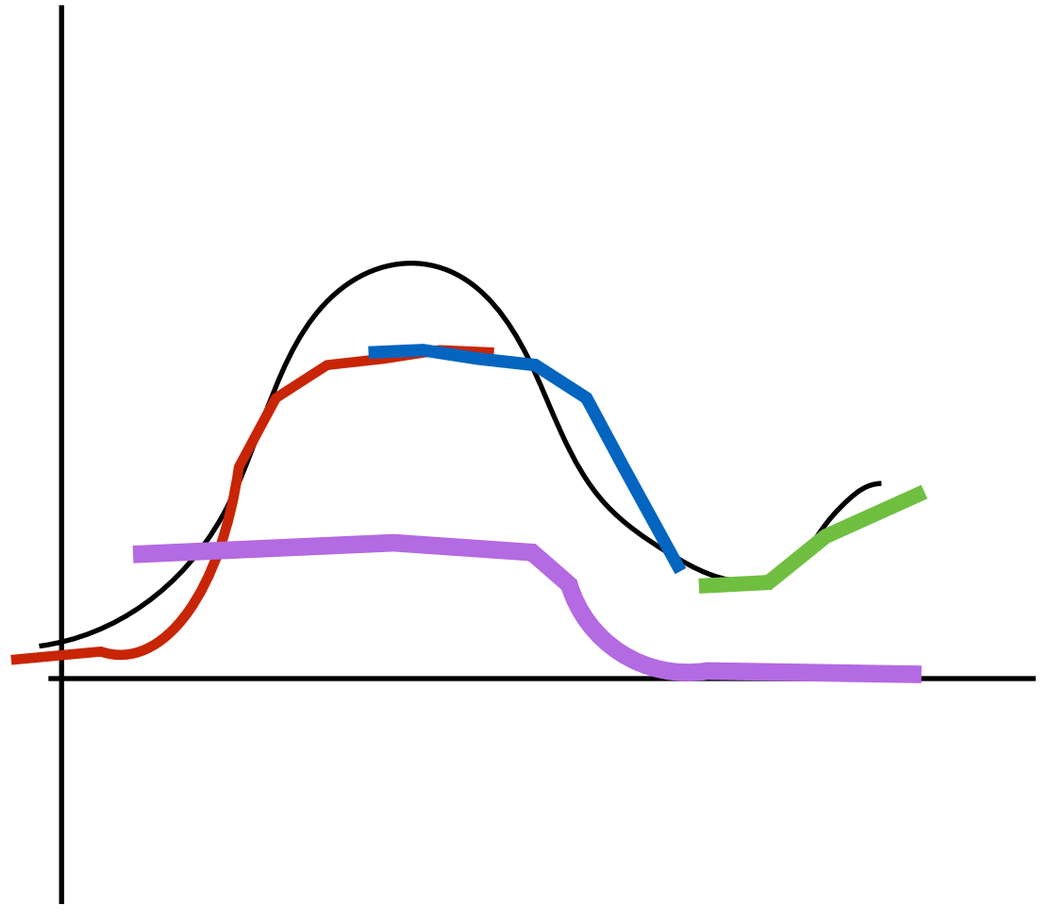

$x$
add

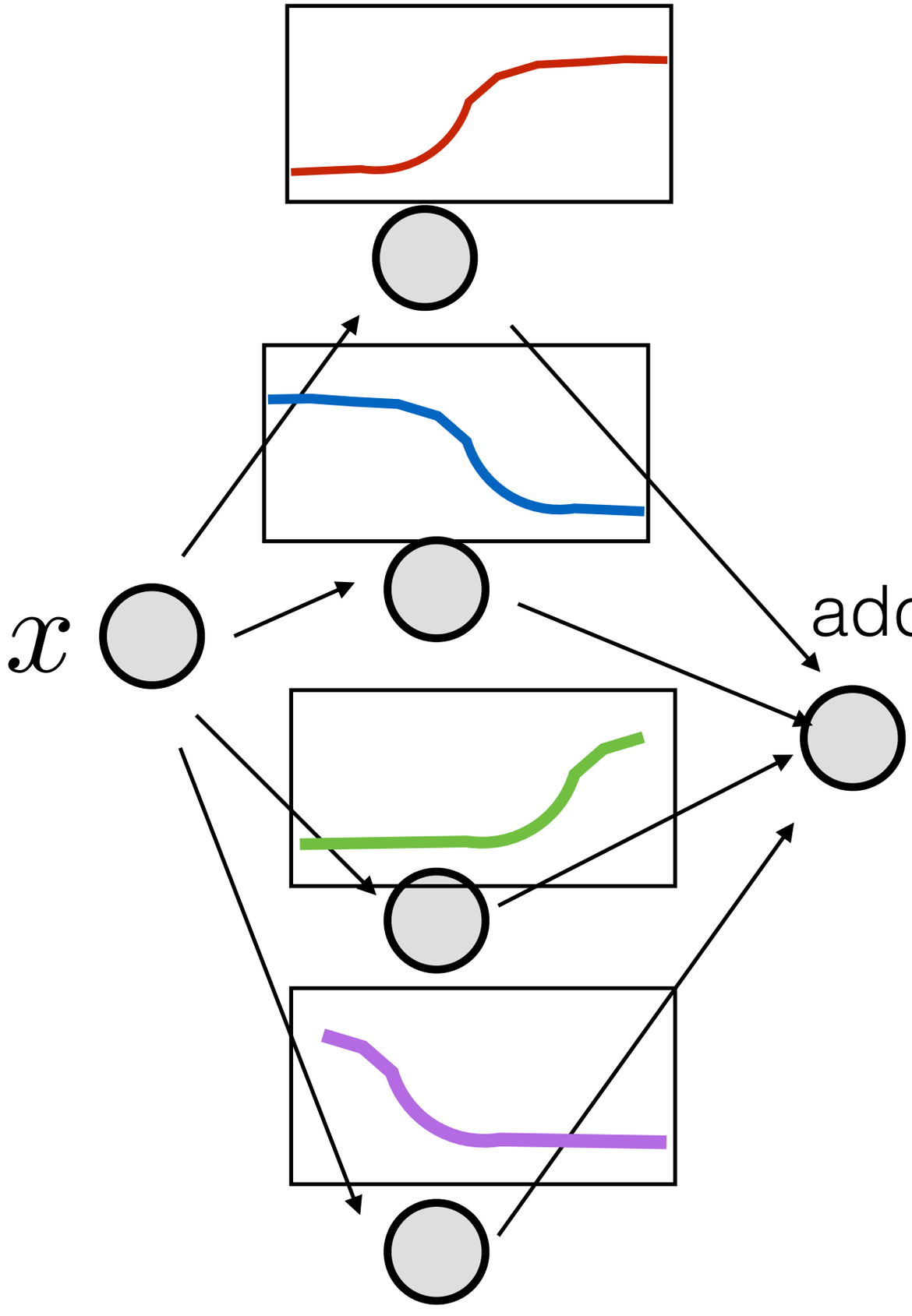


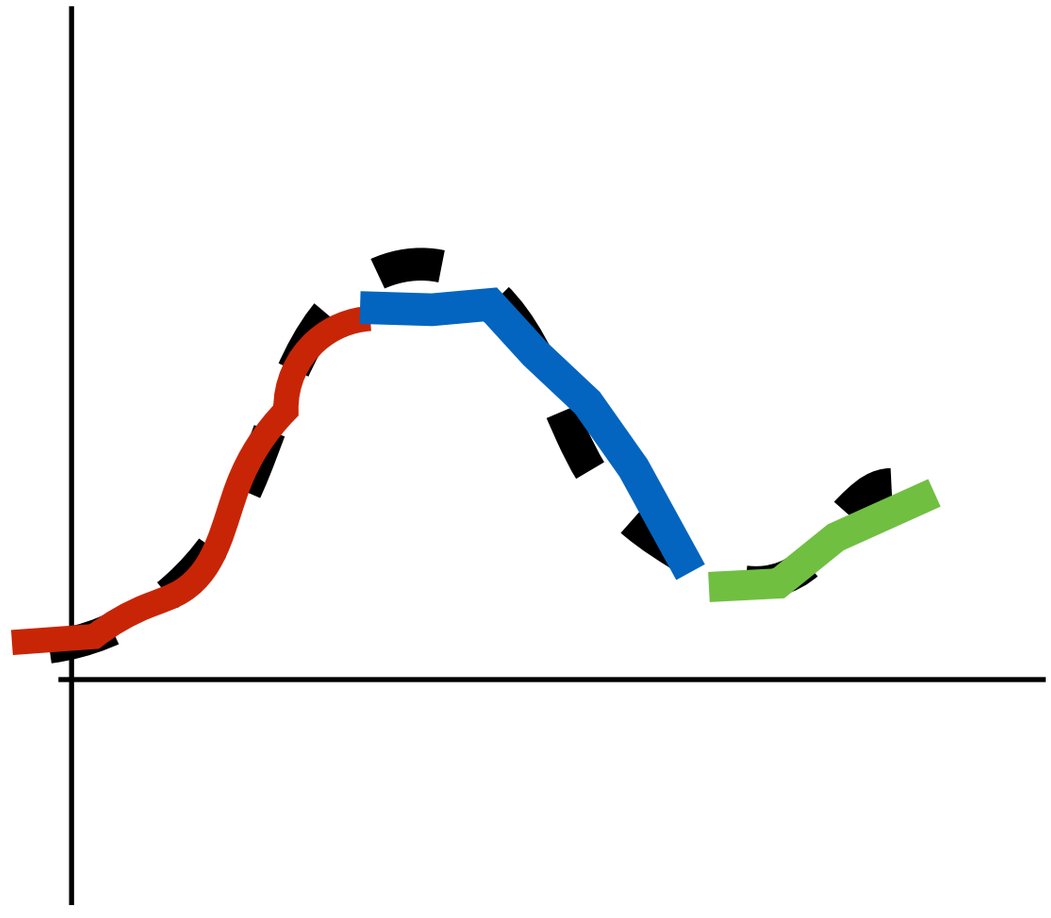

$x$
add



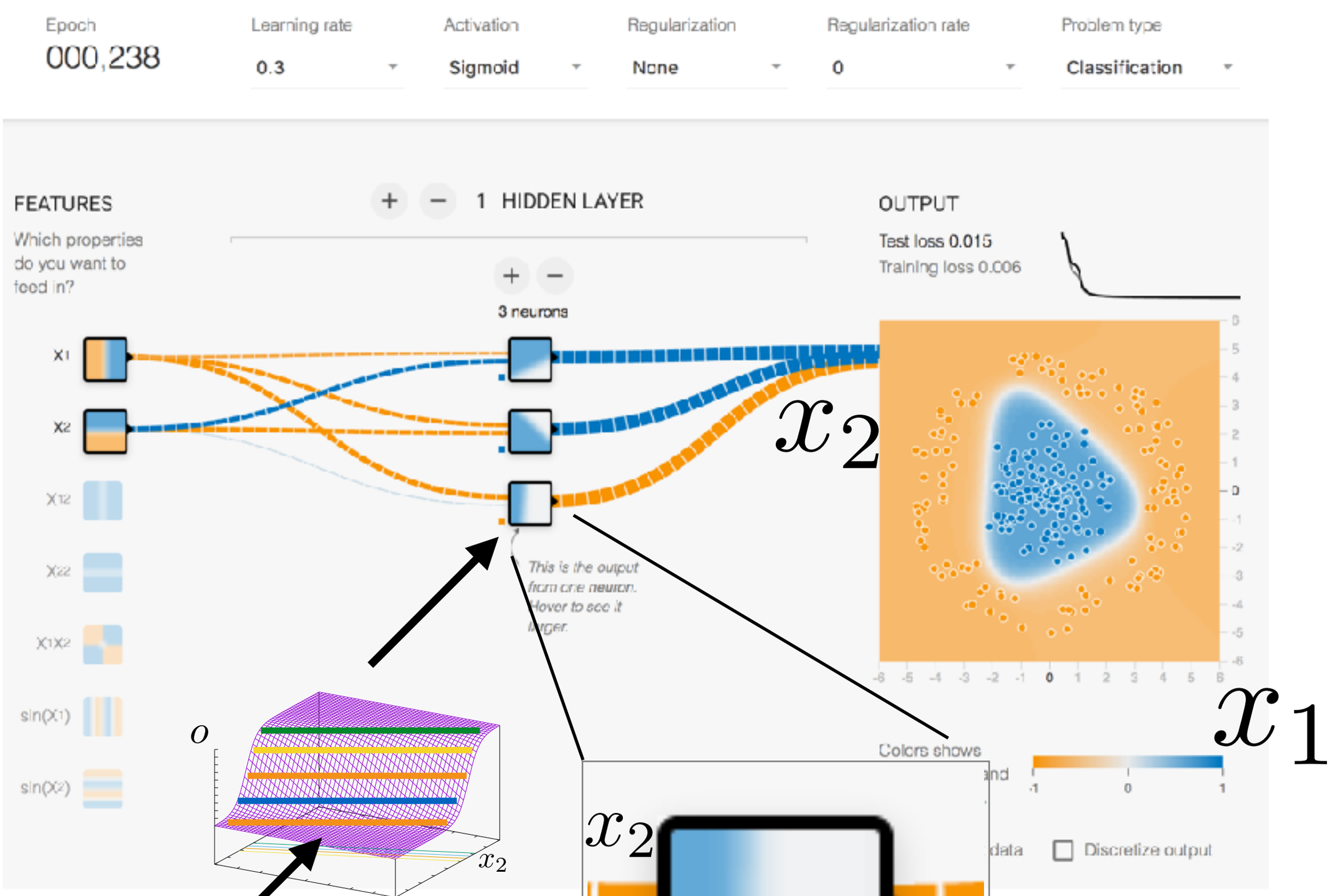

$x$
add




$x$
add



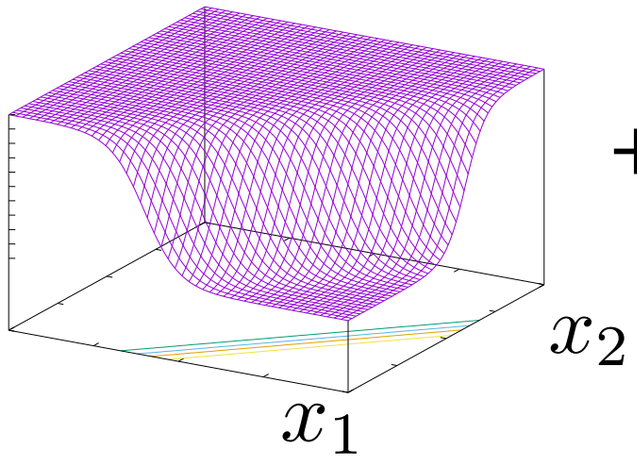
# Function picture of network




level sets form straight lines

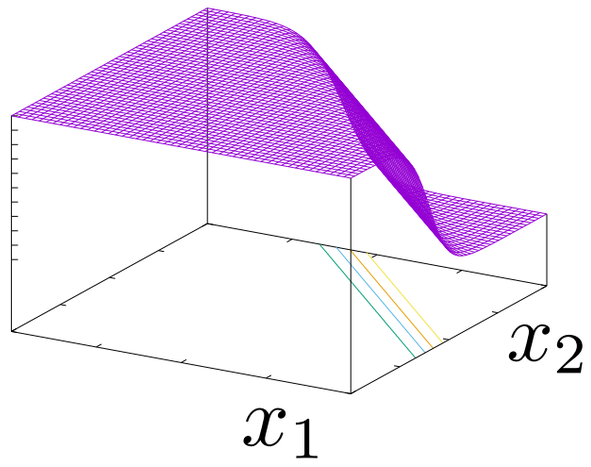
# Function view of neural network

$h_1(x_1, x_2)$



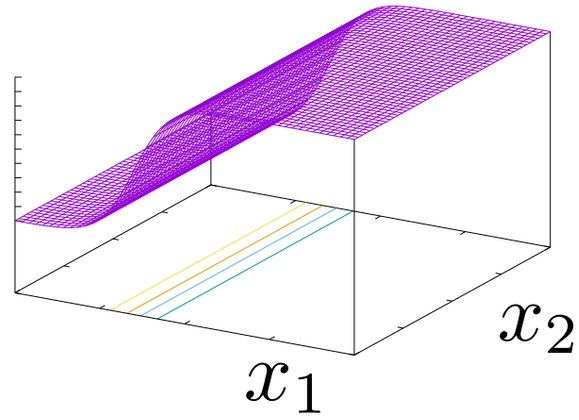

+

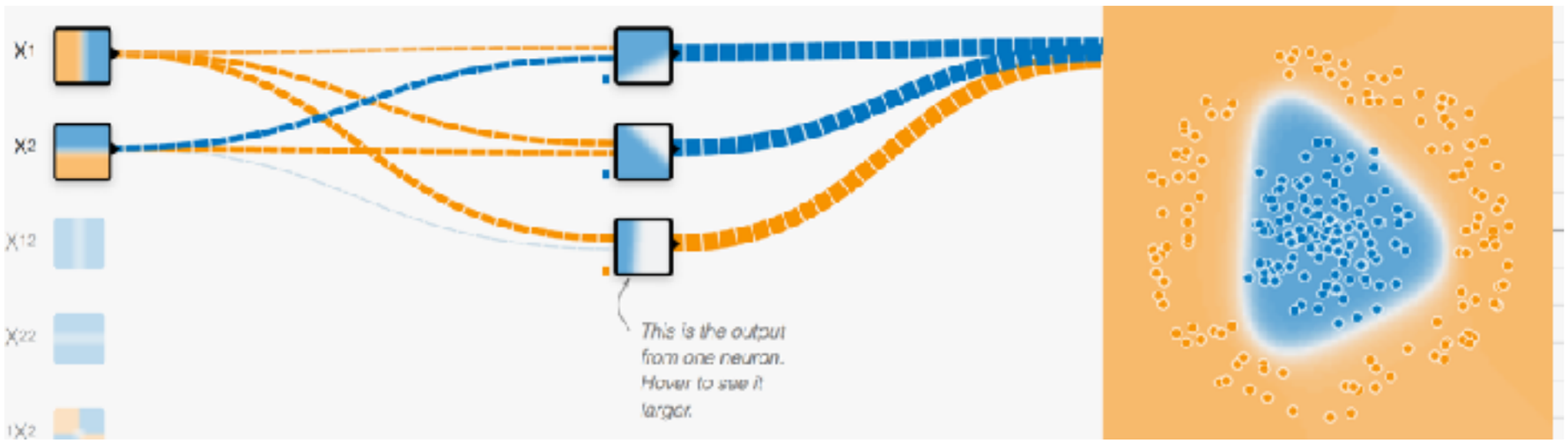
