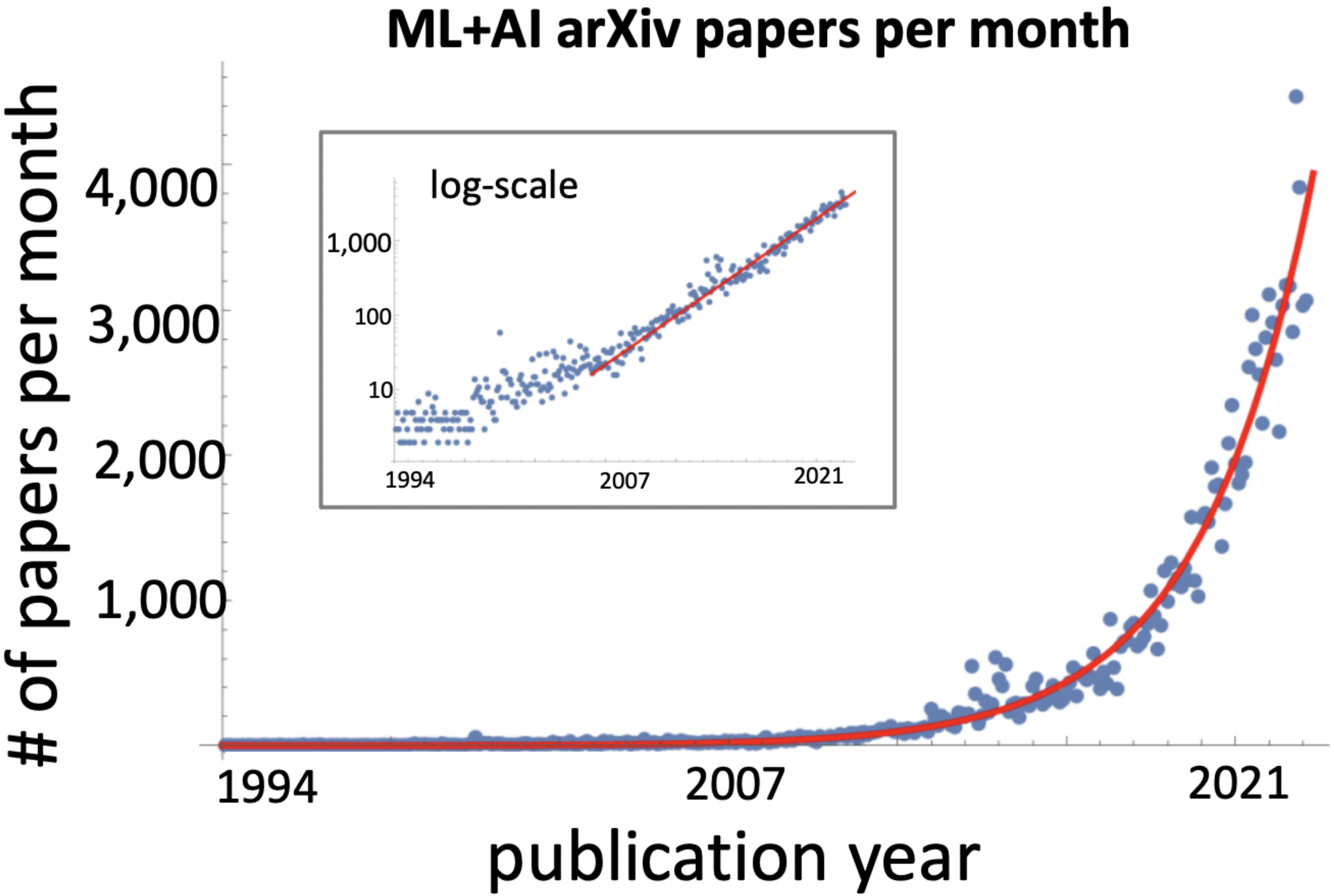
Shallow Networks

Abdul Samad

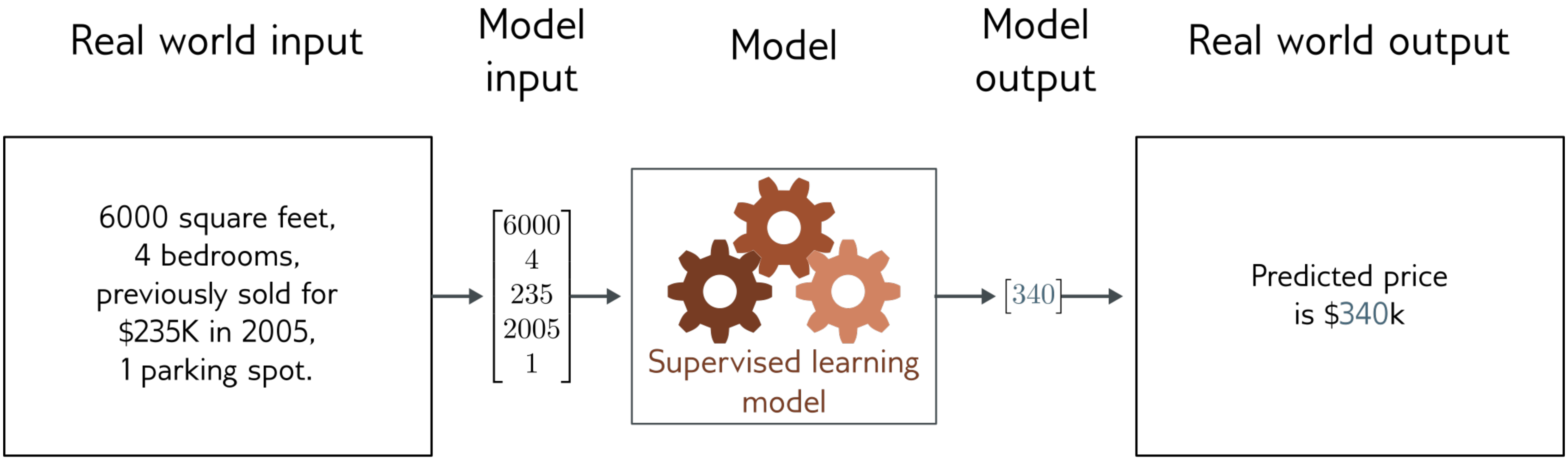
Adopted from Prof. Simon Prince


ML+AI arXiv papers per month
# of papers per month
4,000
3,000
2,000
1,000
1994
2007
2021
publication year
log-scale
1,000
100
10
1994
2007
2021

+ imagen video

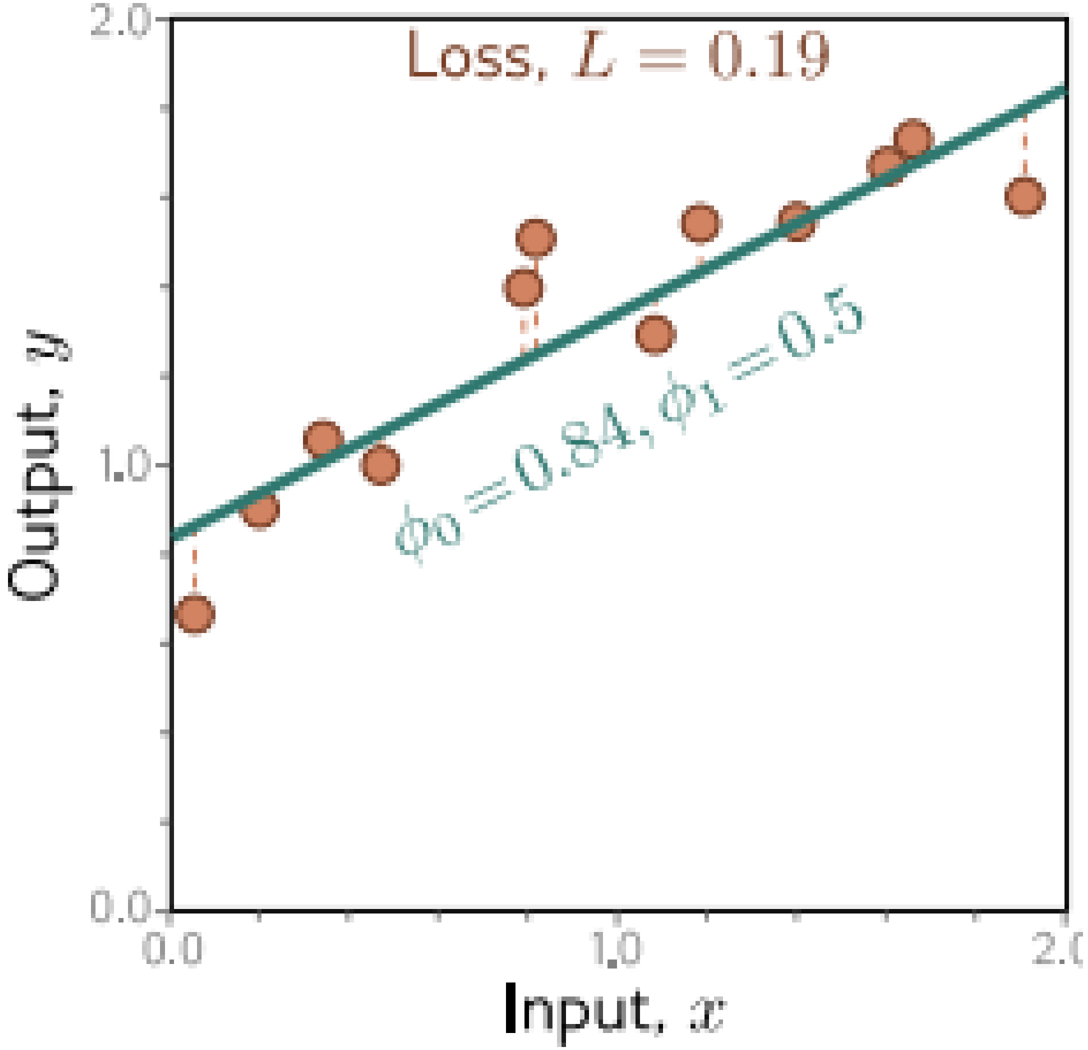
+ dream fusion

Regression



- Univariate regression problem (one output, real value)
- Fully connected network