$h_2(x_1, x_2)$

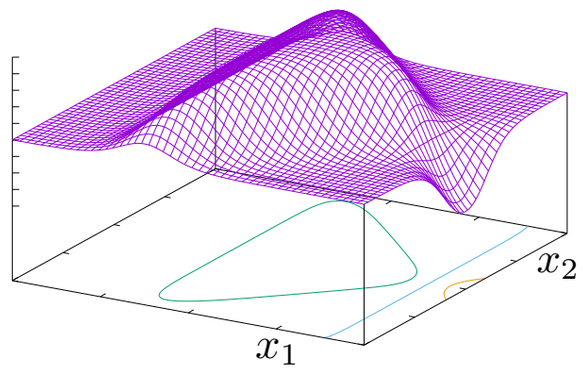



+

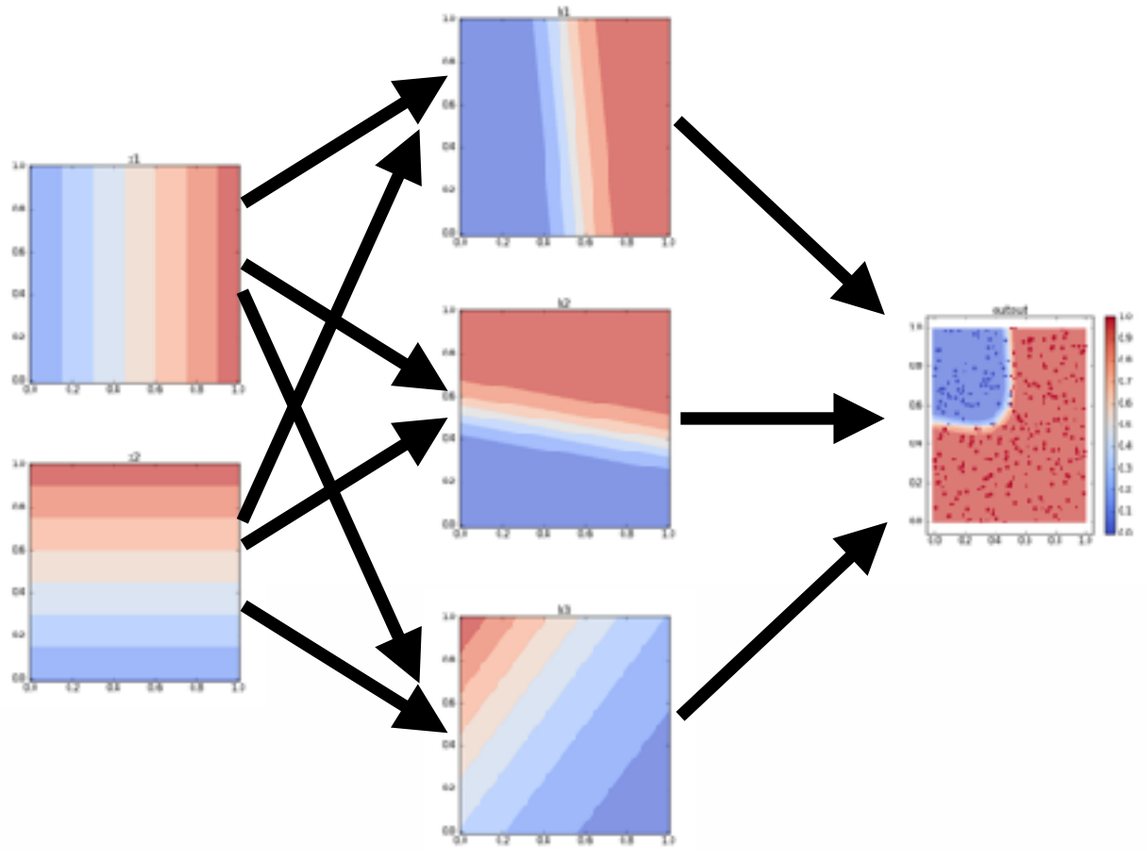
$-h_3(x_1, x_2)$

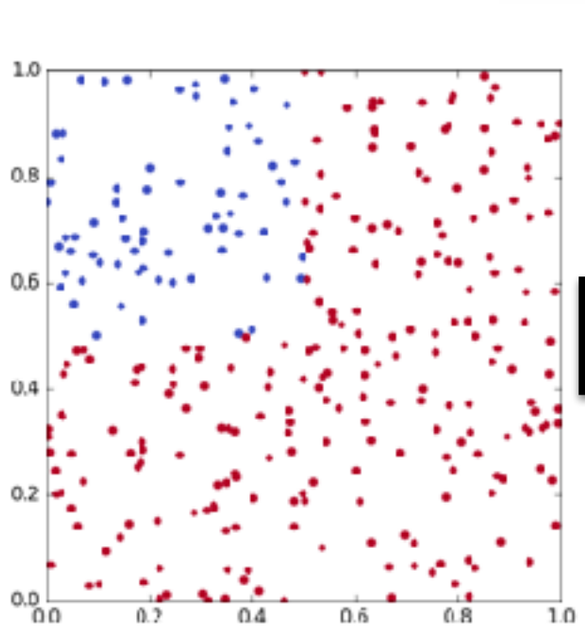


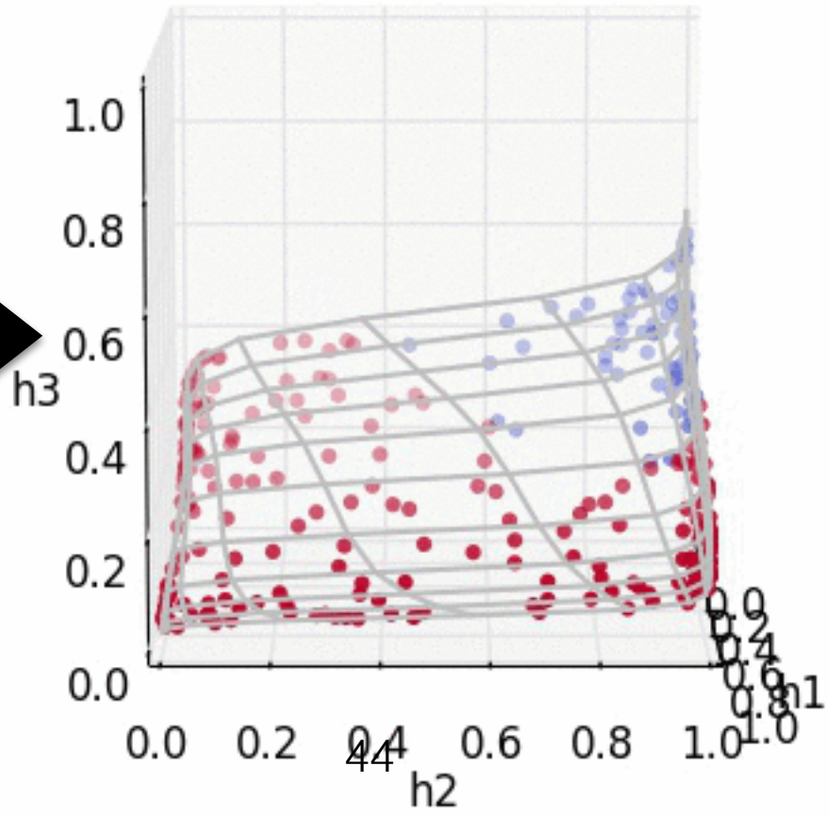



# Manifold view of neural network



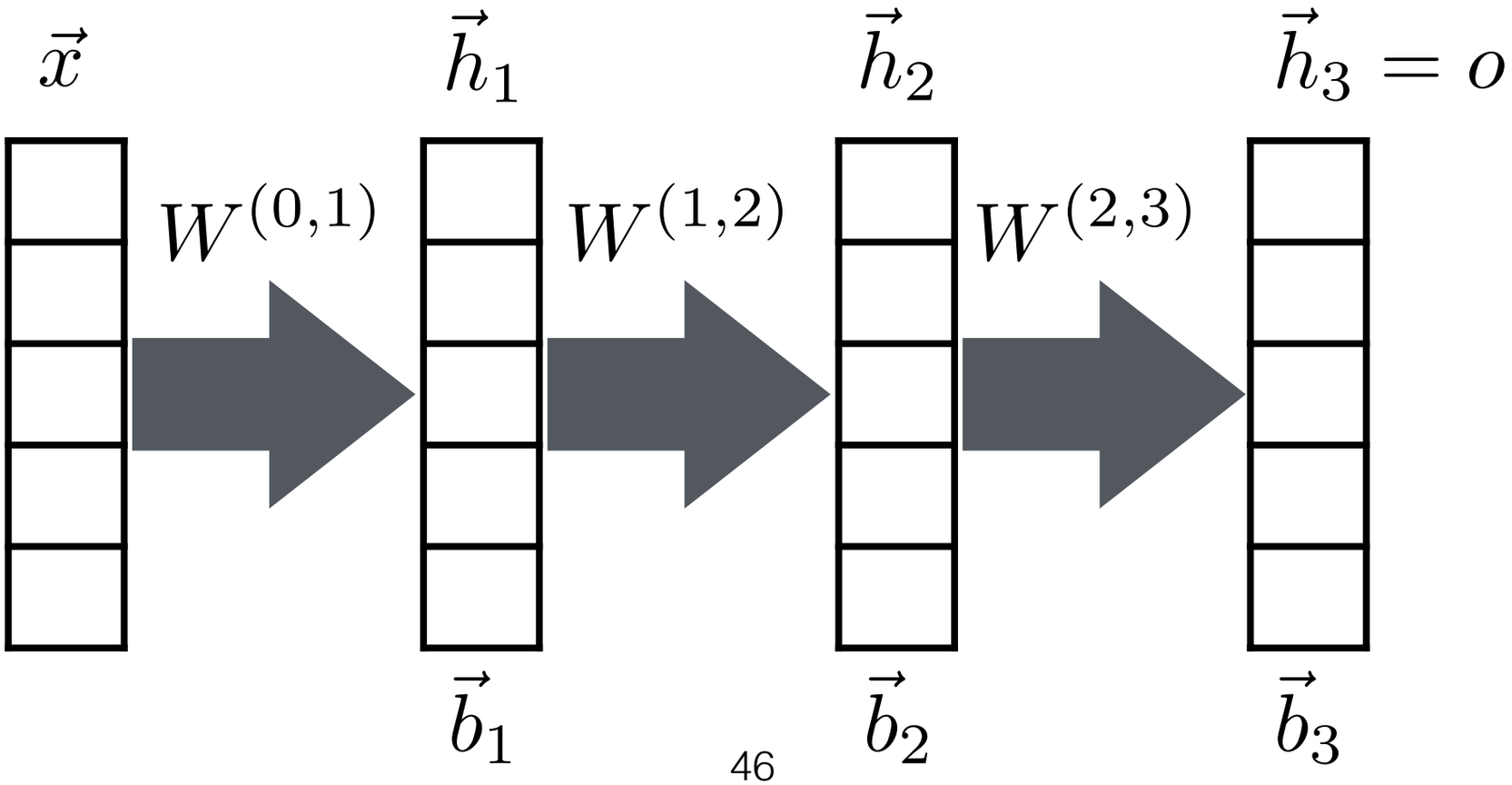


# Matrix notation

$$h_{1,i} = \sigma\left(\sum_j W^{(0,1)}_{j,i} x_j + b_i\right) \qquad i = 1, \cdots$$

$$\vec{h}_1 = \sigma[W^{(0,1)} \cdot \vec{x} + \vec{b}_1]$$ element wise operation

$$\sigma[(z_1, z_2, z_3, \cdots)] = (\sigma(z_1), \sigma(z_2), \sigma(z_3))$$

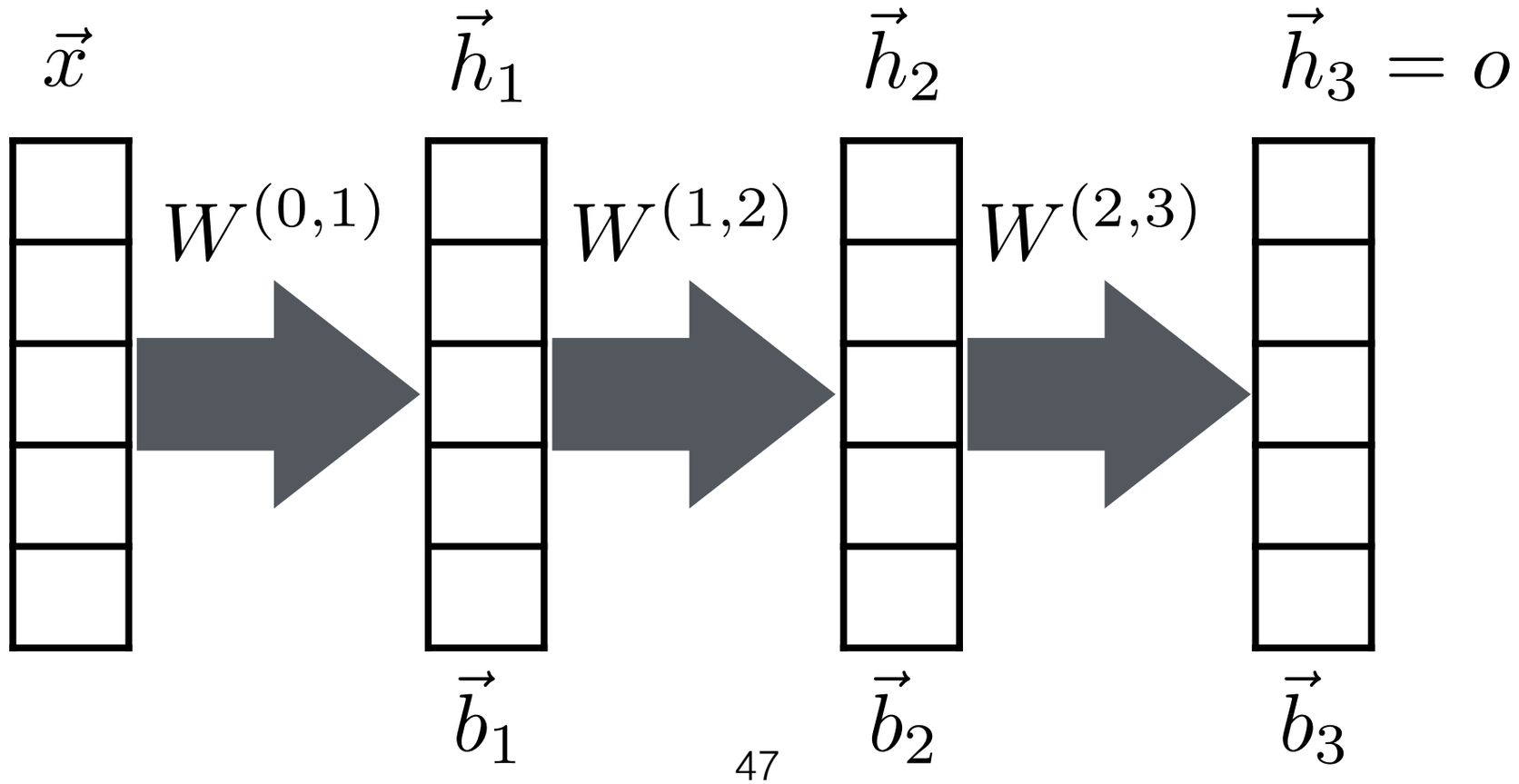


$$\vec{h}_1 = \sigma[W^{(0,1)} \cdot \vec{x} + \vec{b}_1]$$

$$\vec{h}_2 = \sigma[W^{(1,2)} \cdot \vec{h}_1 + \vec{b}_2]$$

$$\vec{h}_3 = \sigma[W^{(2,3)} \cdot \vec{h}_2 + \vec{b}_3]$$

$$\sigma[(z_1, z_2, z_3, \cdots)] = (\sigma(z_1), \sigma(z_2), \sigma(z_3))$$



# Training the network

How does the network know how to fold data space?

# Notation

Let $x \in \mathbb{R}^d$ be the input space

Let $y \in \mathbb{R}$ or $y \in \mathbb{N}$ be the label

Let $o \in \mathbb{R}$ be the output of the neural network

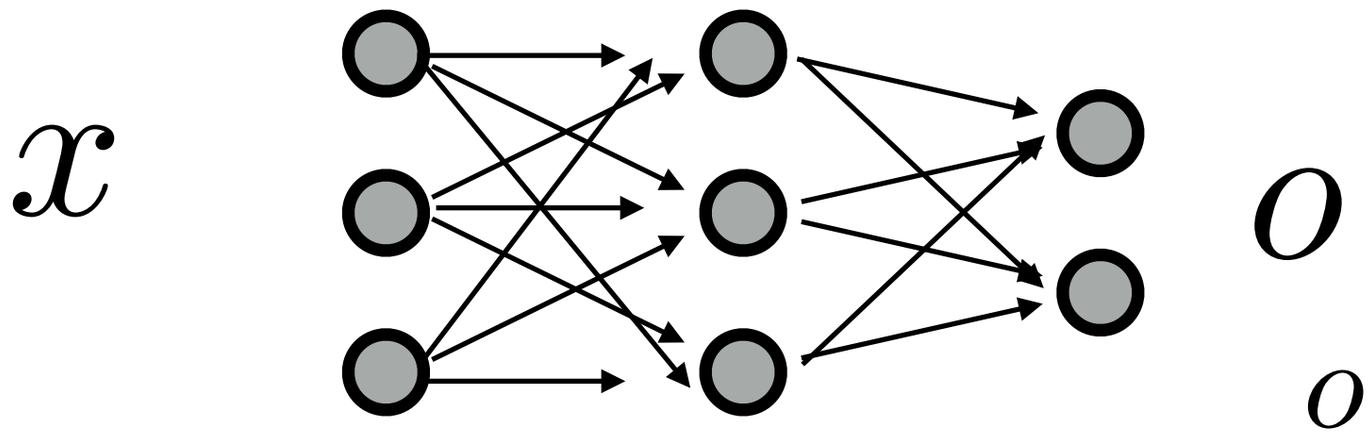
# A general objective

Tweak the output of neural network to be as similar to the label as possible.

$$o \approx y$$

How to tweak? Adjust the weights

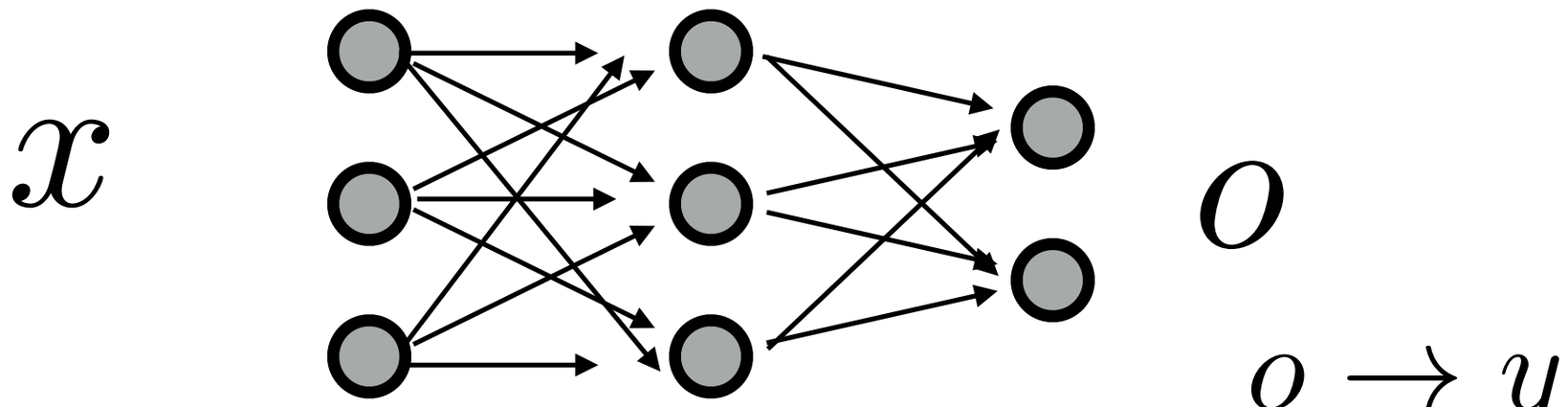
Given **one** example $(x, y)$




For this example, the neural network ‘behaves well’

Given **many** examples

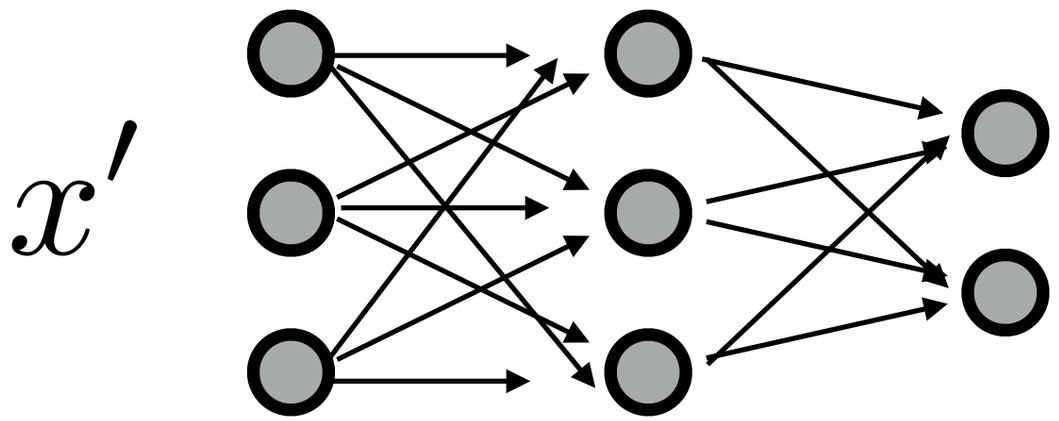
$$
\begin{array}{ll}
(x_1, y_1) & o_1 \to y_1 \\
(x_2, y_2) & o_2 \to y_2 \\
\vdots & \vdots \\
(x_n, y_n) & o_n \to y_n
\end{array}
$$




For these many examples,
the neural network ‘behaves well’

After the first example, give another example

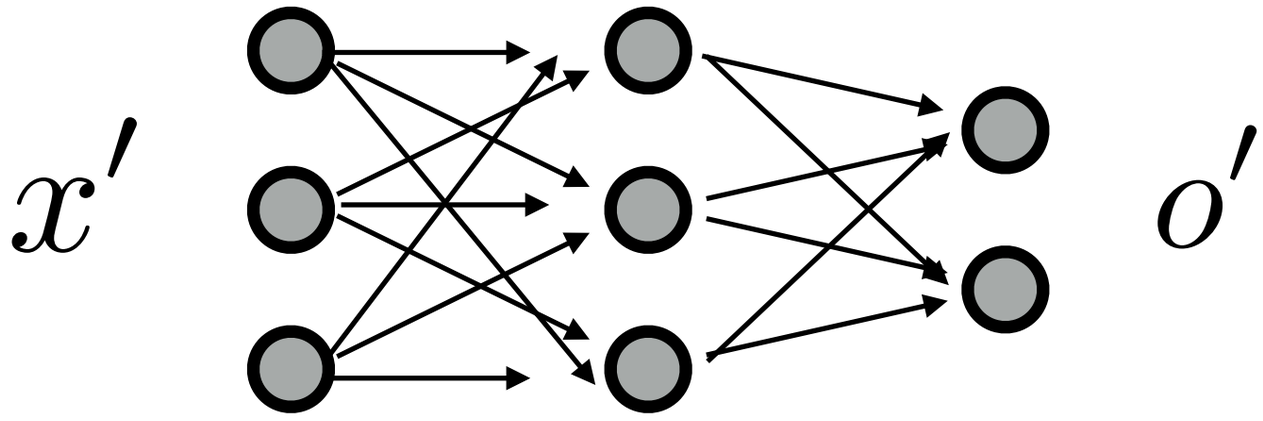
$$(x', ?)$$




Now we say for a well behaved neural network

$$o' = y'$$

# How good is the trained network?

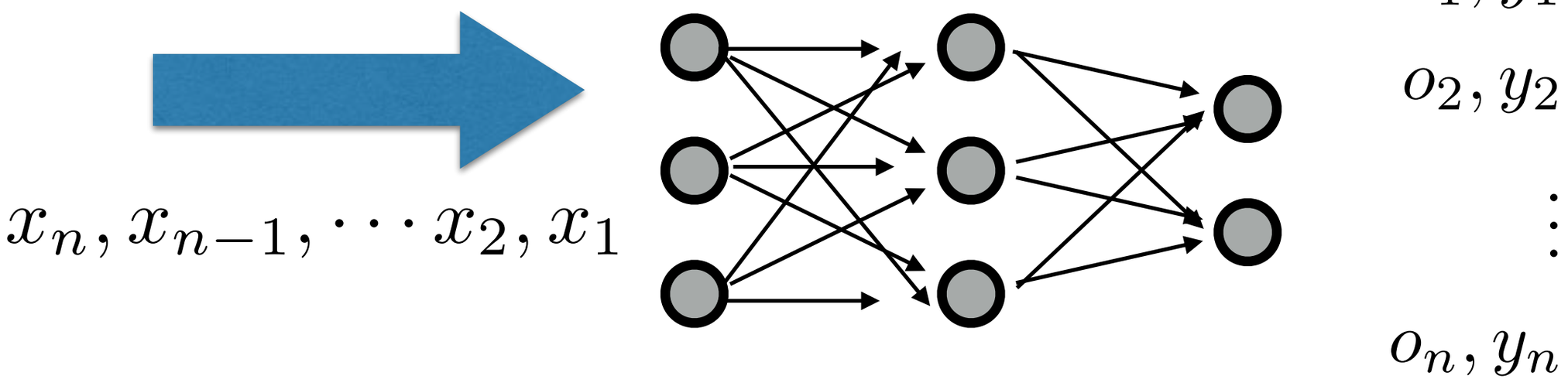


# Square error

Given data points $(x_i, y_i), i = 1, \cdots n$

$$l(y_i, o_i) = \|y_i - o_i\|^2$$

$$C = \frac{1}{n} \sum_i l(y_i, o_i)$$




Adjust the weights until C = 0 (or close to zero)

# Cross entropy cost function

Two class example $y_i \in \{0, 1\}$

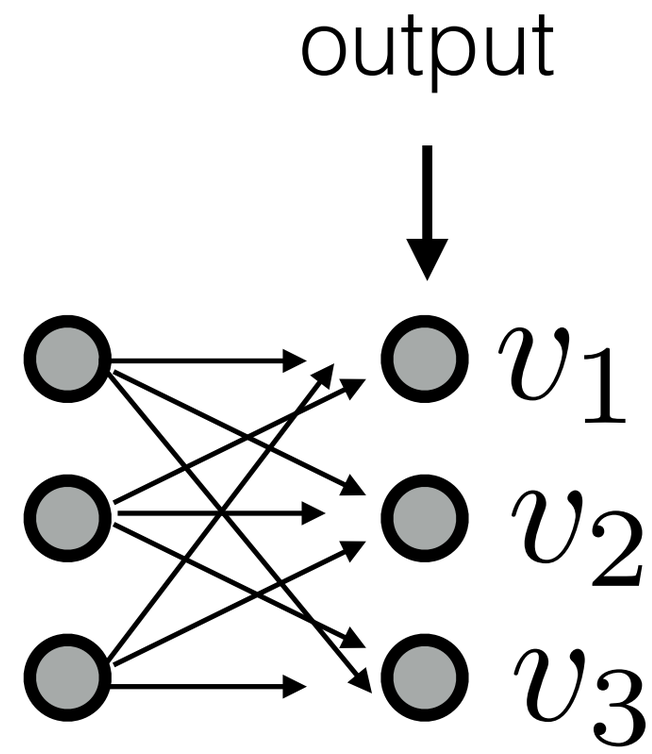
$$C = -\frac{1}{n} \sum_i y_i \log[o(x_i)] + (1 - y_i) \log[1 - o(x_i)]$$