Example: 1D Linear regression loss function



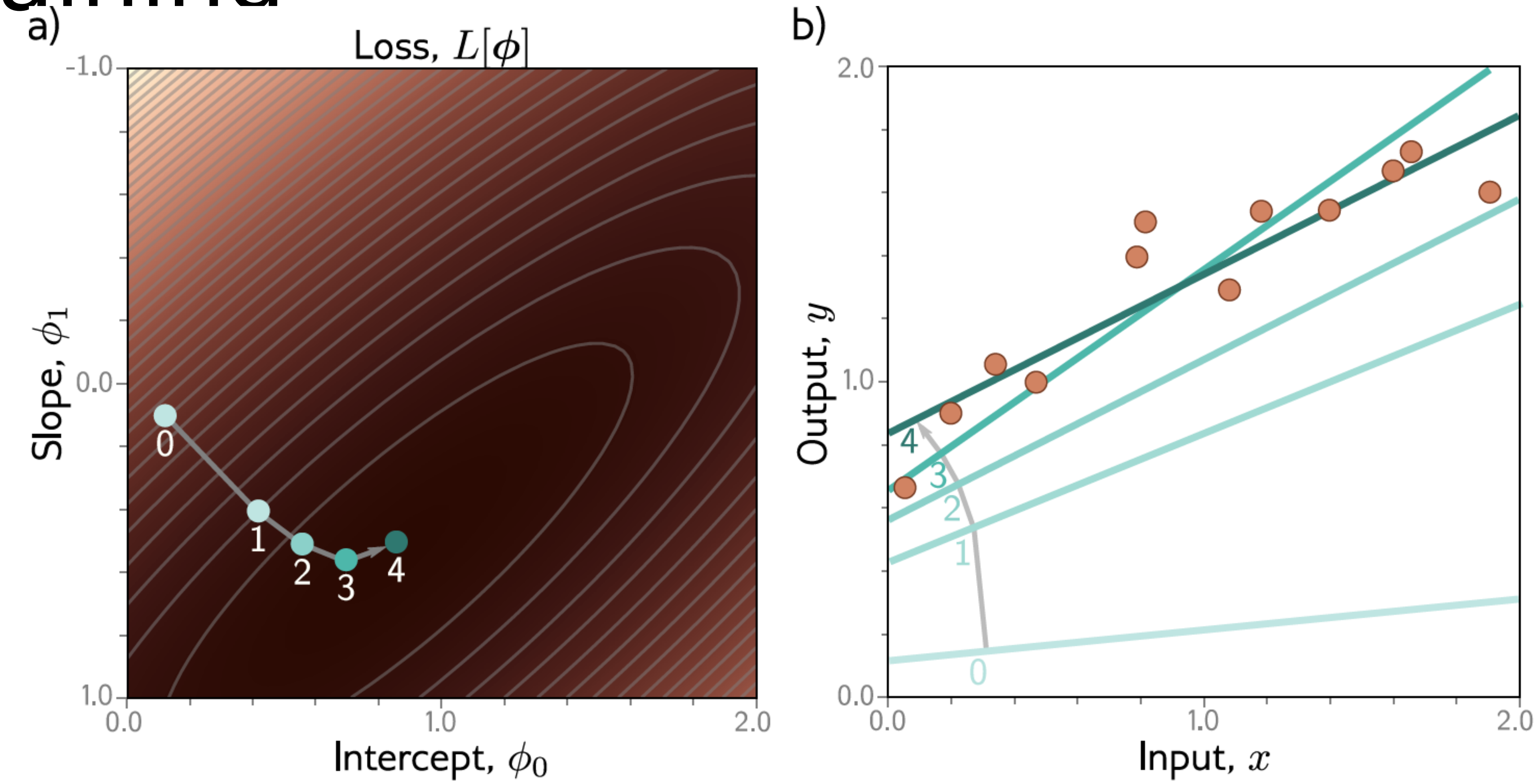
Loss function:

$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} (\mathrm{f}[x_i, \boldsymbol{\phi}] - y_i)^2$$

$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

Example: 1D Linear regression training



This technique is known as gradient descent

Shallow neural networks

- 1D regression model is obviously limited
  - Want to be able to describe input/output that are not lines
  - Want multiple inputs
  - Want multiple outputs

- Shallow neural networks
  - Flexible enough to describe arbitrarily complex input/output mappings
  - Can have as many inputs as we want
  - Can have as many outputs as we want

Shallow neural networks

- Example network, 1 input, 1 output
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

1D Linear Regression

$$\begin{aligned} y &= \mathrm{f}[x, \boldsymbol{\phi}] \\ &= \phi_0 + \phi_1 x \end{aligned}$$

Example shallow network

$$\begin{aligned} y &= \mathrm{f}[x, \boldsymbol{\phi}] \\ &= \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11} x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21} x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31} x] \end{aligned}$$

Example shallow network

$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11} x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21} x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31} x]
\end{aligned}
$$

Example shallow network

Activation function

$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11} x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21} x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31} x]
\end{aligned}
$$

Example shallow network

Activation function

$$y = \mathrm{f}[x, \phi]$$
$$= \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11} x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21} x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31} x]$$

$$\mathrm{a}[z] = \mathrm{ReLU}[z] = \begin{cases} 0 & z < 0 \\ z & z \geq 0 \end{cases}.$$

Rectified Linear Unit

(particular kind of activation function)

Example shallow network

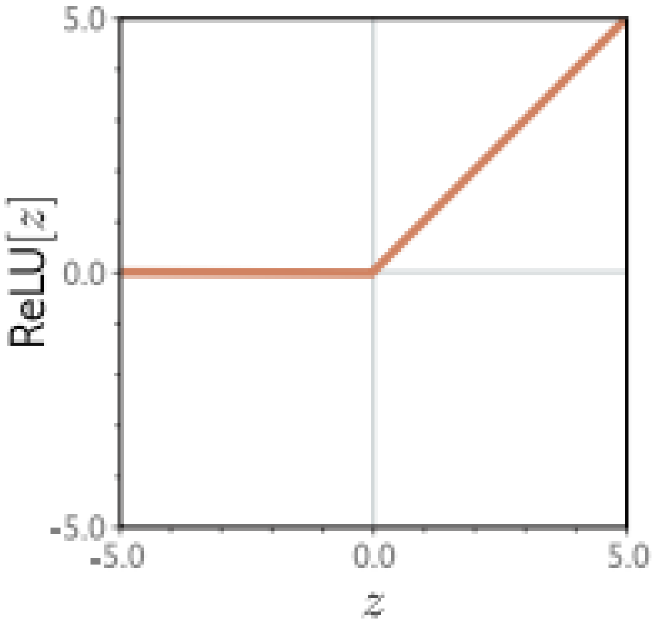
Activation function

$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11}x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21}x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31}x]
\end{aligned}
$$

$$
\mathrm{a}[z] = \mathrm{ReLU}[z] = \begin{cases} 0 & z < 0 \\ z & z \geq 0 \end{cases}.
$$

Rectified Linear Unit

(particular kind of activation function)



Example shallow network

$$
\begin{aligned}
y &= \mathrm{f}[x, \boldsymbol{\phi}] \\
&= \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11}x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21}x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31}x]
\end{aligned}
$$

This model has 10 parameters:

$$\boldsymbol{\phi} = \{\phi_0, \phi_1, \phi_2, \phi_3, \theta_{10}, \theta_{11}, \theta_{20}, \theta_{21}, \theta_{30}, \theta_{31}\}$$

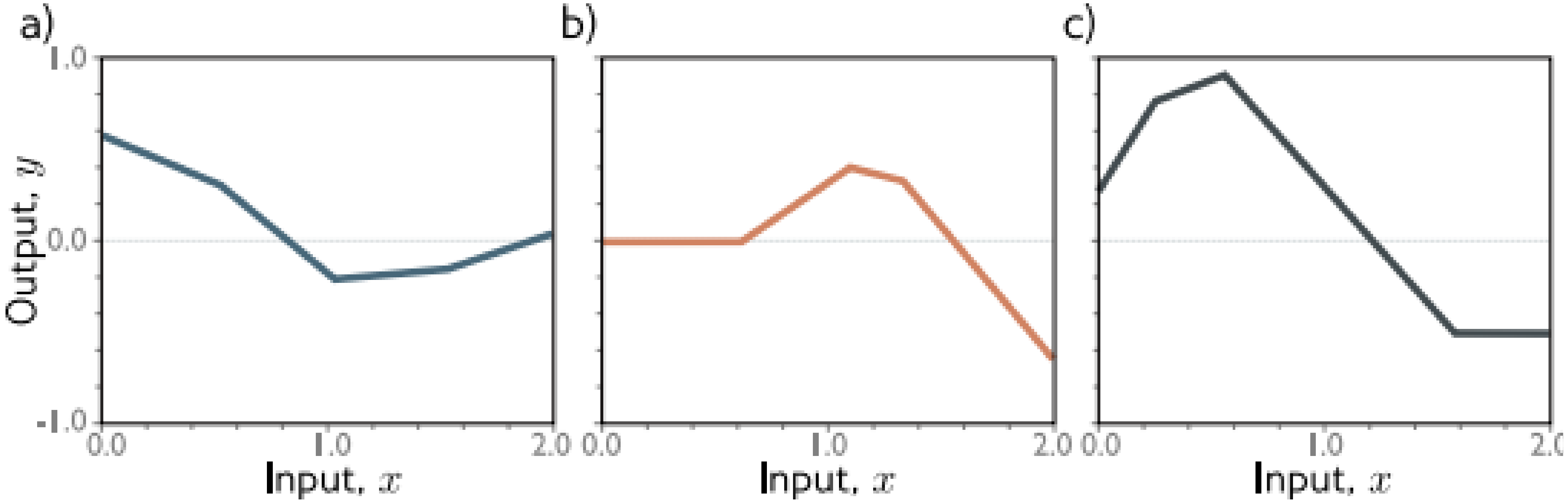
- Represents a family of functions
- Parameters determine particular function
- Given parameters can perform inference (run equation)
- Given training dataset $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^{I}$
- Define loss function $L[\boldsymbol{\phi}]$ (least squares)
- Change parameters to minimize loss function