Three class example

$y_i \in \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$

$$m = \exp(v_1) + \exp(v_2) + \exp(v_3)$$

$$o_i = (\frac{\exp(v_1)}{m}, \frac{\exp(v_2)}{m}, \frac{\exp(v_3)}{m})$$




$$l(y_i, o_i) = -y_i \cdot \log(o_i)$$

Here the log is element wise since oi is a vector

vector dot product

$$C = \frac{1}{n} \sum_i l(y_i, o_i)$$

A good online reference for loss function

http://rohanvarma.me/Loss-Functions/

# Loss function is a surface

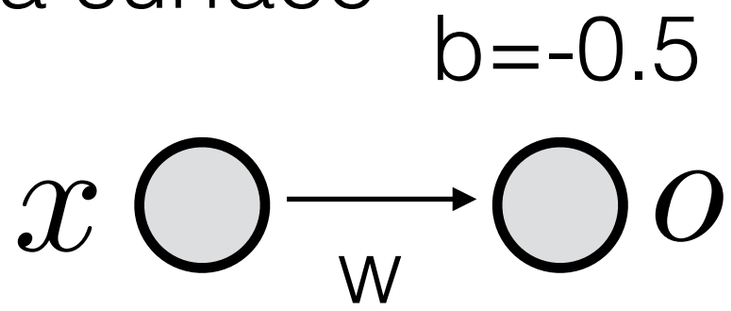
consider 4 data points

x1 = 0.1, y1 = 0

x2 = 0.2, y2 = 0

x3 = 0.6, y3 = 1

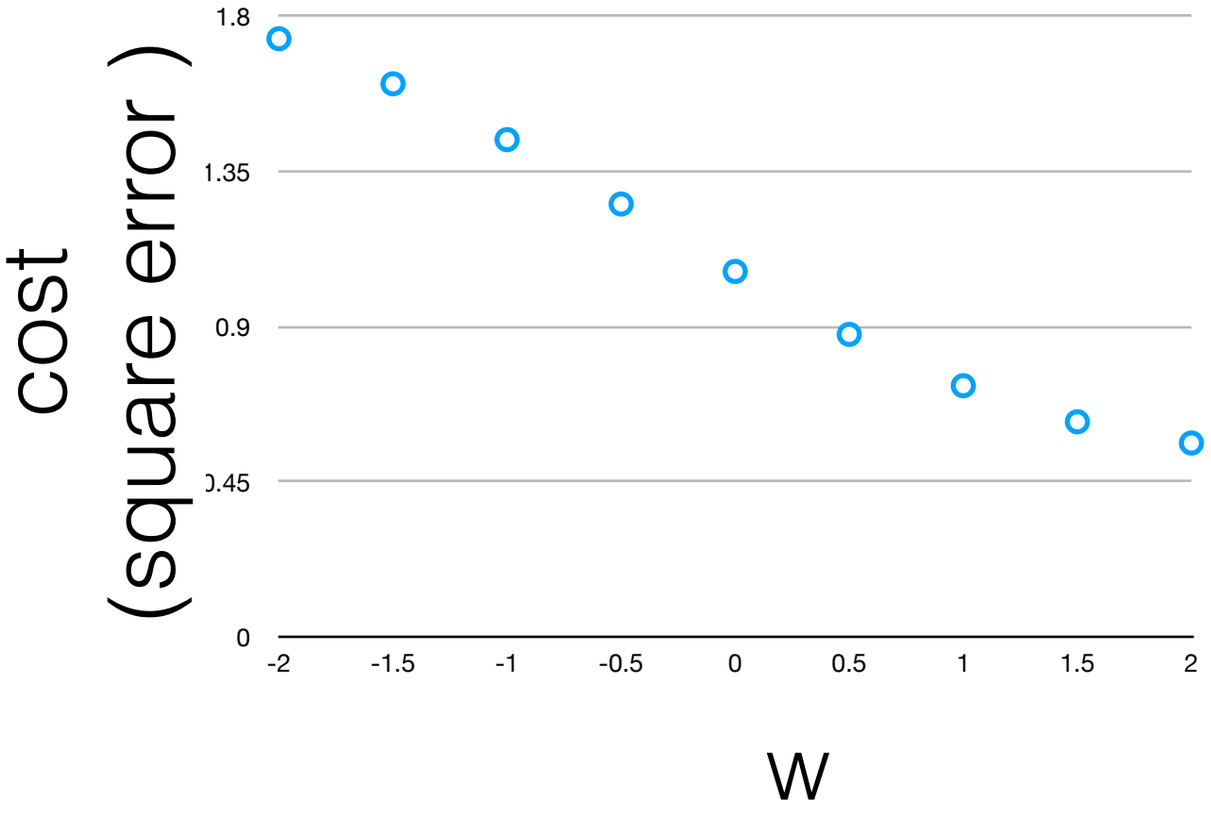
x4 = 0.9, y4 = 1




$$o = \frac{1}{1 + \exp(-wx)}$$

Table 1

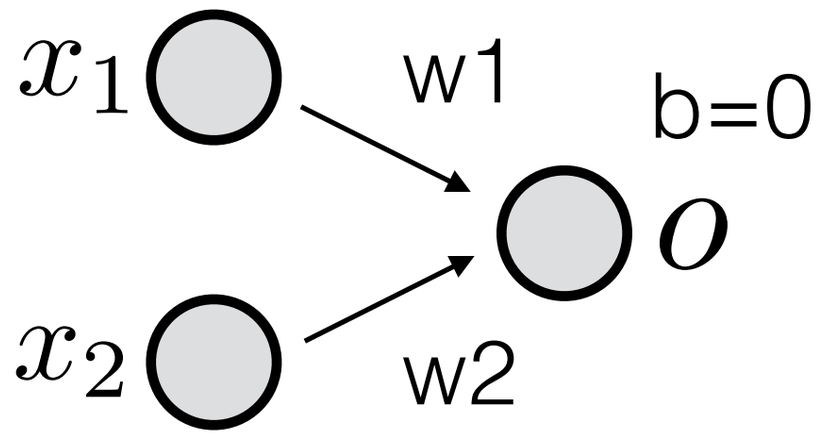
| w | x1 | o1 | y1 | x2 | o2 | y2 | x3 | o3 | y3 | x4 | o4 | y4 | Loss |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **-2** | 0.1 | 0.33 | 0 | 0.2 | 0.29 | 0 | 0.6 | 0.15 | 1 | 0.9 | 0.09 | 1 | 1.7436 |
| **-1.5** | 0.1 | 0.34 | 0 | 0.2 | 0.31 | 0 | 0.6 | 0.20 | 1 | 0.9 | 0.14 | 1 | 1.5913 |
| **-1** | 0.1 | 0.35 | 0 | 0.2 | 0.33 | 0 | 0.6 | 0.25 | 1 | 0.9 | 0.20 | 1 | 1.4339 |
| **-0.5** | 0.1 | 0.37 | 0 | 0.2 | 0.35 | 0 | 0.6 | 0.31 | 1 | 0.9 | 0.28 | 1 | 1.2539 |
| **0** | 0.1 | 0.38 | 0 | 0.2 | 0.38 | 0 | 0.6 | 0.38 | 1 | 0.9 | 0.38 | 1 | 1.0576 |
| **0.5** | 0.1 | 0.39 | 0 | 0.2 | 0.40 | 0 | 0.6 | 0.45 | 1 | 0.9 | 0.49 | 1 | 0.8747 |
| **1** | 0.1 | 0.40 | 0 | 0.2 | 0.43 | 0 | 0.6 | 0.52 | 1 | 0.9 | 0.60 | 1 | 0.7353 |
| **1.5** | 0.1 | 0.41 | 0 | 0.2 | 0.45 | 0 | 0.6 | 0.60 | 1 | 0.9 | 0.70 | 1 | 0.6206 |
| **2** | 0.1 | 0.43 | 0 | 0.2 | 0.48 | 0 | 0.6 | 0.67 | 1 | 0.9 | 0.78 | 1 | 0.5726 |

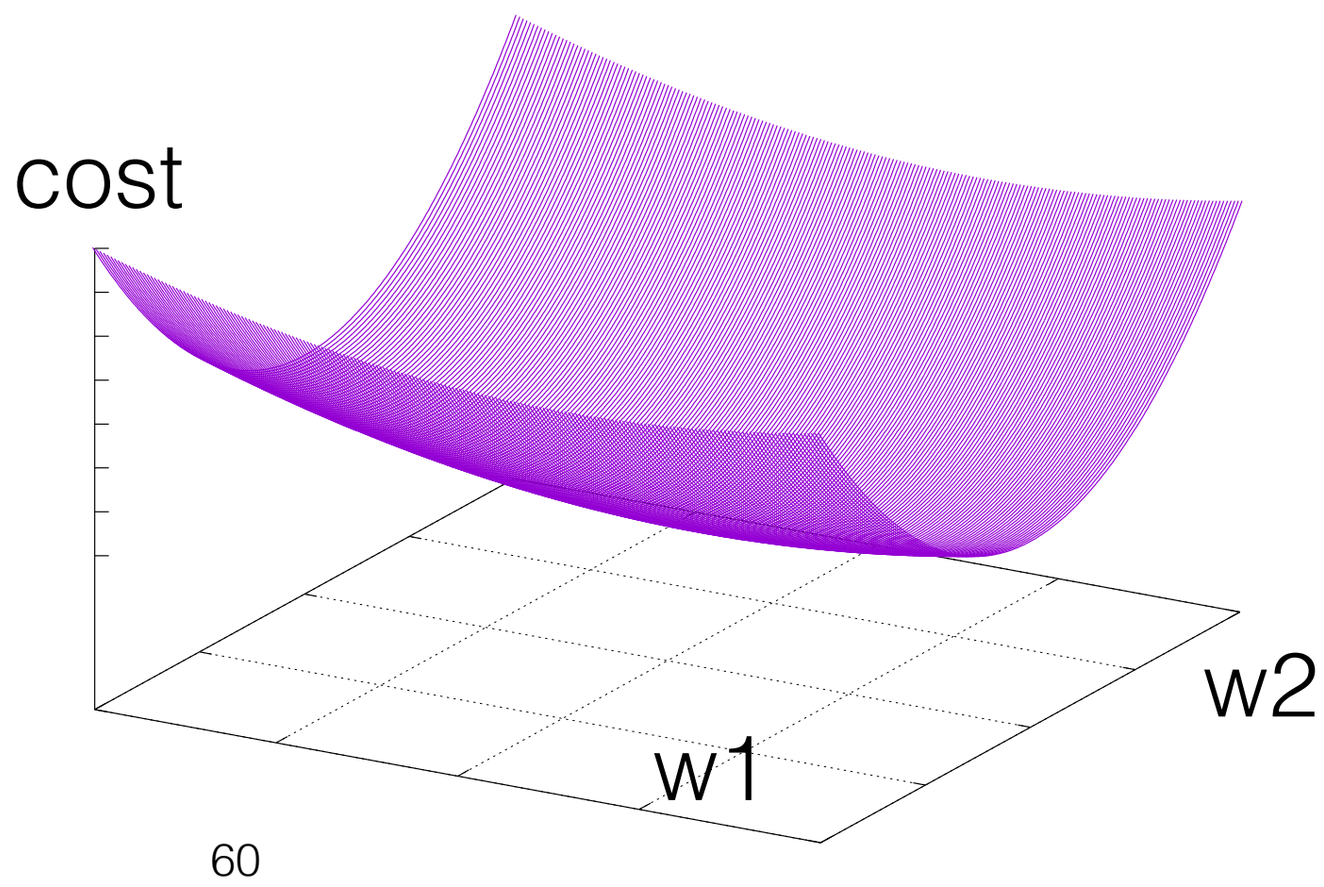



| w | x1 | o1 | y1 | x2 | o2 | y2 | x3 | o3 | y3 | x4 | o4 | y4 | Loss |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **-2** | 0.1 | 0.33 | 0 | 0.2 | 0.29 | 0 | 0.6 | 0.15 | 1 | 0.9 | 0.09 | 1 | 1.7436 |
| **-1.5** | 0.1 | 0.34 | 0 | 0.2 | 0.31 | 0 | 0.6 | 0.20 | 1 | 0.9 | 0.14 | 1 | 1.5913 |
| **-1** | 0.1 | 0.35 | 0 | 0.2 | 0.33 | 0 | 0.6 | 0.25 | 1 | 0.9 | 0.20 | 1 | 1.4339 |
| **-0.5** | 0.1 | 0.37 | 0 | 0.2 | 0.35 | 0 | 0.6 | 0.31 | 1 | 0.9 | 0.28 | 1 | 1.2539 |
| **0** | 0.1 | 0.38 | 0 | 0.2 | 0.38 | 0 | 0.6 | 0.38 | 1 | 0.9 | 0.38 | 1 | 1.0576 |
| **0.5** | 0.1 | 0.39 | 0 | 0.2 | 0.40 | 0 | 0.6 | 0.45 | 1 | 0.9 | 0.49 | 1 | 0.8747 |
| **1** | 0.1 | 0.40 | 0 | 0.2 | 0.43 | 0 | 0.6 | 0.52 | 1 | 0.9 | 0.60 | 1 | 0.7353 |
| **1.5** | 0.1 | 0.41 | 0 | 0.2 | 0.45 | 0 | 0.6 | 0.60 | 1 | 0.9 | 0.70 | 1 | 0.6206 |
| **2** | 0.1 | 0.43 | 0 | 0.2 | 0.48 | 0 | 0.6 | 0.67 | 1 | 0.9 | 0.78 | 1 | 0.5726 |

$$l(y_i, o_i) = \|y_i - o_i\|^2$$

$$C(w_1, w_2) = \frac{1}{n} \sum_i l(y_i, o_i)$$



# How to adjust the weights?

Conceptually, this will work:

> Try all possible weights combinations, for all weights combinations, calculate the cost for all combinations. Then pick the weight combination that has the lowest cost.

**Practically, there are too many weights combinations to try.**

# Back propagation and gradient descend

# Basic calculus required for understanding backpropagation

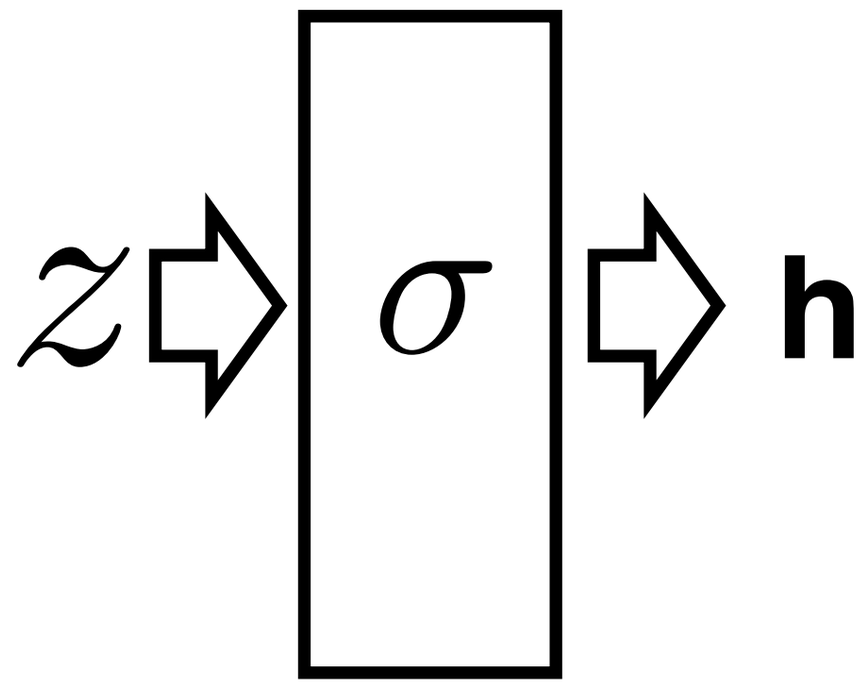
## Composite function

$$f(x_1, x_2, w_1, w_2, b) = w_1 x_1 + w_2 x_2 + b$$

$$f(x_1, x_2, w_1, w_2, b) = z = w_1 x_1 + w_2 x_2 + b$$

$$\sigma(u) = \frac{1}{1 + \exp(-u)}$$

$$h = \sigma(f(x_1, x_2, w_1, w_2, b))$$



How $h$ will change if there is a small change in u?

$$h = \sigma(u) = \frac{1}{1+\exp(-u)}$$

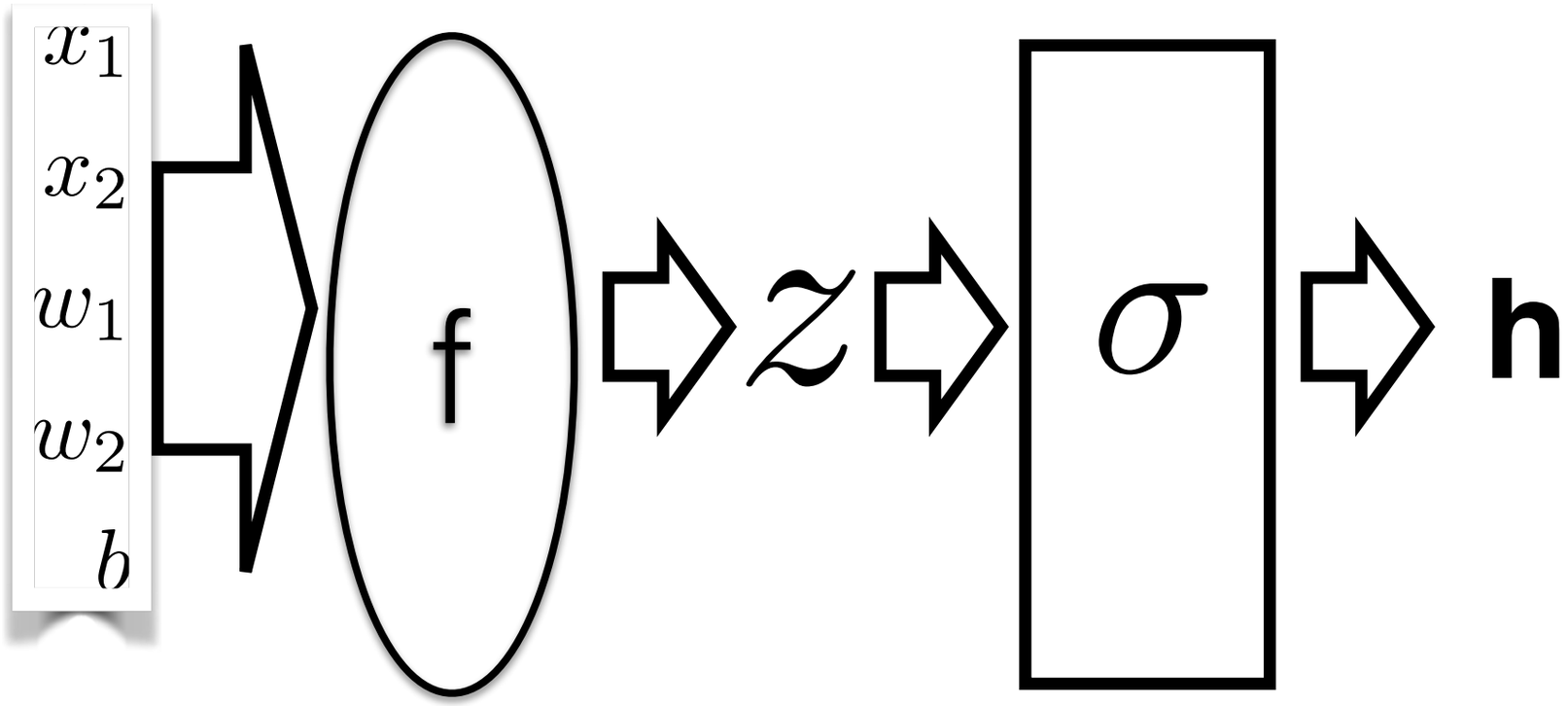
$$\frac{\partial \sigma(u)}{\partial u} = \sigma(u)\left(1-\sigma(u)\right)$$

How $h$ will change if there is a small change in w1?

$$h = \sigma(z) = \sigma(f(x_1, x_2, w_1, w_2, b))$$

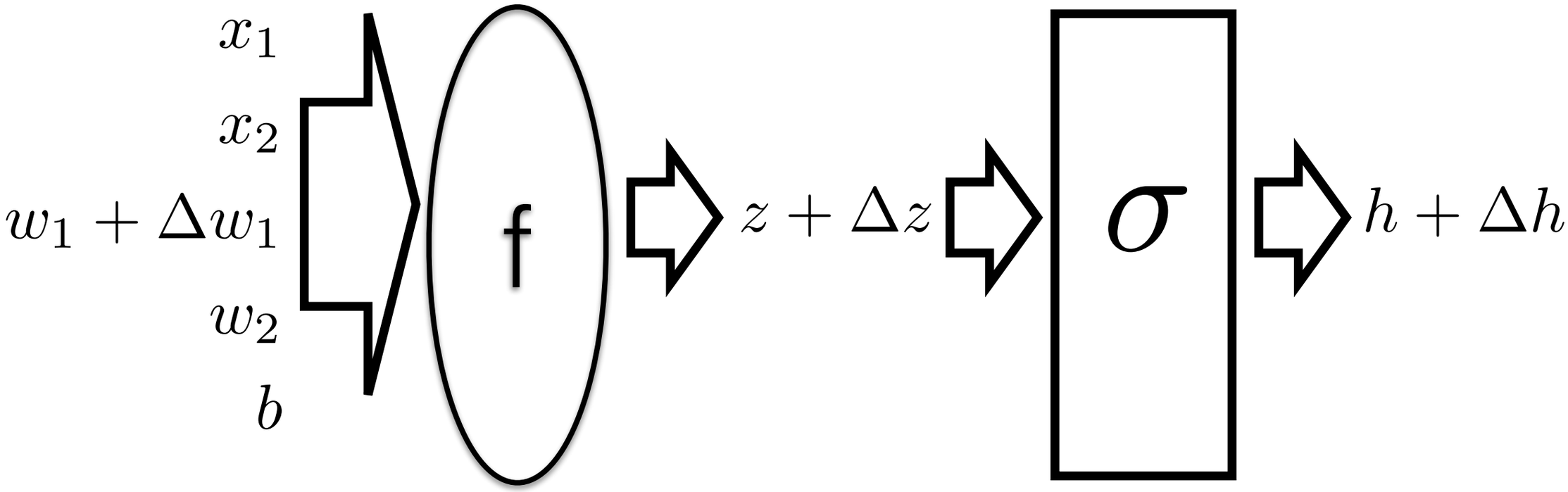
Chain rule $$\frac{\partial h}{\partial w_1} = \frac{\partial \sigma(z)}{\partial w_1} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial w_1}$$

$$\frac{\partial h}{\partial w_2} = \frac{\partial \sigma(z)}{\partial w_2} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial w_2} \qquad \frac{\partial h}{\partial b} = \frac{\partial \sigma(z)}{\partial b} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial b}$$




$$\frac{\partial h}{\partial w_1} = \frac{\partial \sigma(z)}{\partial w_1} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial w_1}$$

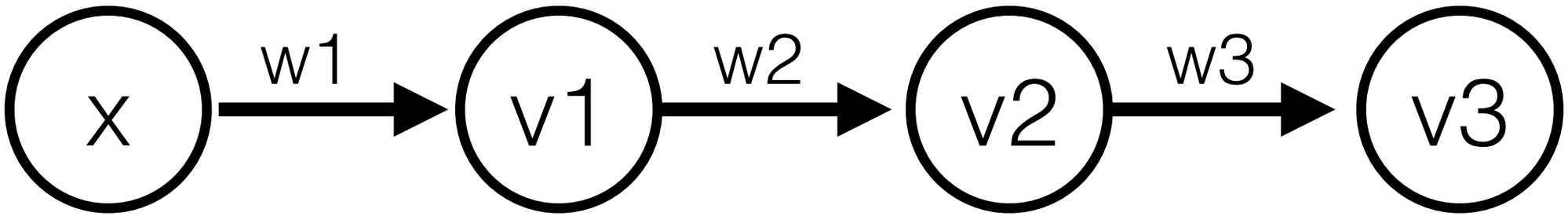
$$\frac{\partial h}{\partial w_2} = \frac{\partial \sigma(z)}{\partial w_2} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial w_2} \qquad \frac{\partial h}{\partial b} = \frac{\partial \sigma(z)}{\partial b} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial b}$$




$$\frac{\partial h}{\partial w_1} = \frac{\partial \sigma(z)}{\partial w_1} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial w_1}$$

$$\frac{\partial h}{\partial w_2} = \frac{\partial \sigma(z)}{\partial w_2} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial w_2} \qquad \frac{\partial h}{\partial b} = \frac{\partial \sigma(z)}{\partial b} = \frac{\partial \sigma(z)}{\partial z}\frac{\partial z}{\partial b}$$

# Gradient descend




$$v_1 = \sigma(z_1) = \sigma(w_1 x + b_1)$$

$$v_2 = \sigma(z_2) = \sigma(w_2 v_1 + b_2)$$

$$v_3 = \sigma(z_3) = \sigma(w_3 v_2 + b_3)$$

$$\frac{\partial C}{\partial w_j} = 0 \qquad \frac{\partial C}{\partial b_j} = 0$$

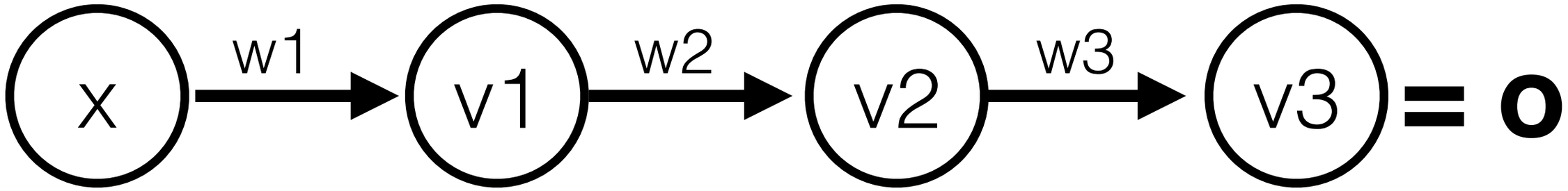
$$\frac{\partial C}{\partial w_j} = 0 \qquad\qquad \frac{\partial C}{\partial b_j} = 0$$

$$\frac{\partial C}{\partial w_j} = \frac{1}{n}\sum_i \frac{\partial (y_i - o_i)^2}{\partial w_j}$$

$$= -\frac{2}{n}\sum_i (y_i - o_i)\frac{\partial o_i}{\partial w_j}$$

We just need $\dfrac{\partial o_i}{\partial w_j}$

$$w_j(t+1) = w_j(t) - \eta \frac{\partial C}{\partial w_j}$$




$$v_1 = \sigma(z_1) = \sigma(w_1 x + b_1)$$

$$v_2 = \sigma(z_2) = \sigma(w_2 v_1 + b_2)$$

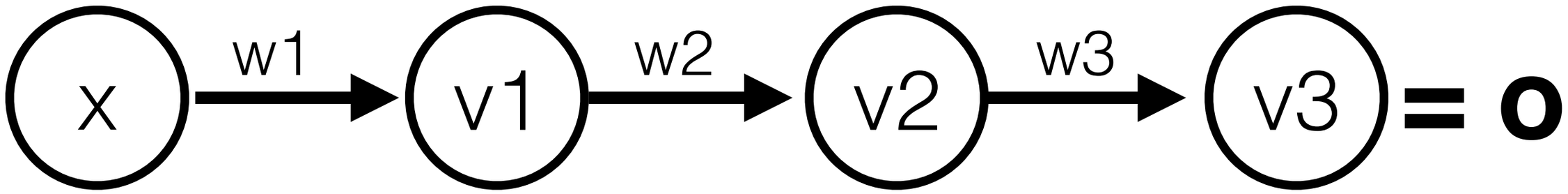
$$v_3 = \sigma(z_3) = \sigma(w_3 v_2 + b_3)$$

$$\frac{\partial v_3}{\partial w_3} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_3} = \sigma'(z_3)v_2$$

$$\frac{\partial v_3}{\partial w_2} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_2} = \sigma'(z_3)w_3\frac{\partial v_2}{\partial w_2} = \sigma'(z_3)w_3\sigma'(z_2)v_1$$

$$\frac{\partial v_3}{\partial w_1} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_1} = \sigma'(z_3)w_3\frac{\partial v_2}{\partial w_2} = \sigma'(z_3)w_3\sigma'(z_2)w_2\frac{\partial v_1}{\partial w_1}$$

$$= \sigma'(z_3)w_3\sigma'(z_2)w_2\sigma'(z_1)x$$



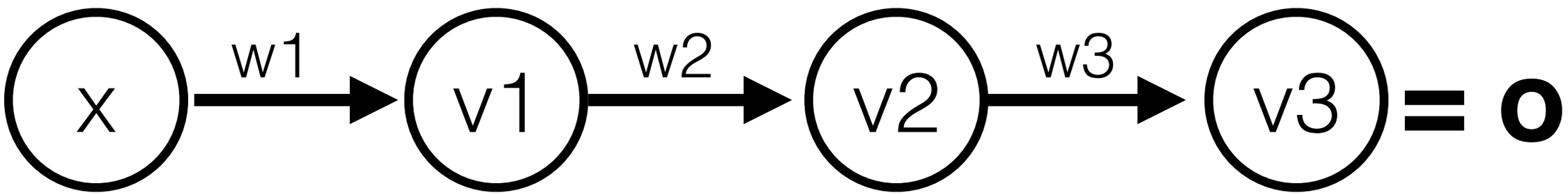

$$\frac{\partial v_3}{\partial w_3} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_3} = \sigma'(z_3)v_2$$

$$\frac{\partial v_3}{\partial w_2} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_2} = \sigma'(z_3)w_3\frac{\partial v_2}{\partial w_2} = \sigma'(z_3)w_3\sigma'(z_2)v_1$$

$$\frac{\partial v_3}{\partial w_1} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_1} = \sigma'(z_3)w_3\frac{\partial v_2}{\partial w_2} = \sigma'(z_3)w_3\sigma'(z_2)w_2\frac{\partial v_1}{\partial w_1}$$
$$= \sigma'(z_3)w_3\sigma'(z_2)w_2\sigma'(z_1)x$$

**problem!!** : long mathematical expression leads to large computational time for deep network

# Compute and store strategy



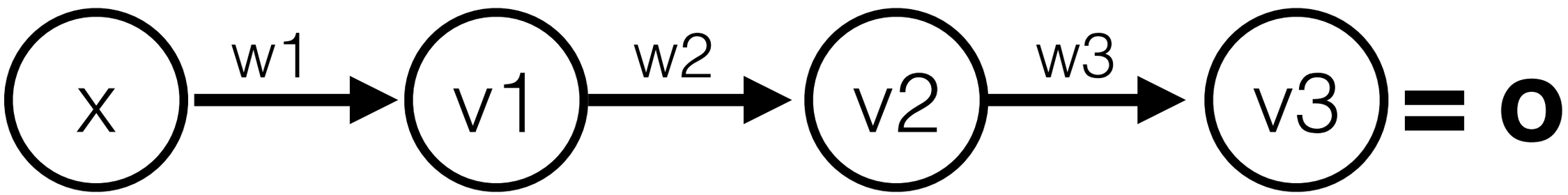

$$\frac{\partial v_3}{\partial z_1} \qquad \frac{\partial v_3}{\partial z_2} \qquad \frac{\partial v_3}{\partial z_3}$$

$$\frac{\partial v_3}{\partial z_3} = \sigma'(z_3)$$

$$\frac{\partial v_3}{\partial z_2} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial z_2} = \frac{\partial v_3}{\partial z_3} w_3 \sigma'(z_2)$$

$$\frac{\partial v_3}{\partial z_1} = \frac{\partial v_3}{\partial z_2}\frac{\partial z_2}{\partial z_1} = \frac{\partial v_3}{\partial z_2} w_2 \sigma'(z_1)$$