Example shallow network

$$y = \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11}x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21}x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31}x].$$

Example shallow network

$$y = \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11}x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21}x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31}x].$$



Piecewise linear functions with three joints

Hidden units

$$y = \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11} x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21} x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31} x].$$

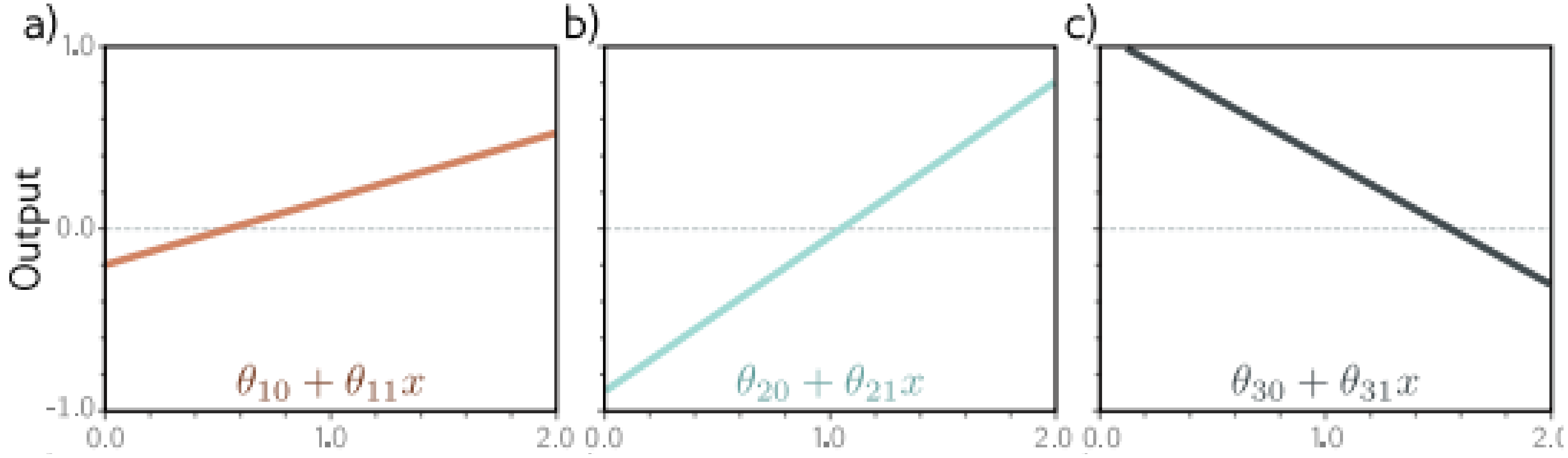
Break down into two parts:

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$

where:

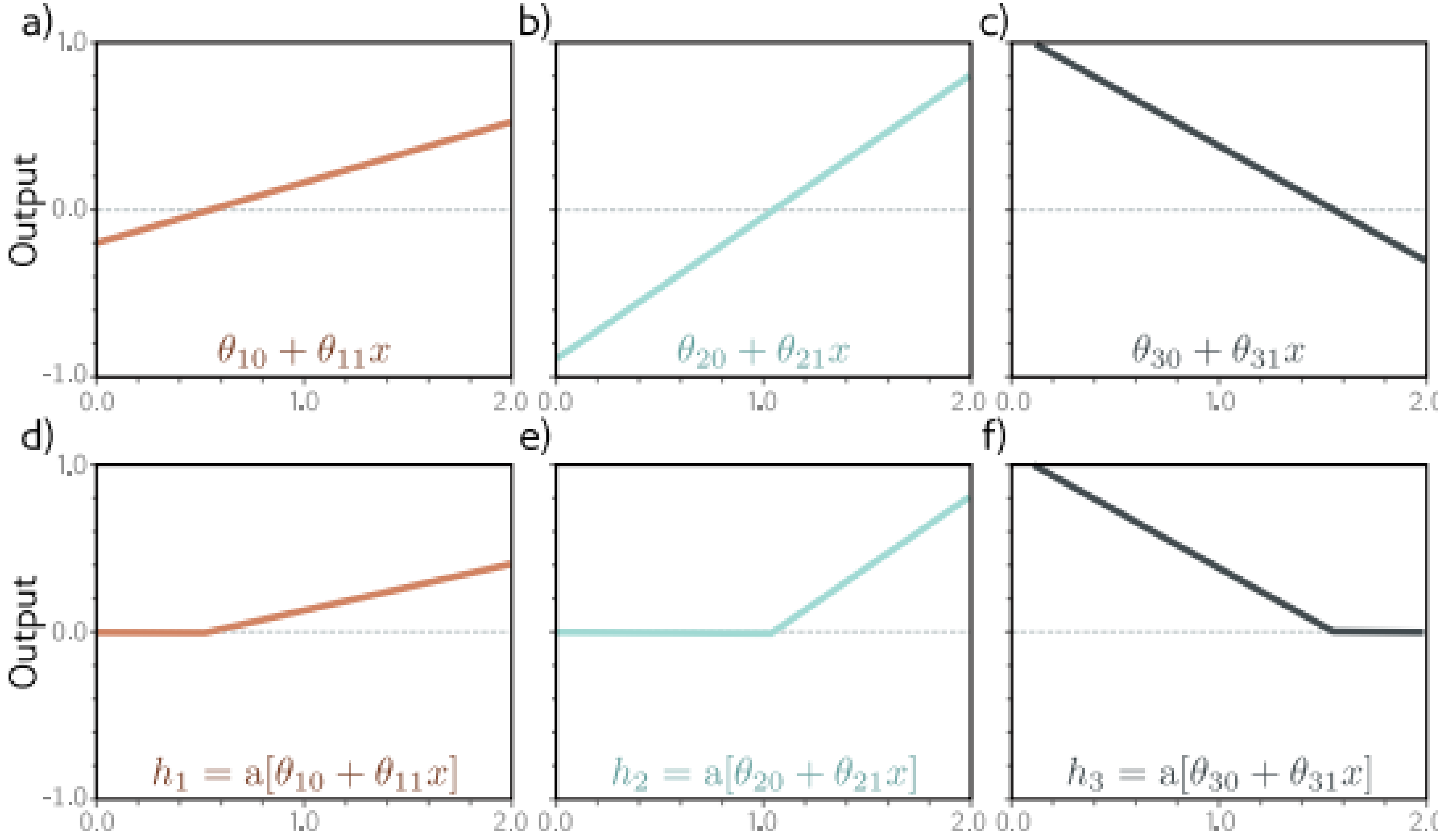
$$\text{Hidden units} \left\{ \begin{aligned} h_1 &= \mathrm{a}[\theta_{10} + \theta_{11} x] \\ h_2 &= \mathrm{a}[\theta_{20} + \theta_{21} x] \\ h_3 &= \mathrm{a}[\theta_{30} + \theta_{31} x] \end{aligned} \right.$$