# Compute and store strategy




$$\frac{\partial v_3}{\partial z_1} \qquad \frac{\partial v_3}{\partial z_2} \qquad \frac{\partial v_3}{\partial z_3}$$

$$\frac{\partial v_3}{\partial w_3} = \frac{\partial v_3}{\partial z_3}\frac{\partial z_3}{\partial w_3} = \frac{\partial v_3}{\partial z_3} v_2$$

$$\frac{\partial v_3}{\partial w_2} = \frac{\partial v_3}{\partial z_2}\frac{\partial z_2}{\partial w_2} = \frac{\partial v_3}{\partial z_2} v_1$$

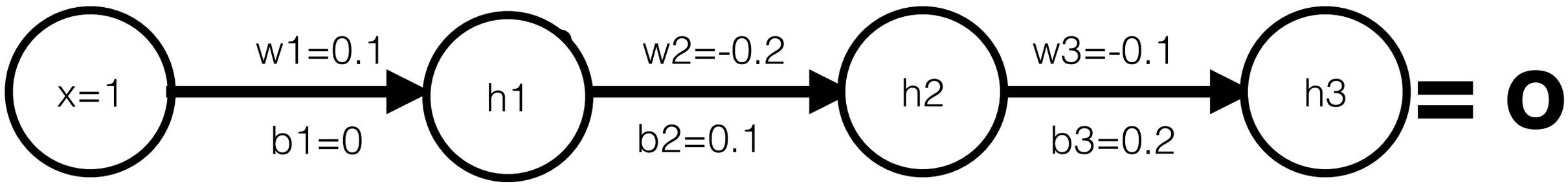
$$\frac{\partial v_3}{\partial w_1} = \frac{\partial v_3}{\partial z_1}\frac{\partial z_1}{\partial w_1} = \frac{\partial v_1}{\partial z_1} x$$

$\frac{\partial v_3}{\partial b_j}$? please work this out

# Forward pass

$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$

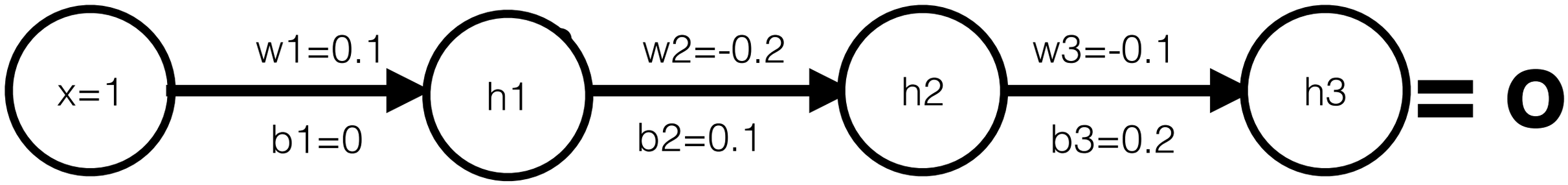



Compute h1,h2,h3 using Relu : please spend 5 minutes on this

# Now we put in real numbers

$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$




z1 = 0.1*1+0 = 0.1
h1 = 0.1

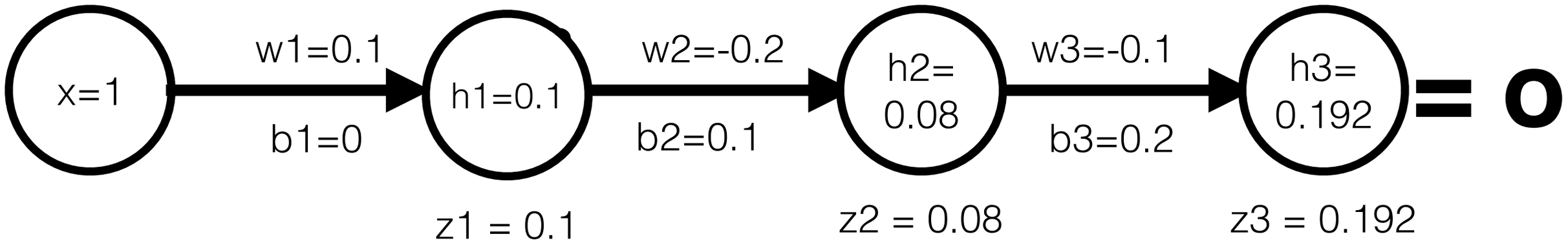
z2 = -0.2*0.1+0.1 = 0.08
h1 = 0.08

z3 = -0.1*0.08+0.2 = 0.192
h3 = 0.192

# Backward pass, compute all the gradients

$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

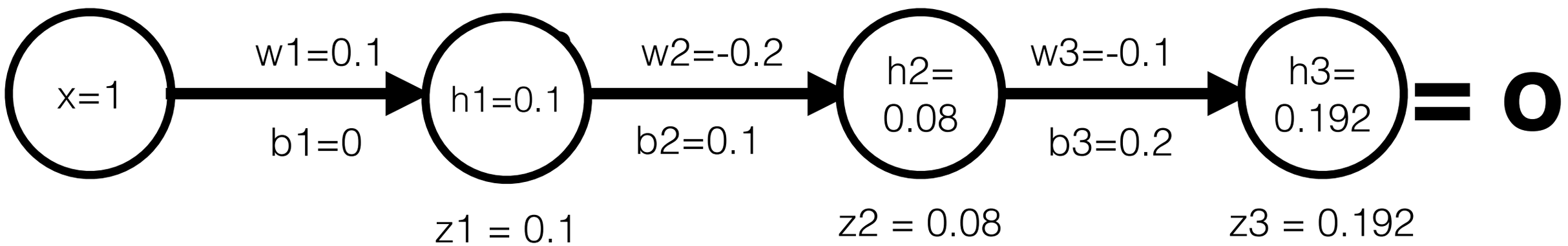
$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$



# Backward pass, compute all the gradients

$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$




$$\frac{\partial h_3}{\partial z_3} = 1$$

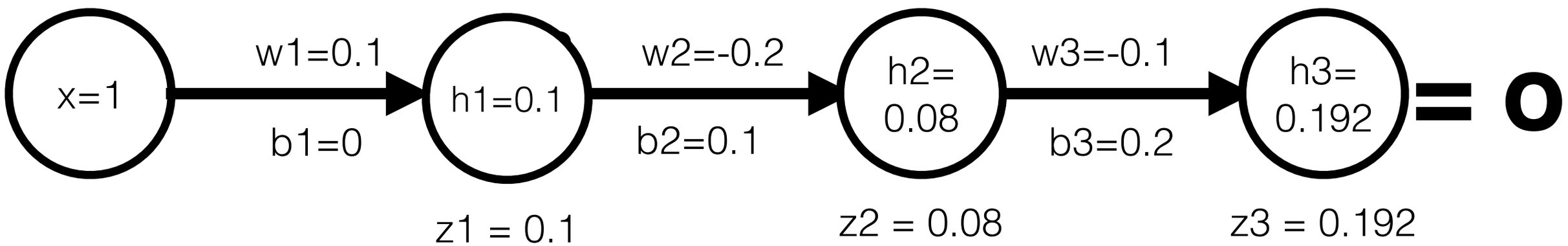
$$\frac{\partial h_3}{\partial z_2} = \frac{\partial h_3}{\partial z_3}\frac{\partial z_3}{\partial h_2}\frac{\partial h_2}{\partial z_2}$$

$$\frac{\partial h_3}{\partial z_1} = \frac{\partial h_3}{\partial z_3}\frac{\partial z_3}{\partial h_2}\frac{\partial h_2}{\partial z_2}\frac{\partial z_2}{\partial h_1}\frac{\partial h_1}{\partial z_1} = \frac{\partial h_3}{\partial z_2}\frac{\partial z_2}{\partial h_1}\frac{\partial h_1}{\partial z_1}$$

# Backward pass, compute all the gradients

$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$




$$\frac{\partial h_3}{\partial z_3} = 1$$

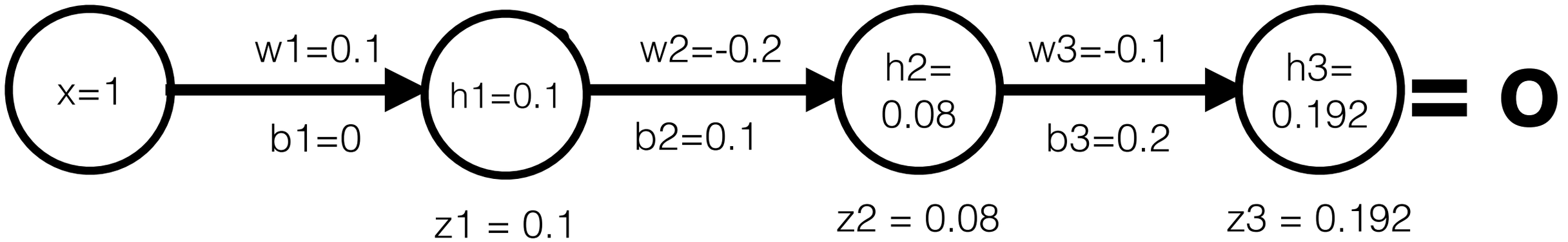
$$\frac{\partial h_3}{\partial z_2} =? \qquad \frac{\partial h_3}{\partial z_2} = \frac{\partial h_3}{\partial z_3}\frac{\partial z_3}{\partial h_2}\frac{\partial h_2}{\partial z_2} = (1)(-0.1)(1) = -0.1$$

$$\frac{\partial h_3}{\partial z_1} =? \; = \frac{\partial h_3}{\partial z_2}\frac{\partial z_2}{\partial h_1}\frac{\partial h_1}{\partial z_1} = (-0.1)(-0.2)(1) = 0.02$$

# Backward pass, compute all the gradients

$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$




$$\frac{\partial h_3}{\partial w_3} = \frac{\partial h_3}{\partial z_3}\frac{\partial z_3}{\partial w_3} = \quad (1)(0.08) = 0.08 \qquad \frac{\partial h_3}{\partial z_3} = 1$$

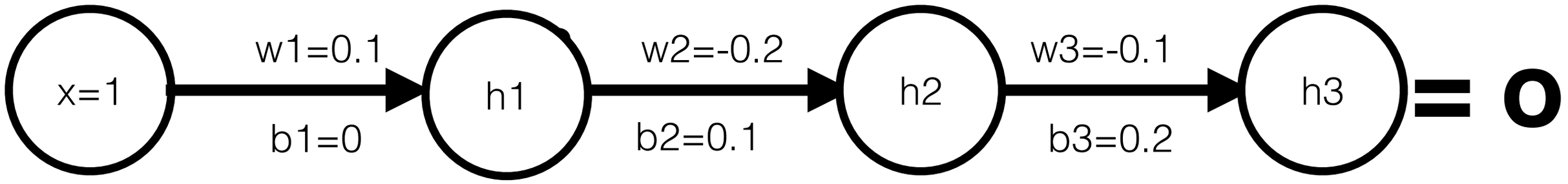
$$\frac{\partial h_3}{\partial w_2} = \frac{\partial h_3}{\partial z_2}\frac{\partial z_2}{\partial w_2} = (-0.1)(0.1) = -0.01 \qquad \frac{\partial h_3}{\partial z_2} = -0.1$$

$$\frac{\partial h_3}{\partial w_1} = \frac{\partial h_3}{\partial z_1}\frac{\partial z_1}{\partial w_1} = \quad (0.02)(1) = 0.02 \qquad \frac{\partial h_3}{\partial z_1} = 0.02$$

# Backward pass, compute all the gradients

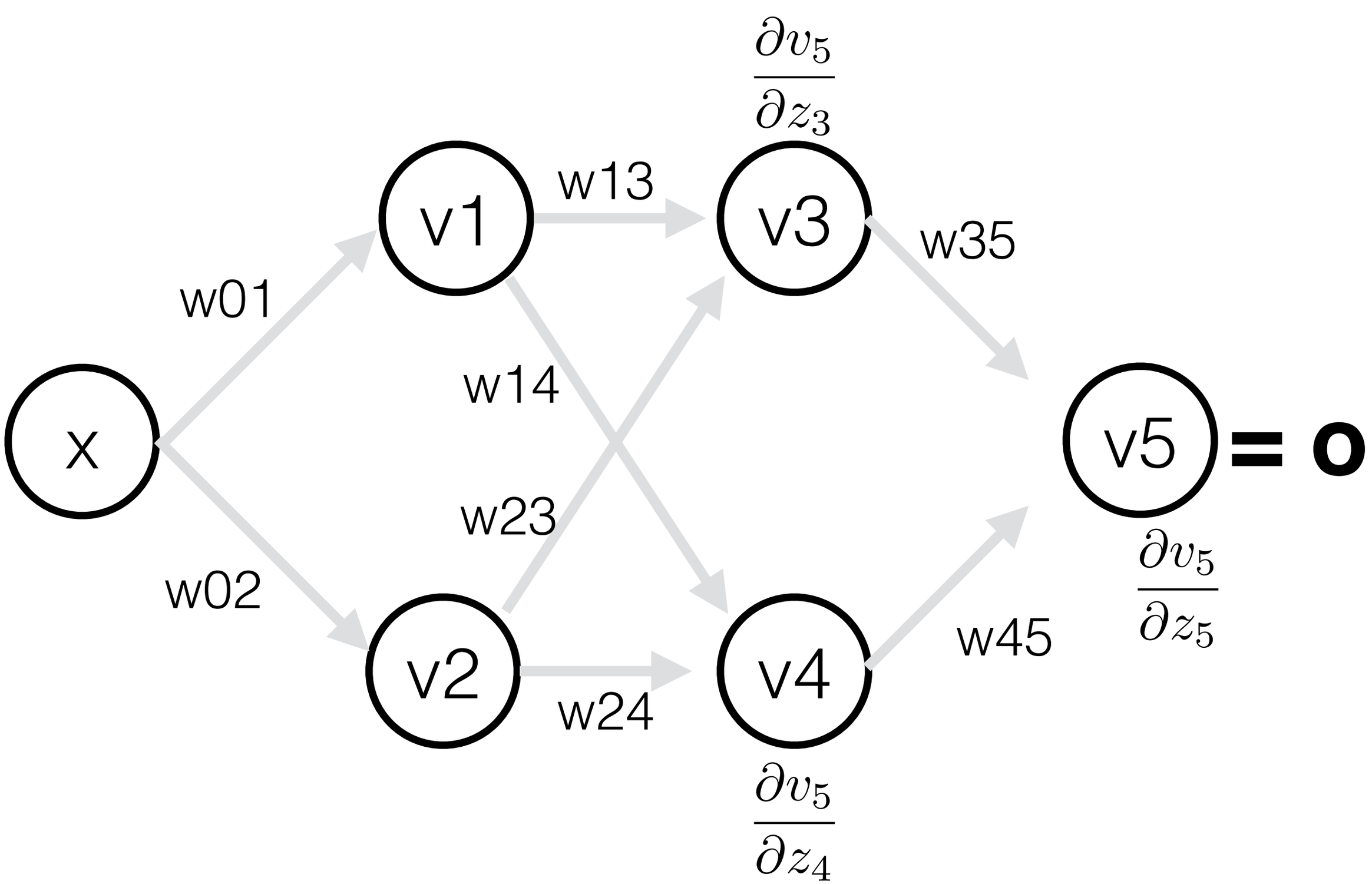
$$z_1 = w_1 x + b_1 \qquad z_2 = w_2 h_1 + b_2 \qquad z_3 = w_3 h_2 + b_3$$

$$h_1 = \sigma(z_1) \qquad h_2 = \sigma(z_2) \qquad h_3 = \sigma(z_3)$$

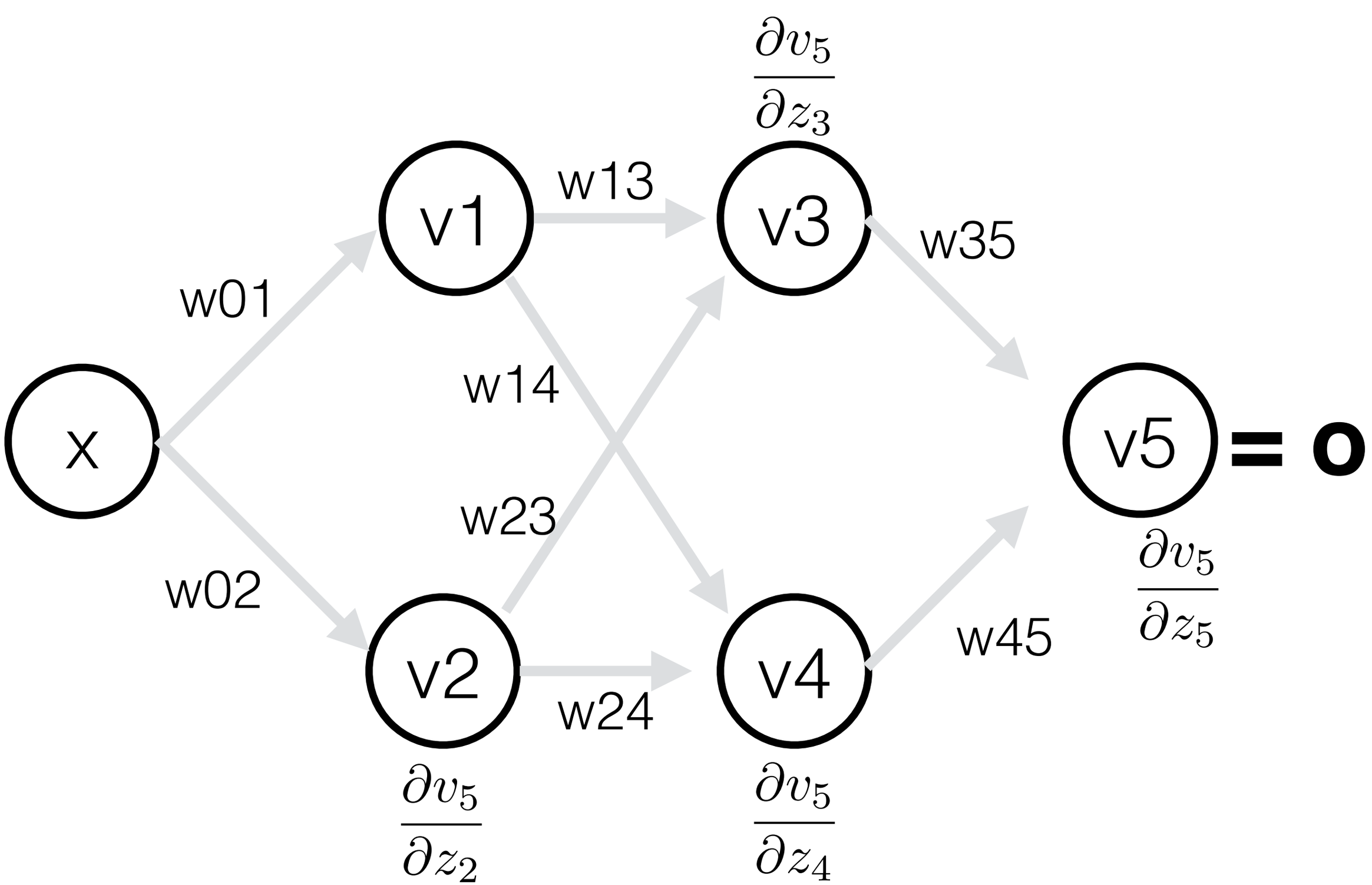



Please spend 2 minutes
to compute gradients for

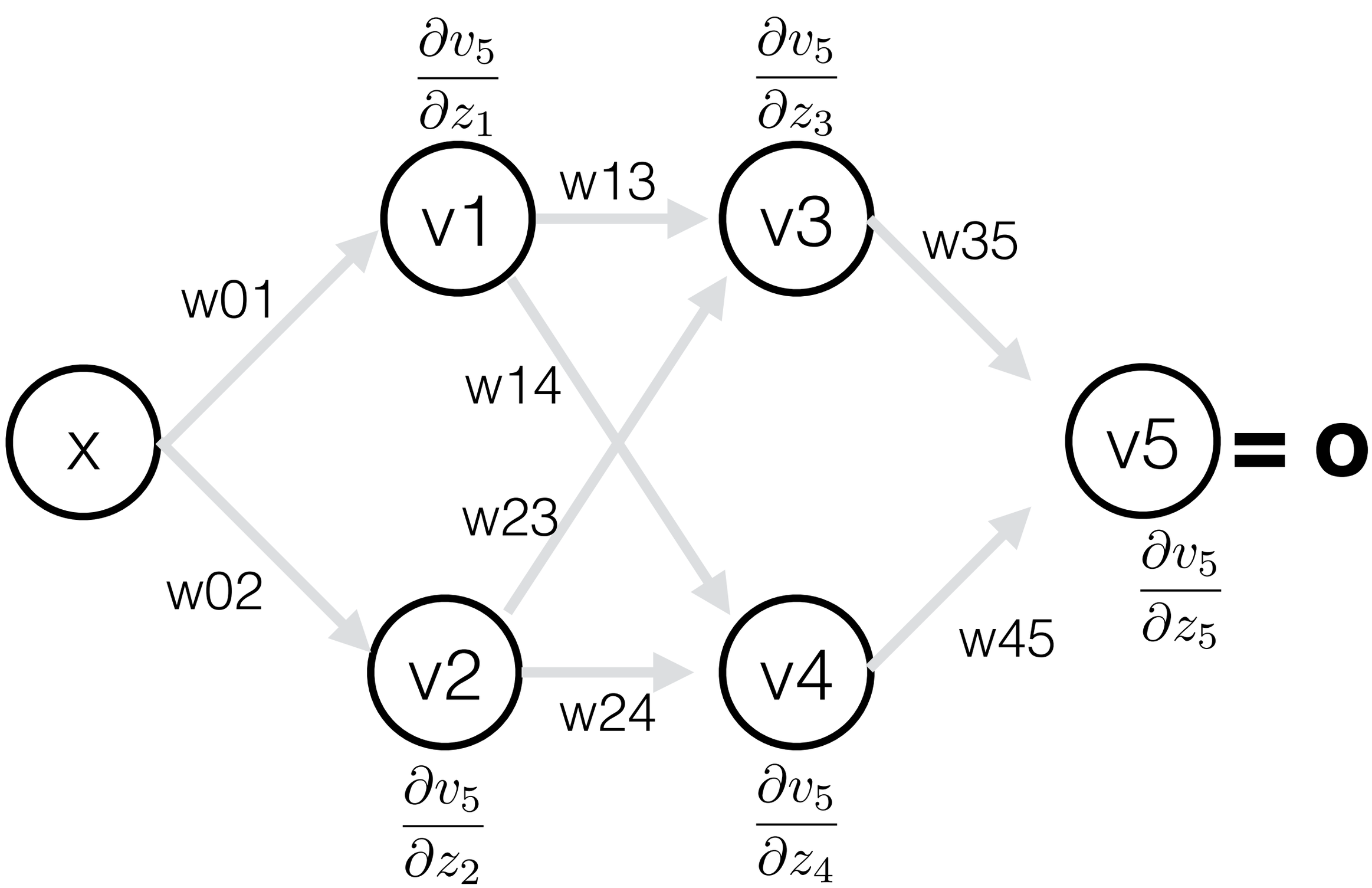
$$\frac{\partial h_3}{\partial b_3}, \frac{\partial h_3}{\partial b_2}, \frac{\partial h_3}{\partial b_1},$$




$$\frac{\partial v_5}{\partial z_4} = \frac{\partial v_5}{\partial z_5}\sigma'(z_4)w_{45}$$



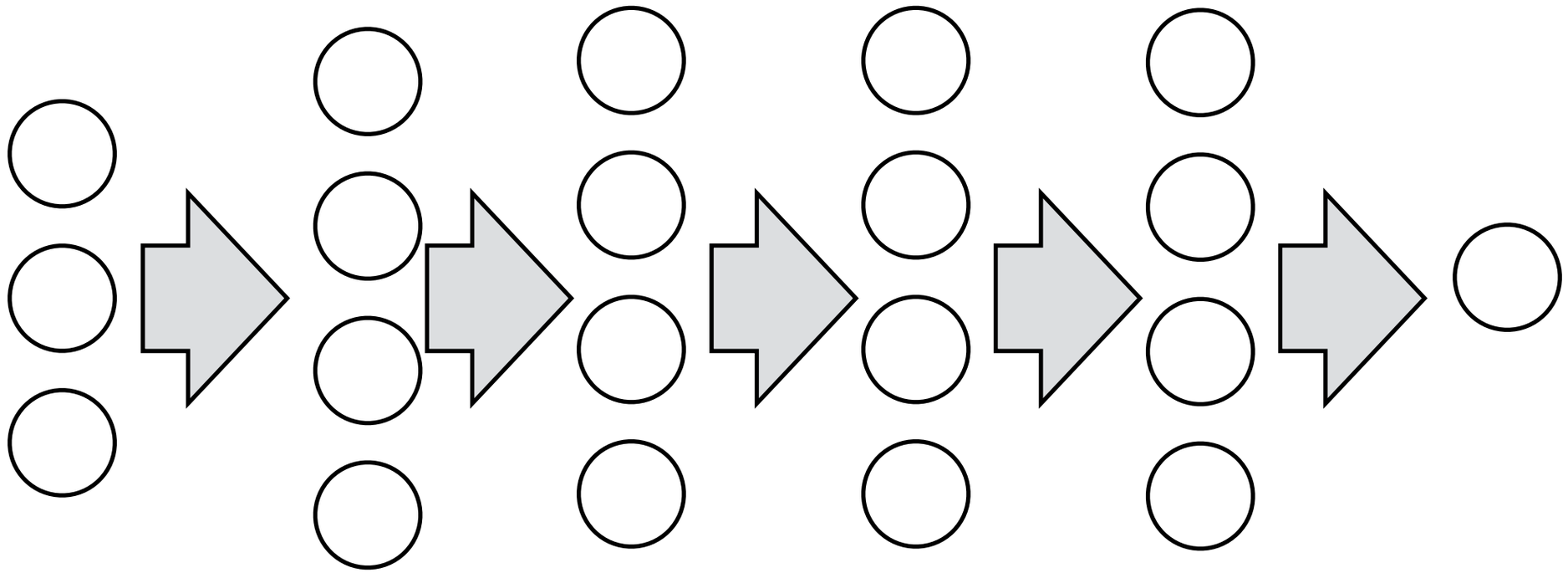

$$\frac{\partial v_5}{\partial z_2} = \frac{\partial v_5}{\partial z_4}\sigma'(z_2)w_{24} + \frac{\partial v_5}{\partial z_3}\sigma'(z_2)w_{23}$$