1. compute three linear functions

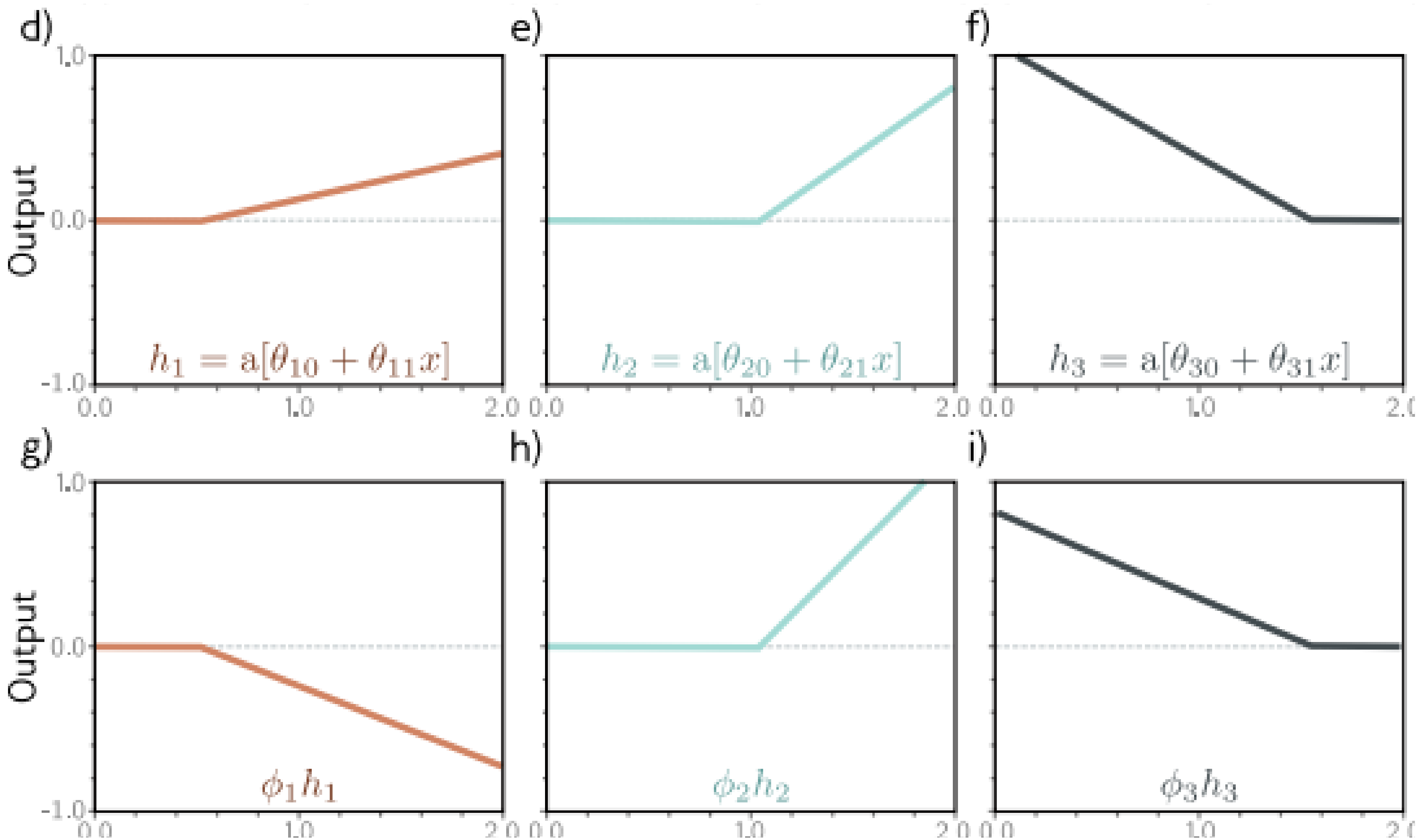


2. Weight the hidden units

$$h_1 = \mathrm{a}[\theta_{10} + \theta_{11}x]$$
$$h_2 = \mathrm{a}[\theta_{20} + \theta_{21}x]$$
$$h_3 = \mathrm{a}[\theta_{30} + \theta_{31}x],$$

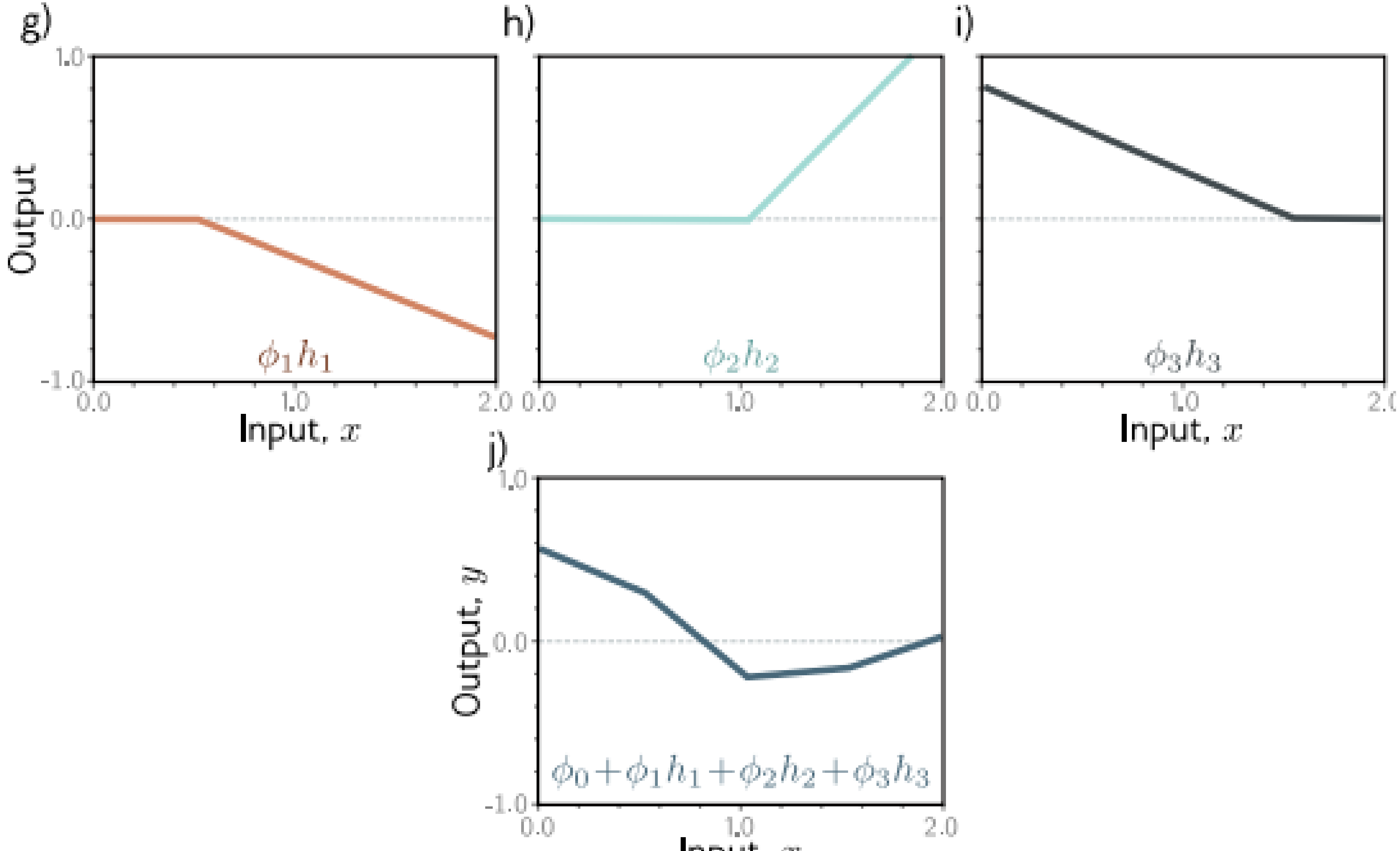


2. Pass through ReLU functions to compute hidden units



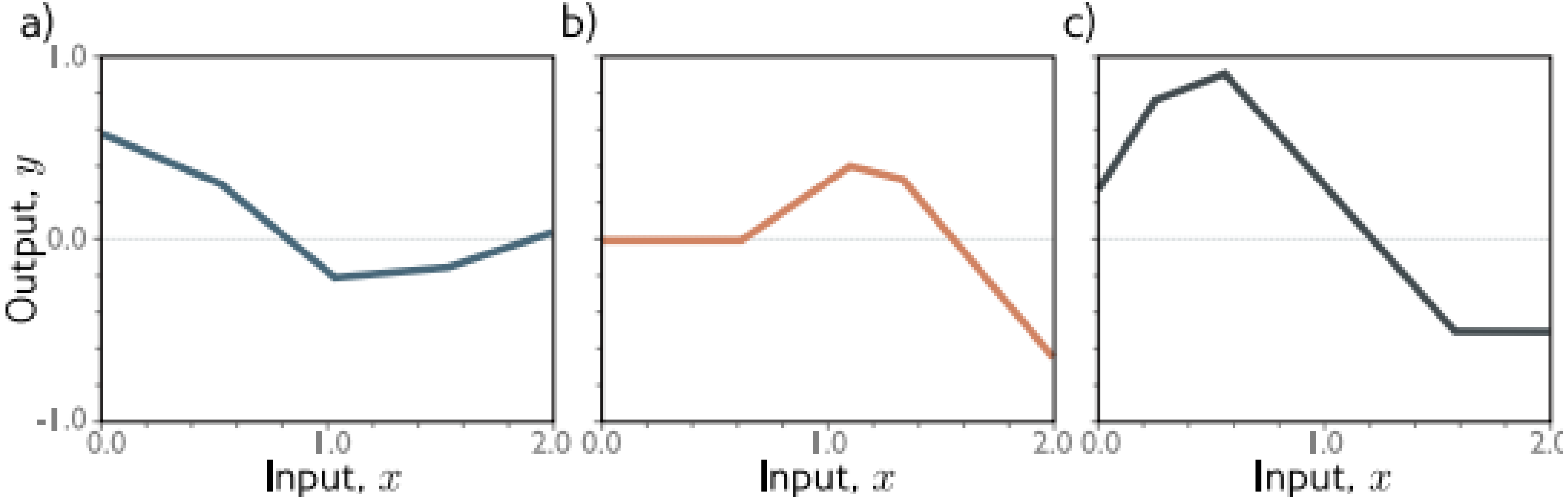
2. Pass through ReLU functions to compute hidden units