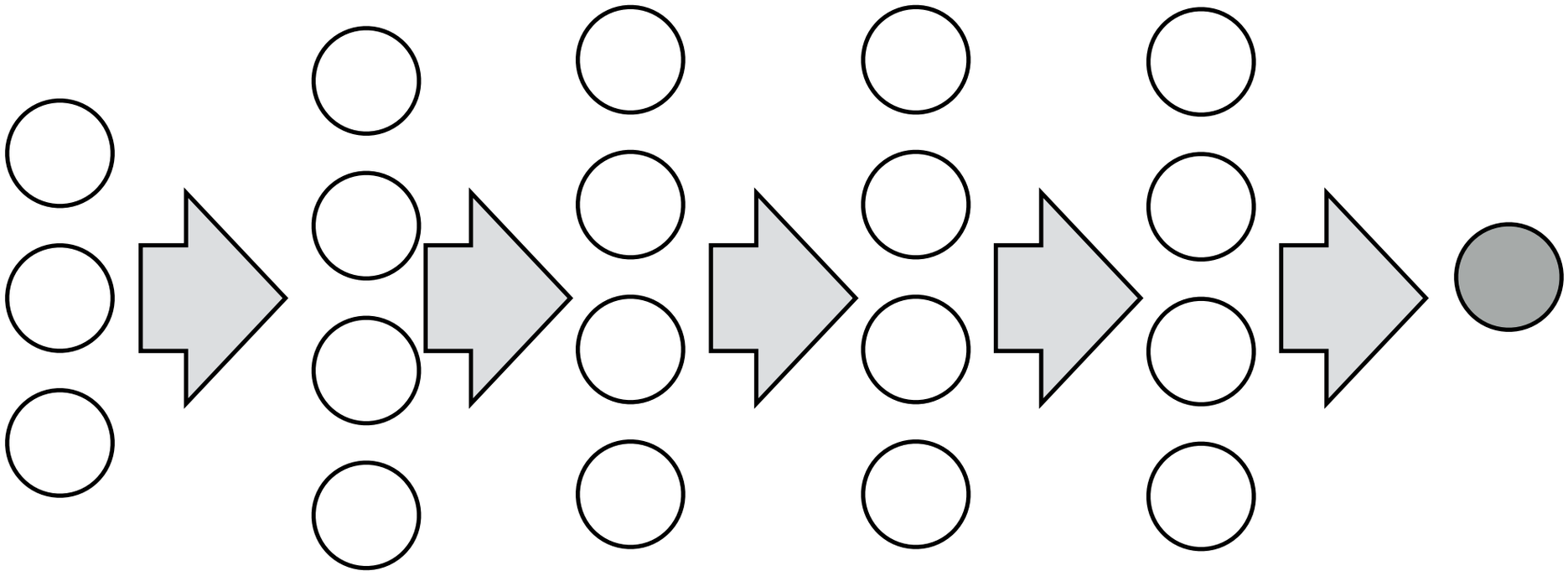


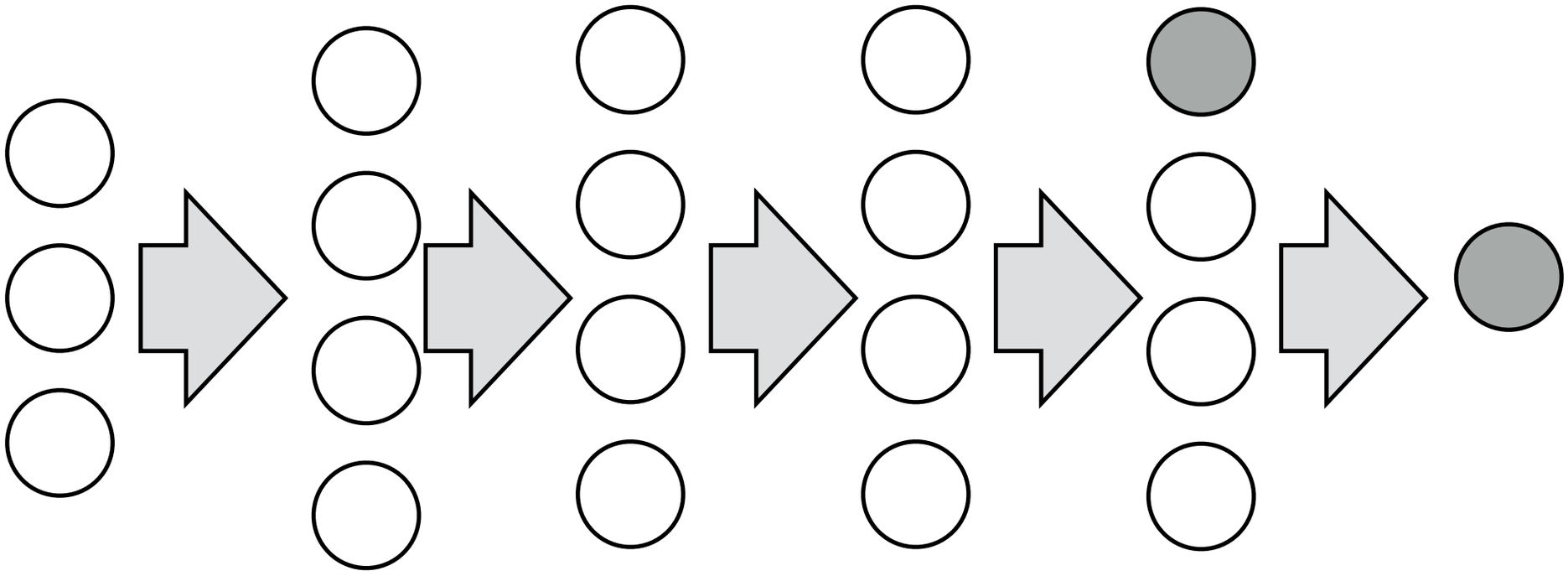

This is call the back propagation algorithm

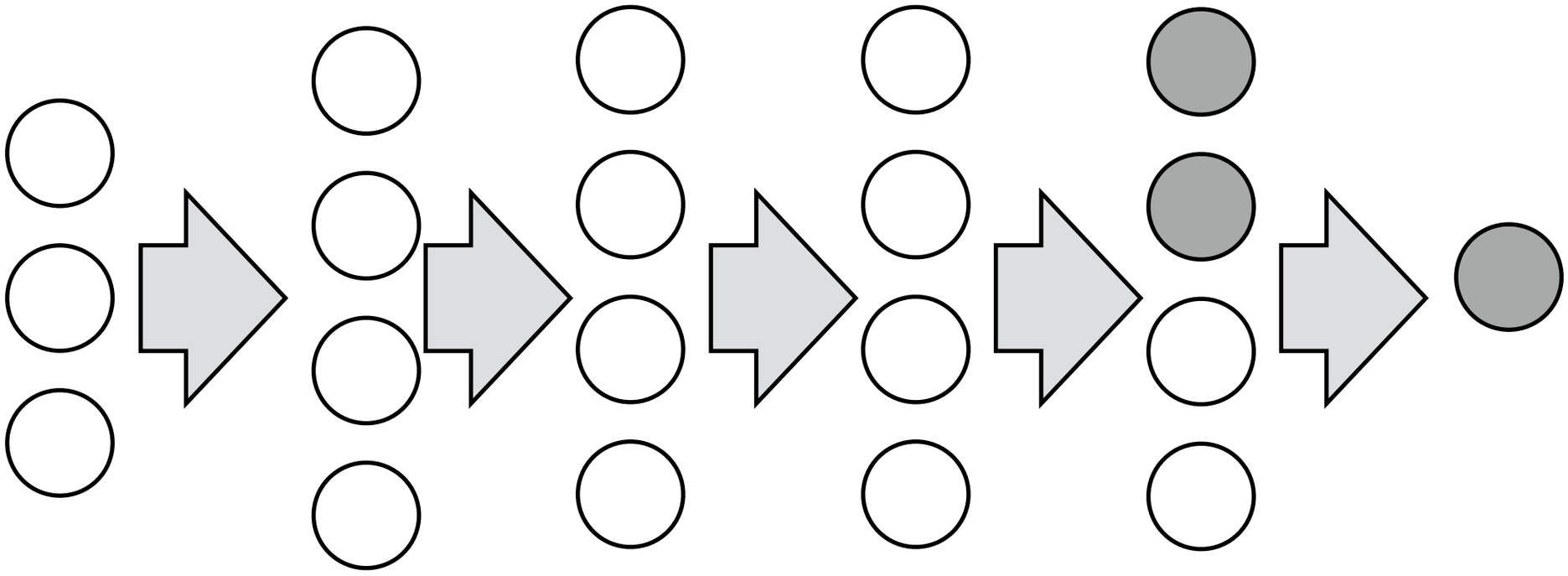
Have a look at:

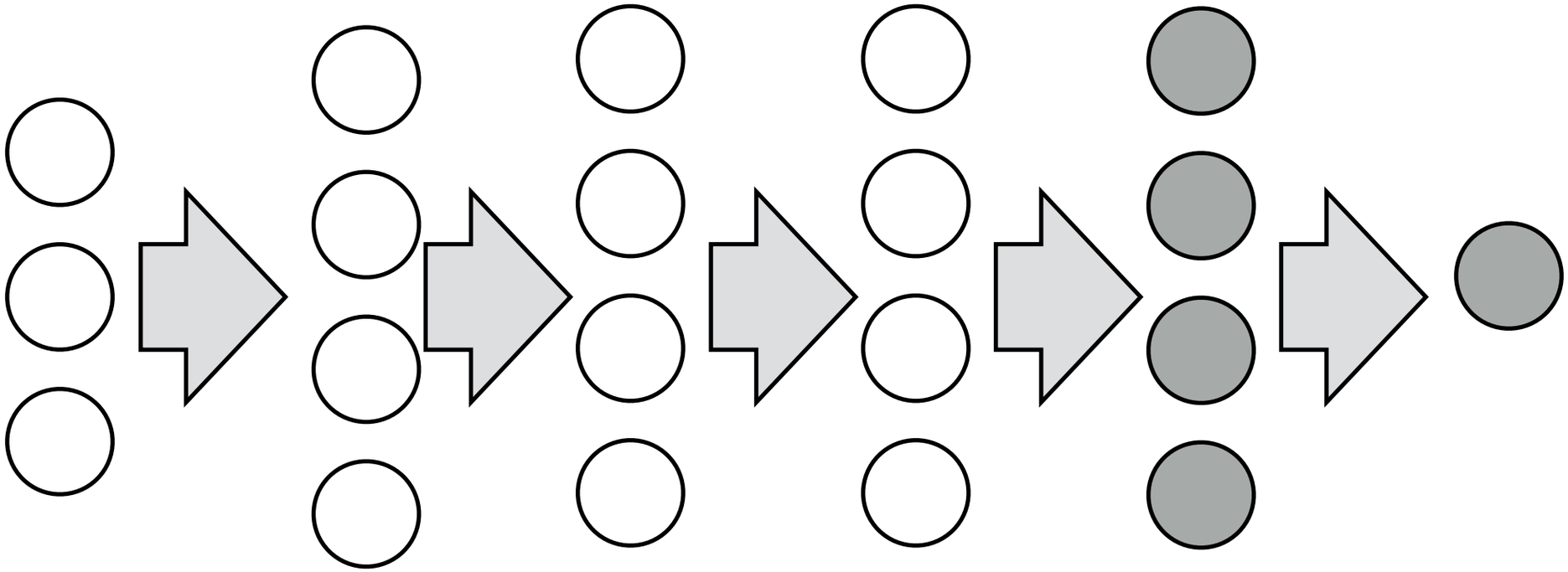
http://colah.github.io/posts/2015-08-Backprop/

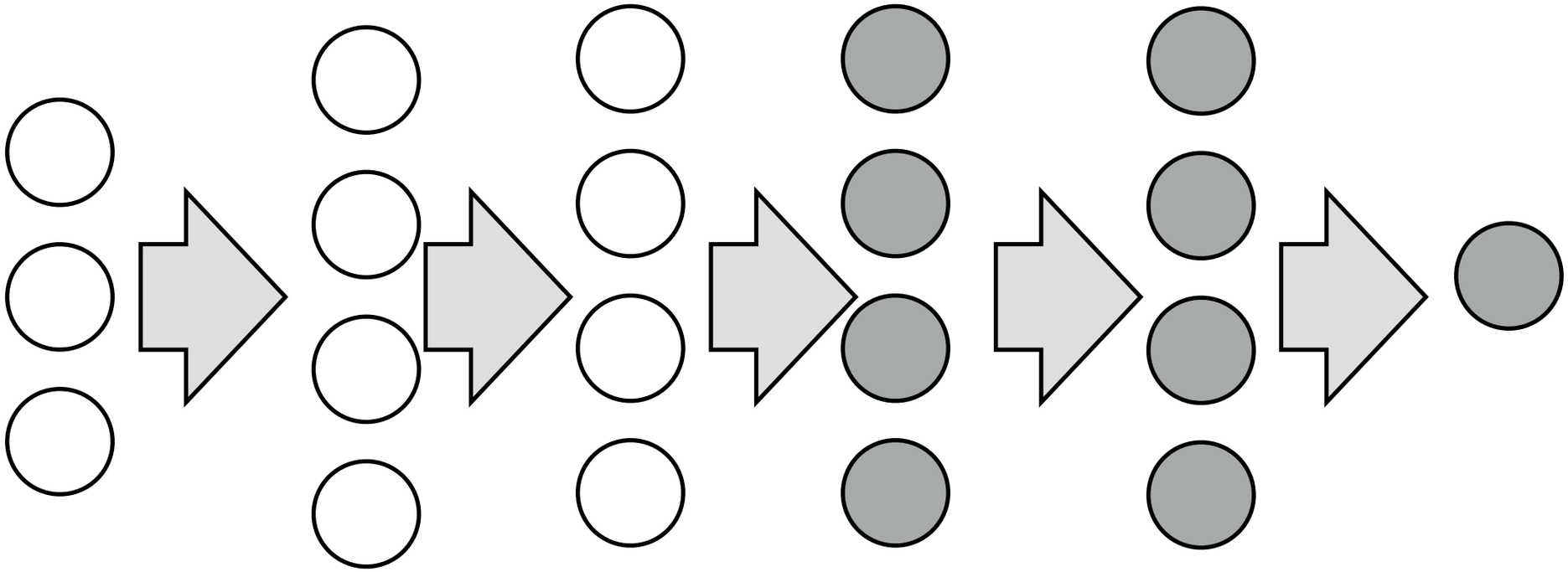


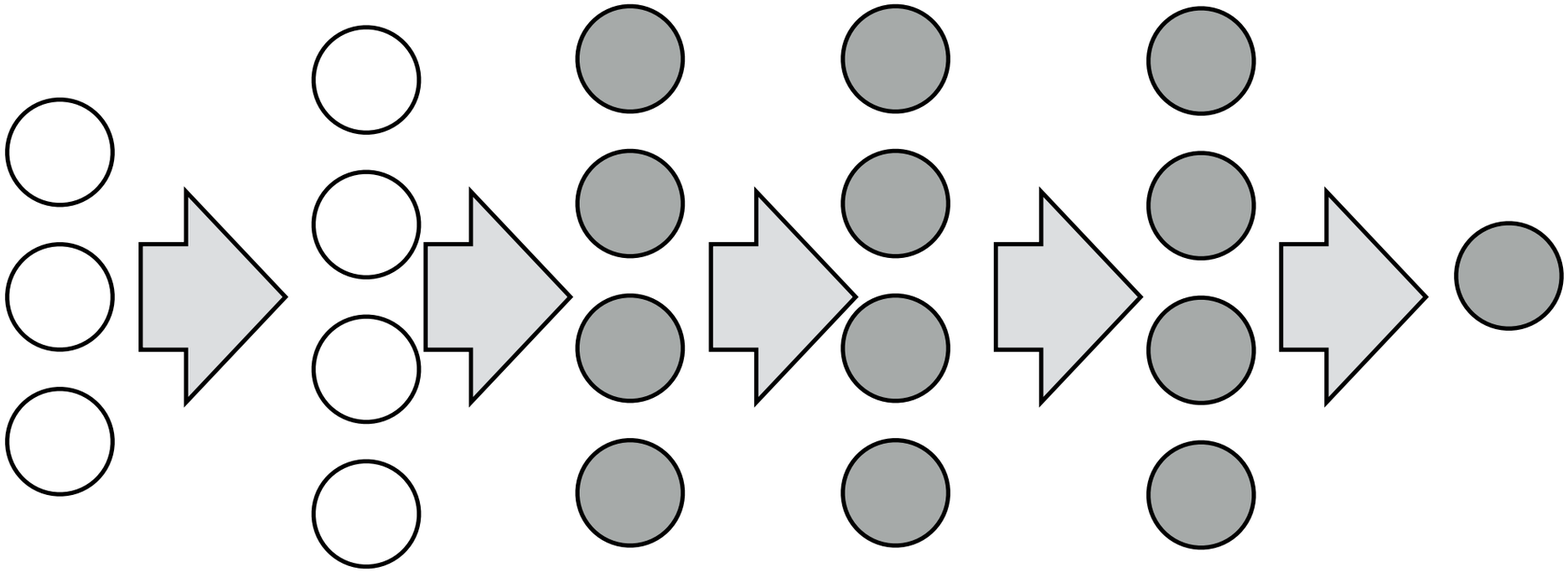


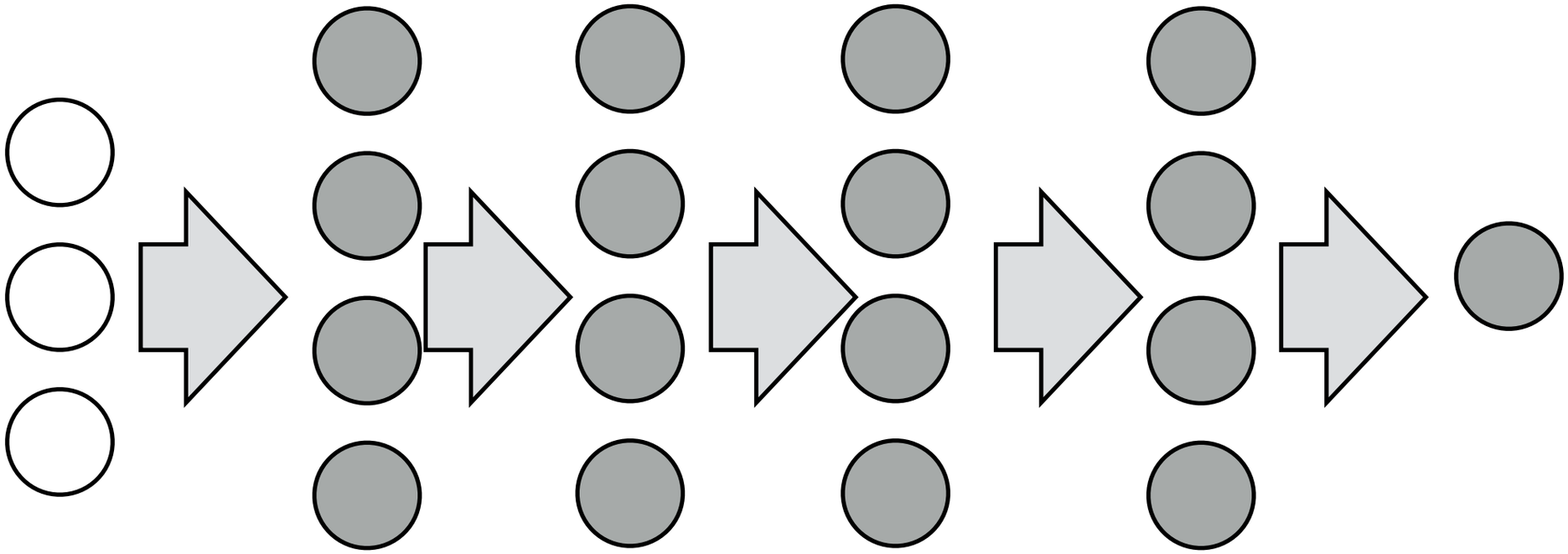












Pytorch implementation of computational graph

https://www.youtube.com/watch?v=syLFCVYua6Q

Watch until minute: 6:00