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$



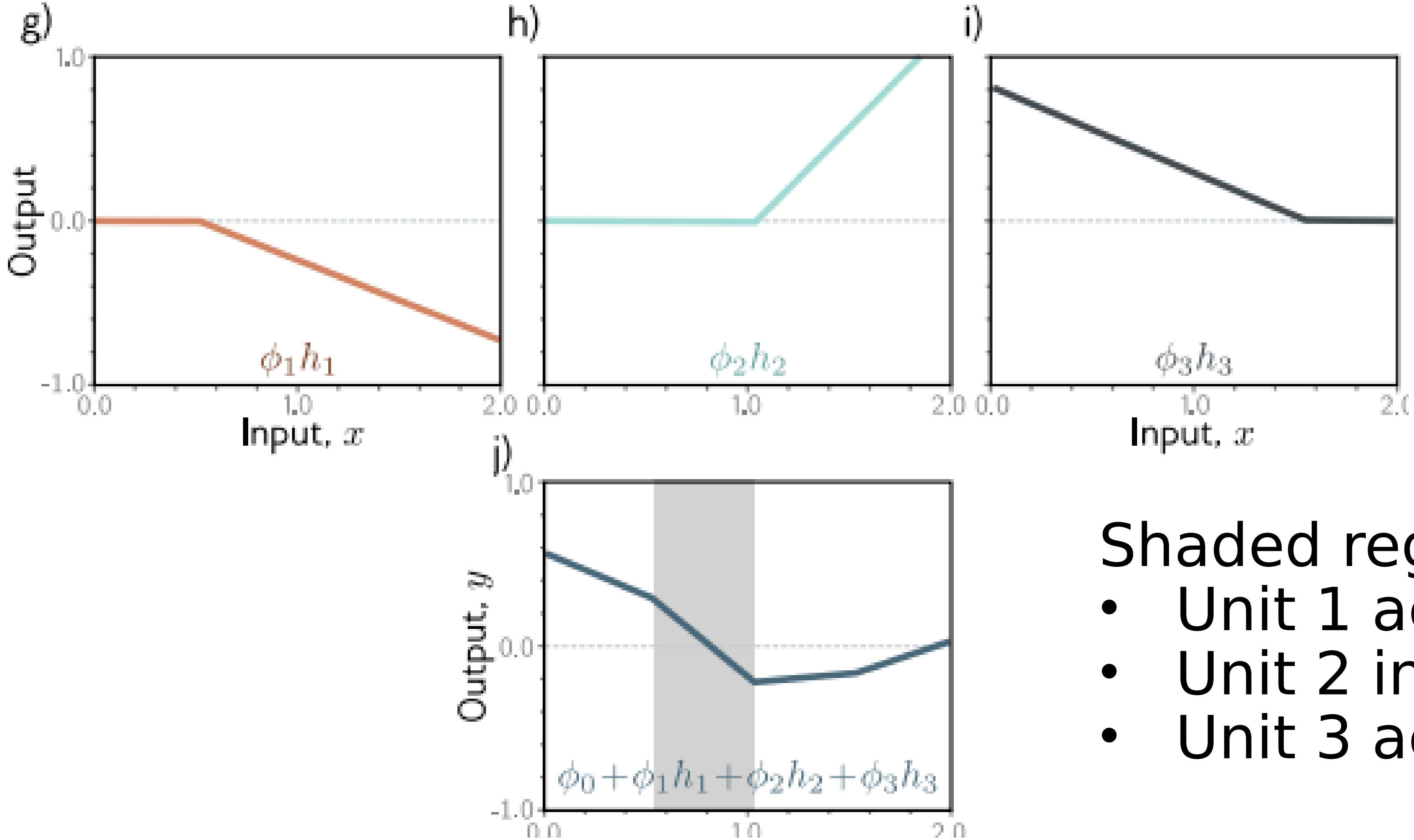
Example shallow network

$$y = \phi_0 + \phi_1 \mathrm{a}[\theta_{10} + \theta_{11}x] + \phi_2 \mathrm{a}[\theta_{20} + \theta_{21}x] + \phi_3 \mathrm{a}[\theta_{30} + \theta_{31}x].$$



Example shallow network = piecewise linear functions
1 “joint” per ReLU function

Activation pattern = which hidden units are activated



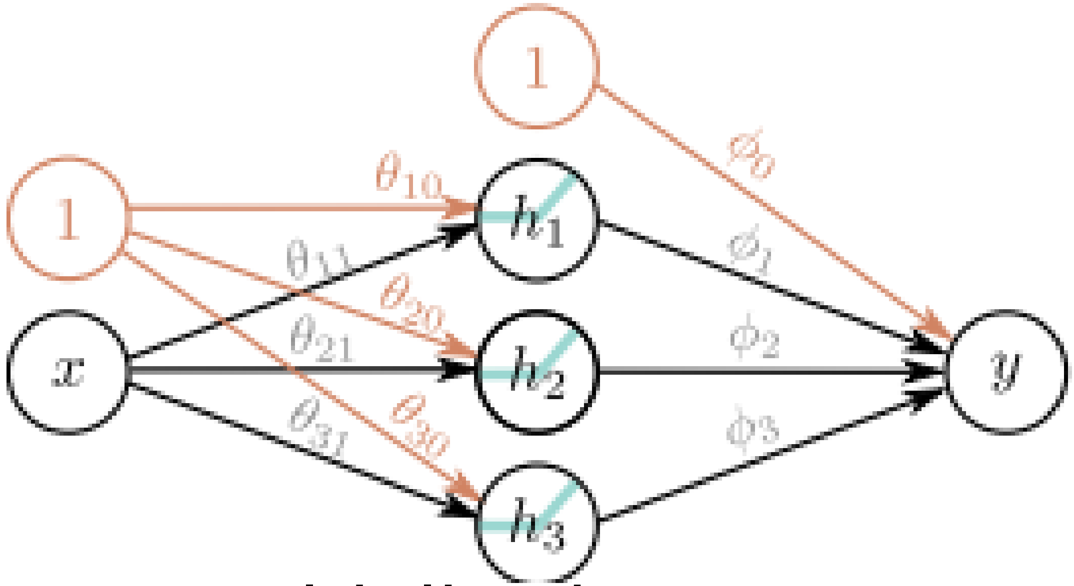
Shaded region:

- Unit 1 active
- Unit 2 inactive
- Unit 3 active

Depicting neural networks

$$h_1 = \mathrm{a}[\theta_{10} + \theta_{11}x]$$
$$h_2 = \mathrm{a}[\theta_{20} + \theta_{21}x]$$
$$h_3 = \mathrm{a}[\theta_{30} + \theta_{31}x]$$

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$

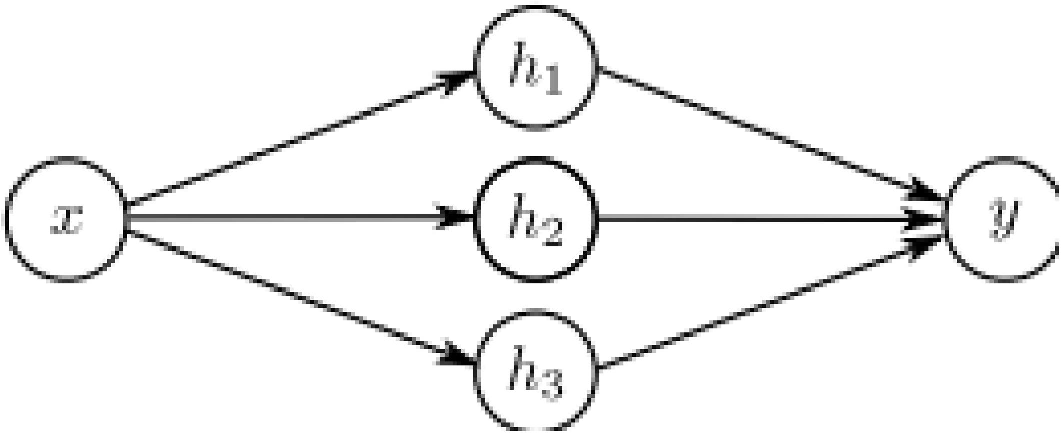


Each parameter multiplies its source and adds to its target

Depicting neural networks

$$h_1 = \mathrm{a}[\theta_{10} + \theta_{11}x]$$
$$h_2 = \mathrm{a}[\theta_{20} + \theta_{21}x]$$
$$h_3 = \mathrm{a}[\theta_{30} + \theta_{31}x]$$

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$



Shallow neural networks

- Example network, 1 input, 1 output
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

With 3 hidden units:

$$h_1 = \mathrm{a}[\theta_{10} + \theta_{11}x]$$
$$h_2 = \mathrm{a}[\theta_{20} + \theta_{21}x]$$
$$h_3 = \mathrm{a}[\theta_{30} + \theta_{31}x]$$

$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$

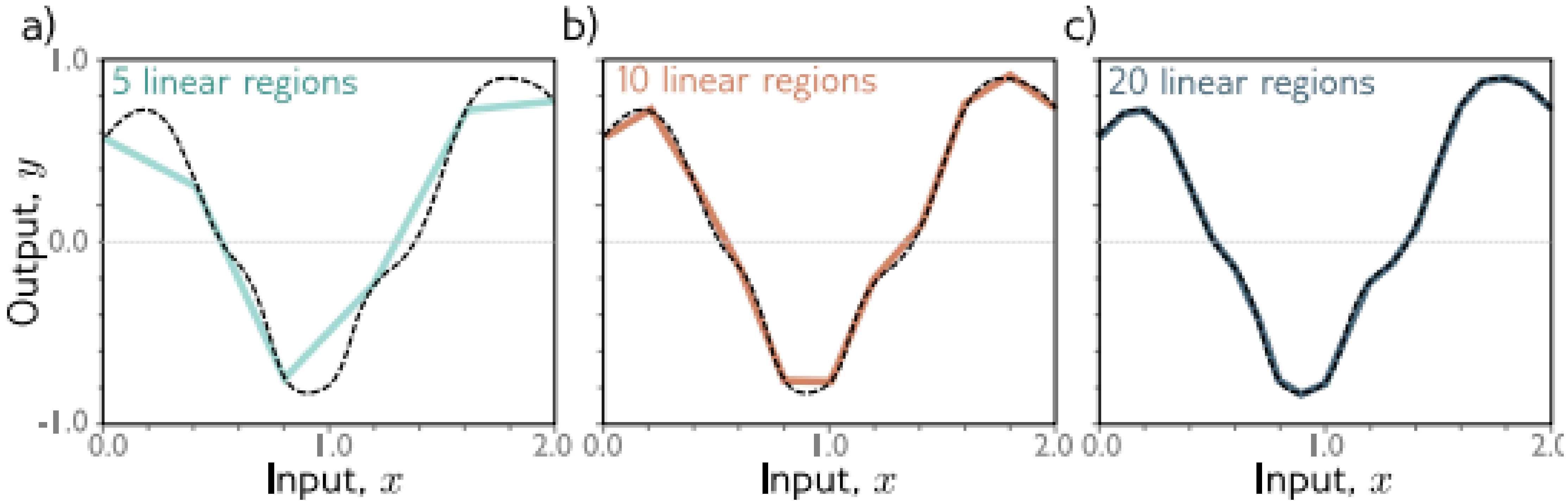
With D hidden units:

$$h_d = \mathrm{a}[\theta_{d0} + \theta_{d1}x]$$

$$y = \phi_0 + \sum_{d=1}^{D} \phi_d h_d$$

With enough hidden units…

… we can describe any 1D function to arbitrary accuracy



Universal approximation theorem

“a formal proof that, with enough hidden units, a shallow neural network can describe any continuous function on a compact subset of $\mathbb{R}^D$ to arbitrary precision”

Shallow neural networks

- Example network, 1 input, 1 ouput
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

Two outputs

- 1 input, 4 hidden units, 2 outputs

$$
\begin{aligned}
h_1 &= \mathrm{a}[\theta_{10} + \theta_{11}x] \\
h_2 &= \mathrm{a}[\theta_{20} + \theta_{21}x] \\
h_3 &= \mathrm{a}[\theta_{30} + \theta_{31}x] \\
h_4 &= \mathrm{a}[\theta_{40} + \theta_{41}x]
\end{aligned}
$$

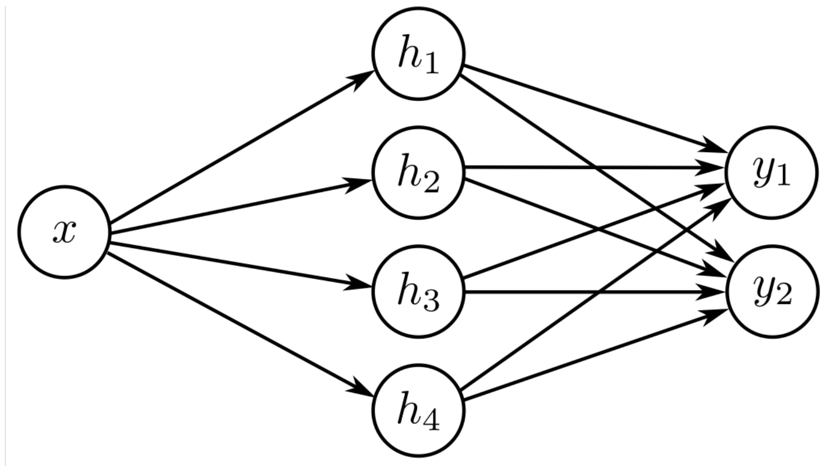
$$
\begin{aligned}
y_1 &= \phi_{10} + \phi_{11}h_1 + \phi_{12}h_2 + \phi_{13}h_3 + \phi_{14}h_4 \\
y_2 &= \phi_{20} + \phi_{21}h_1 + \phi_{22}h_2 + \phi_{23}h_3 + \phi_{24}h_4
\end{aligned}
$$

Two outputs

- 1 input, 4 hidden units, 2 outputs

$$
\begin{aligned}
h_1 &= \mathrm{a}[\theta_{10} + \theta_{11}x] \\
h_2 &= \mathrm{a}[\theta_{20} + \theta_{21}x] \\
h_3 &= \mathrm{a}[\theta_{30} + \theta_{31}x] \\
h_4 &= \mathrm{a}[\theta_{40} + \theta_{41}x]
\end{aligned}
$$

$$
\begin{aligned}
y_1 &= \phi_{10} + \phi_{11}h_1 + \phi_{12}h_2 + \phi_{13}h_3 + \phi_{14}h_4 \\
y_2 &= \phi_{20} + \phi_{21}h_1 + \phi_{22}h_2 + \phi_{23}h_3 + \phi_{24}h_4
\end{aligned}
$$

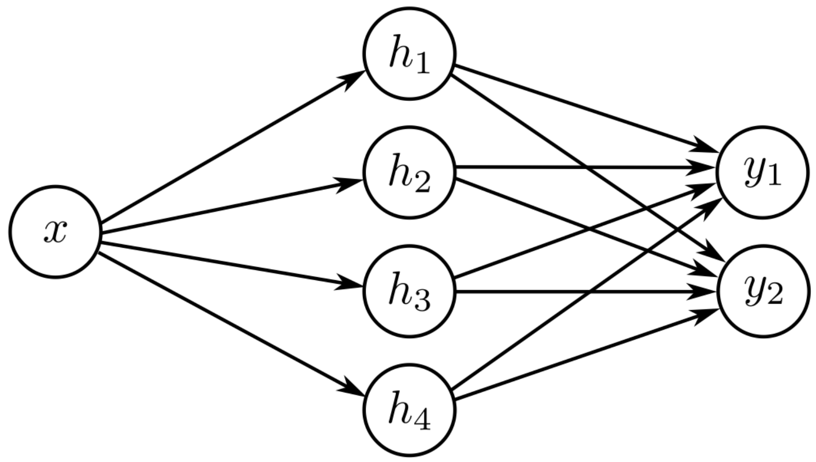


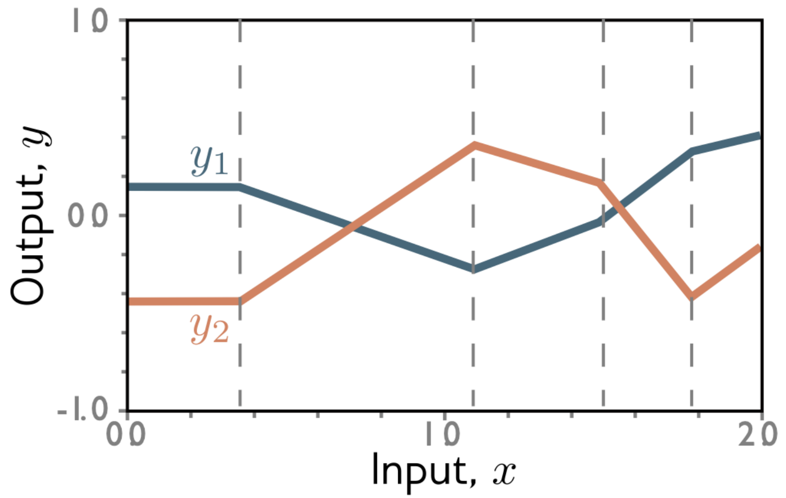
Two outputs

- 1 input, 4 hidden units, 2 outputs

$$
\begin{aligned}
h_1 &= \mathrm{a}[\theta_{10} + \theta_{11}x] \\
h_2 &= \mathrm{a}[\theta_{20} + \theta_{21}x] \\
h_3 &= \mathrm{a}[\theta_{30} + \theta_{31}x] \\
h_4 &= \mathrm{a}[\theta_{40} + \theta_{41}x]
\end{aligned}
$$

$$
\begin{aligned}
y_1 &= \phi_{10} + \phi_{11}h_1 + \phi_{12}h_2 + \phi_{13}h_3 + \phi_{14}h_4 \\
y_2 &= \phi_{20} + \phi_{21}h_1 + \phi_{22}h_2 + \phi_{23}h_3 + \phi_{24}h_4
\end{aligned}
$$





Shallow neural networks

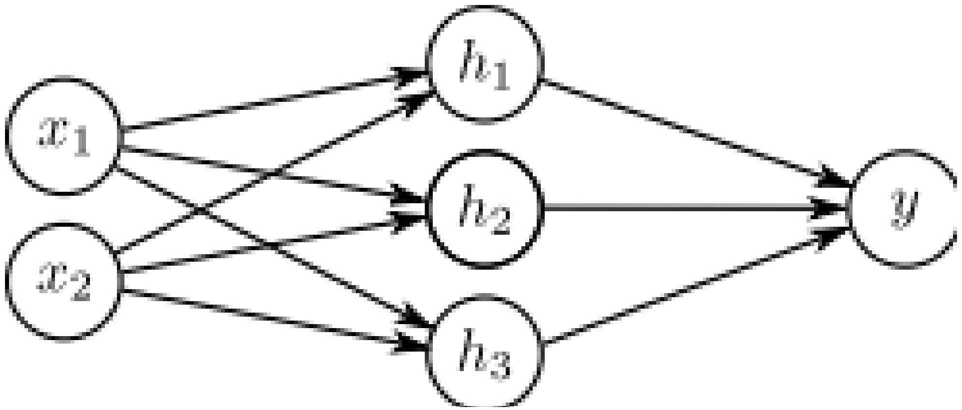
- Example network, 1 input, 1 ouput
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

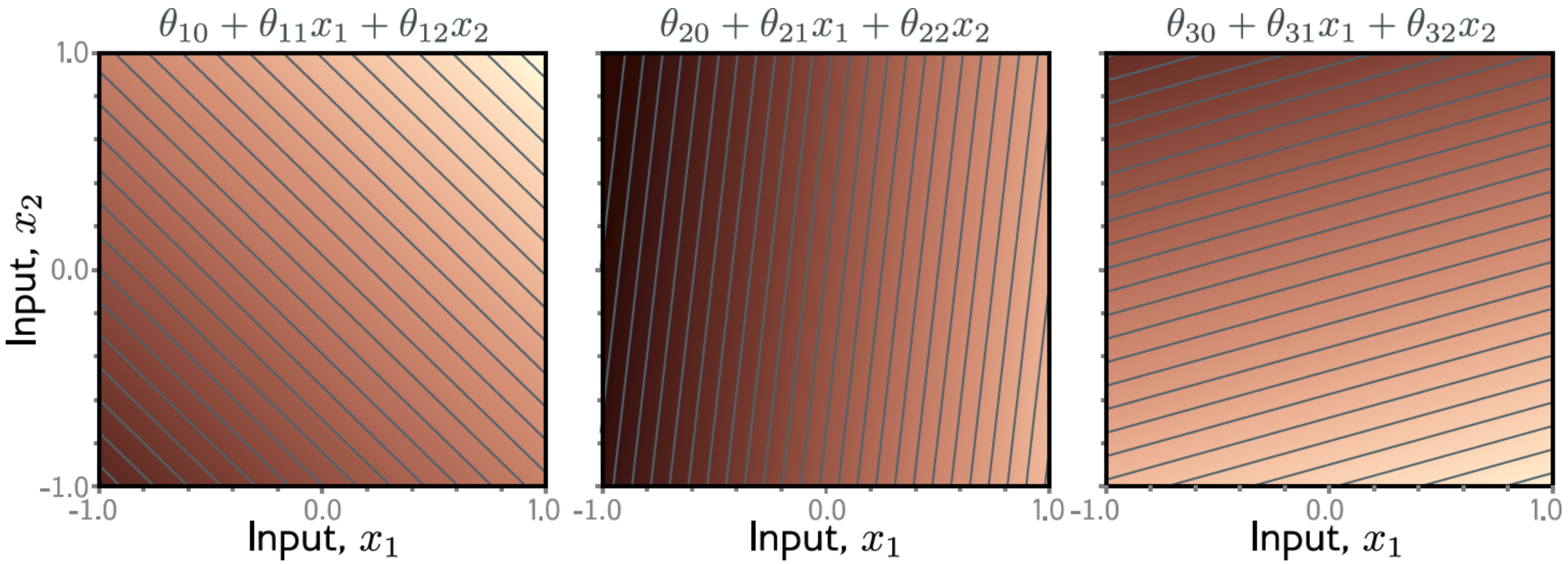
Two inputs

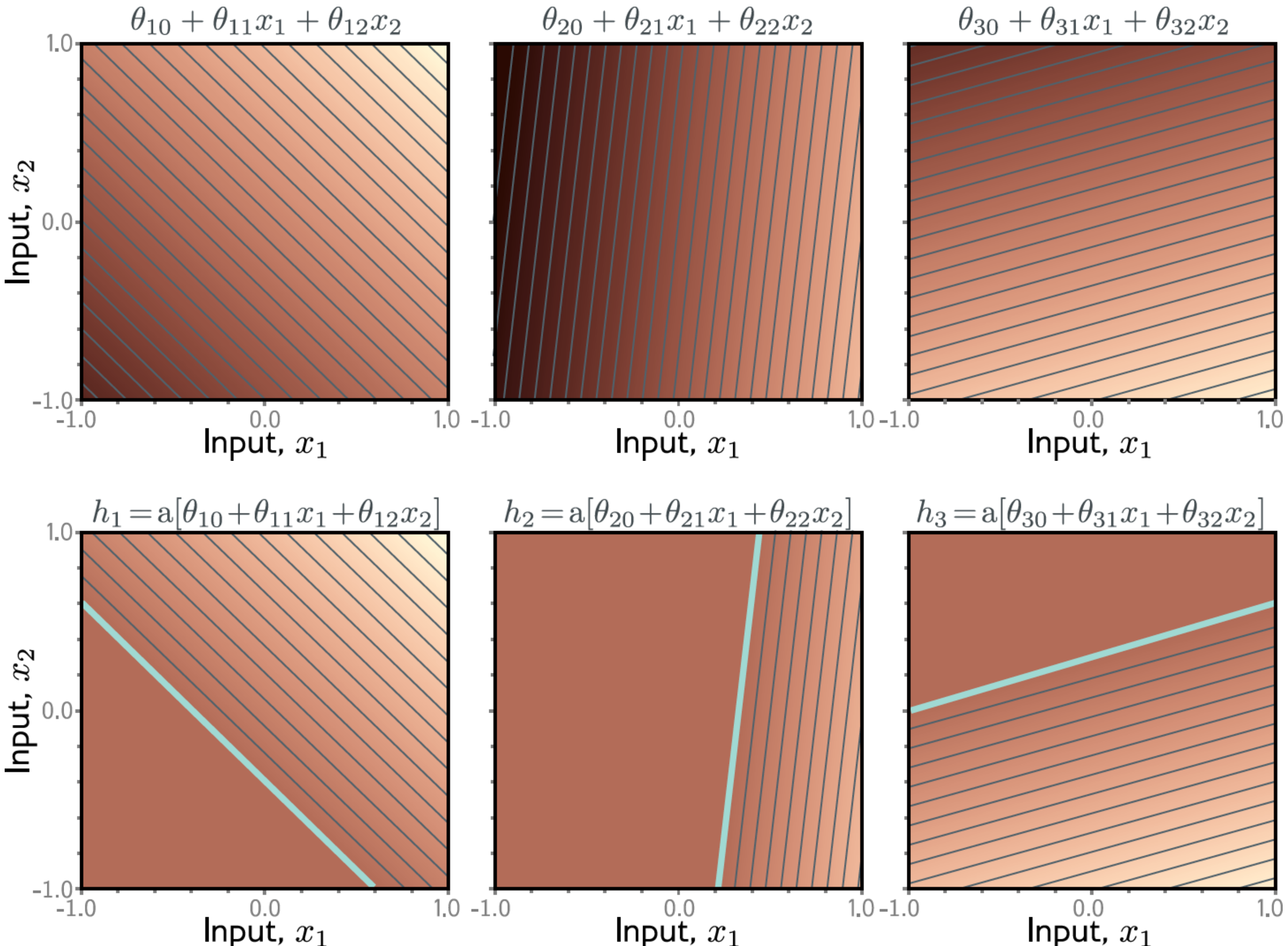
- 2 inputs, 3 hidden units, 1 output

$$
\begin{aligned}
h_1 &= \mathrm{a}[\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2] \\
h_2 &= \mathrm{a}[\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2] \\
h_3 &= \mathrm{a}[\theta_{30} + \theta_{31}x_1 + \theta_{32}x_2]
\end{aligned}
$$

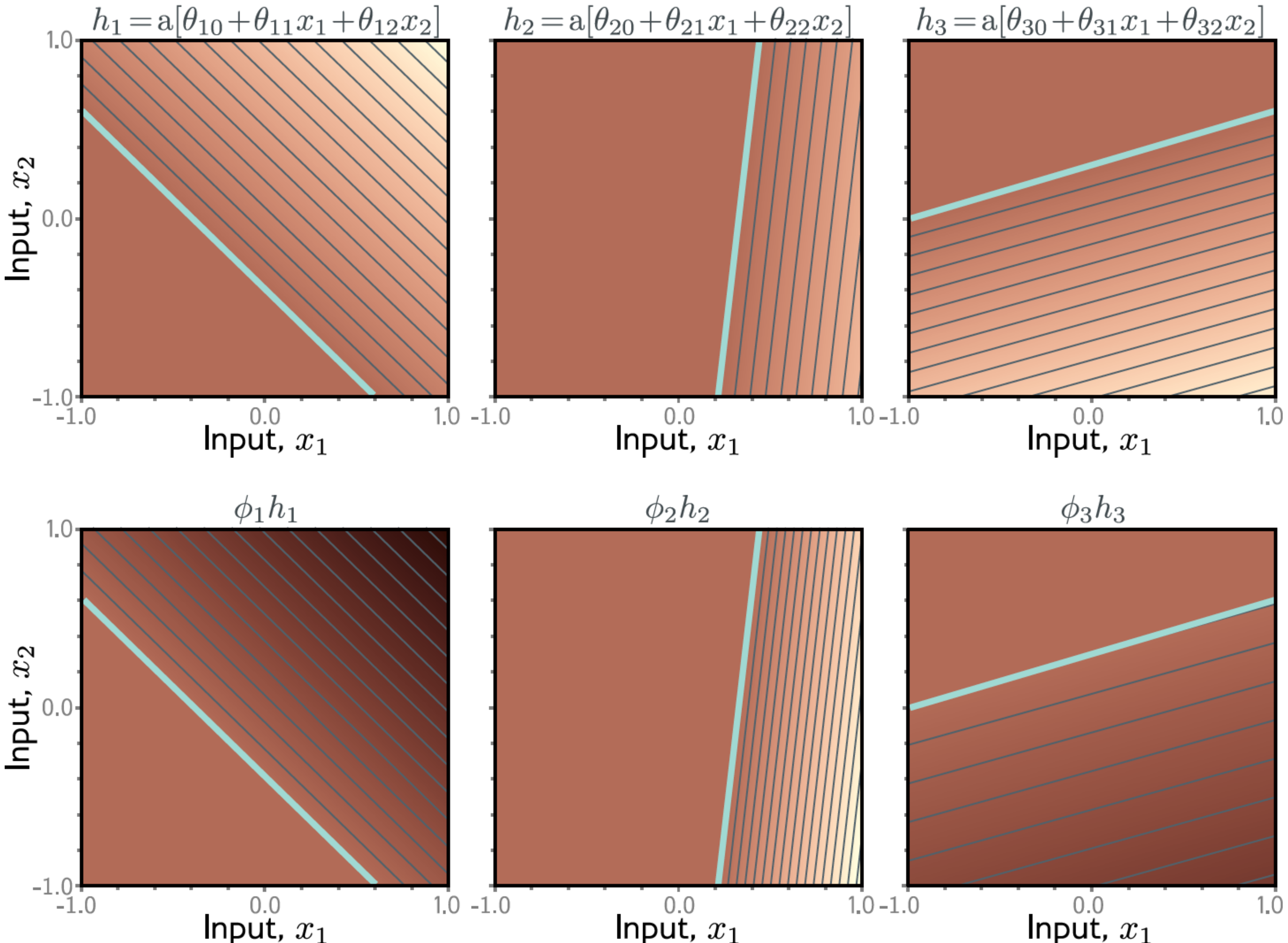
$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$



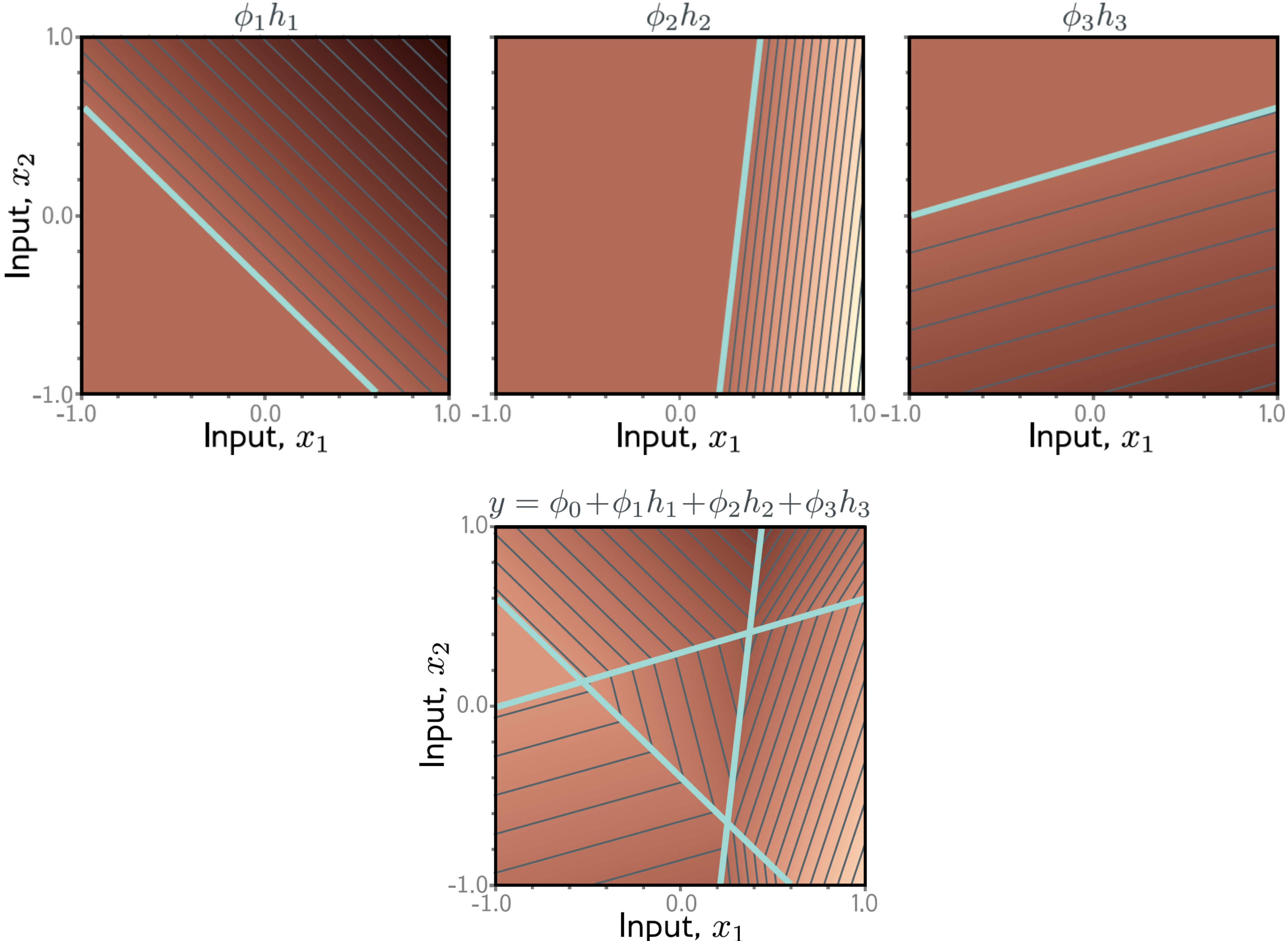

$\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2$
$\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2$
$\theta_{30} + \theta_{31}x_1 + \theta_{32}x_2$
Input, x_2
1.0
0.0
-1.0
-1.0
0.0
1.0
Input, x_1
Input, x_1
Input, x_1



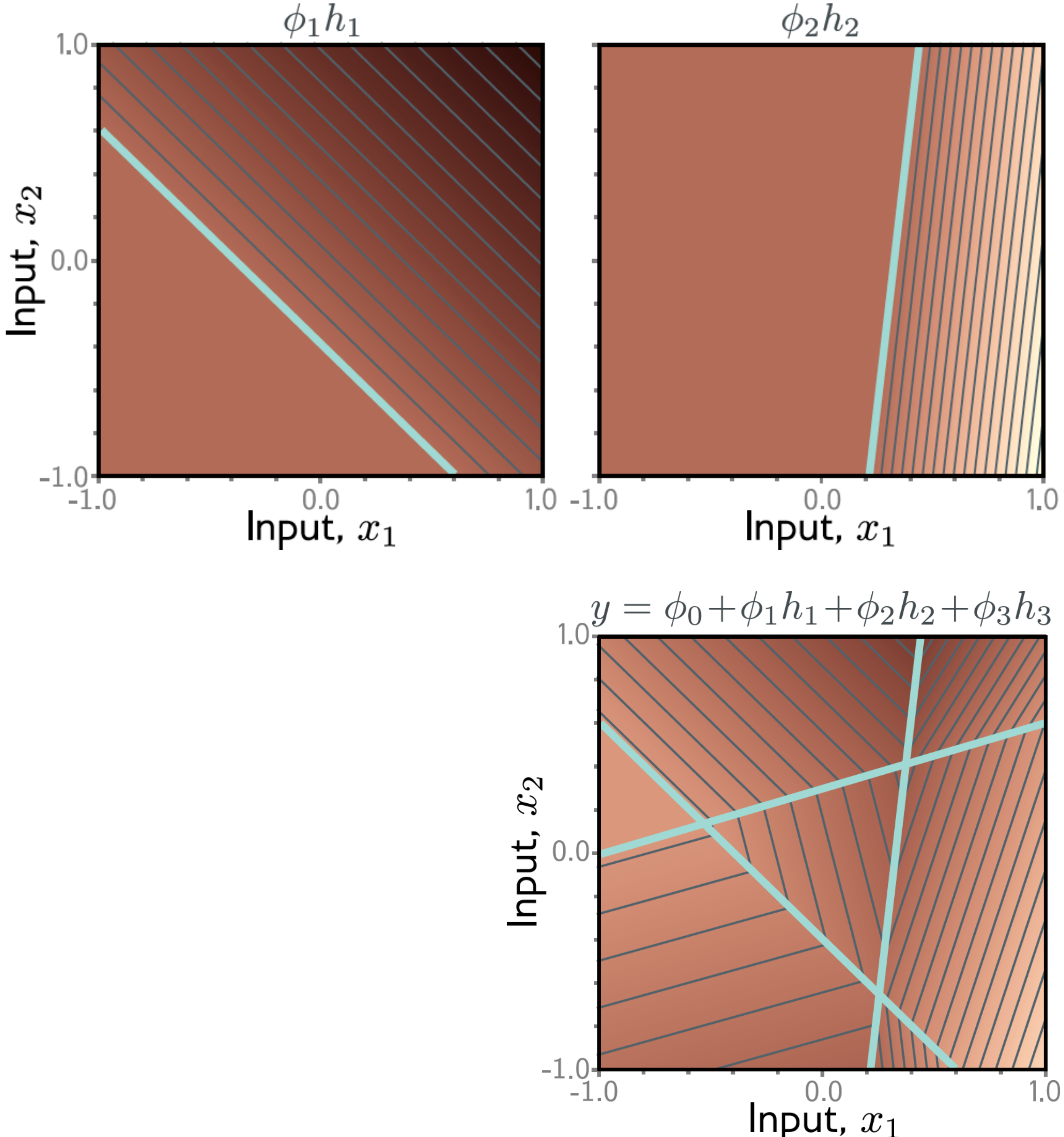
$\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2$
$\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2$
$\theta_{30} + \theta_{31}x_1 + \theta_{32}x_2$
$h_1 = \mathrm{a}[\theta_{10}+\theta_{11}x_1+\theta_{12}x_2]$
$h_2 = \mathrm{a}[\theta_{20}+\theta_{21}x_1+\theta_{22}x_2]$
$h_3 = \mathrm{a}[\theta_{30}+\theta_{31}x_1+\theta_{32}x_2]$
Input, x_2
Input, x_1
1.0
0.0
-1.0



$h_1 = \mathrm{a}[\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2]$
$h_2 = \mathrm{a}[\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2]$
$h_3 = \mathrm{a}[\theta_{30} + \theta_{31}x_1 + \theta_{32}x_2]$
$\phi_1 h_1$
$\phi_2 h_2$
$\phi_3 h_3$
Input, x_1
Input, x_2
1.0
0.0
-1.0



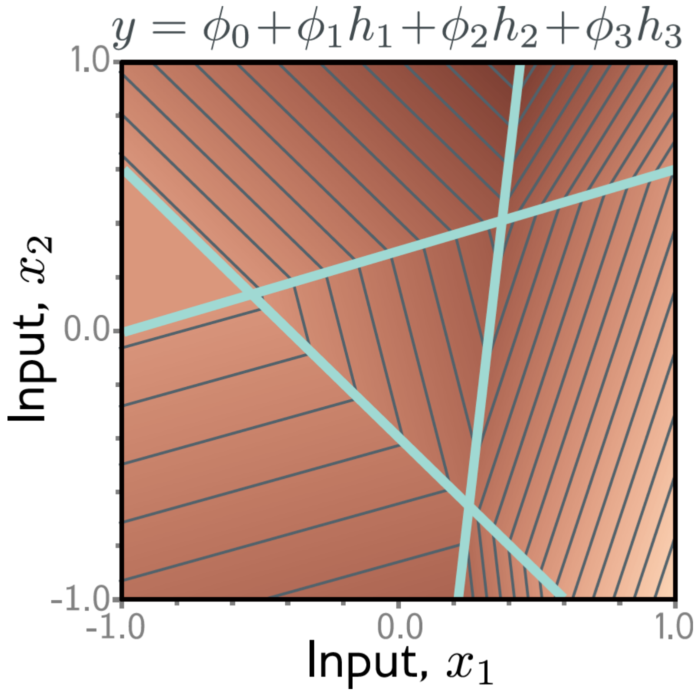
$\phi_1 h_1$
$\phi_2 h_2$
$\phi_3 h_3$
Input, x_2
Input, x_1
1.0
0.0
-1.0
$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$



Convex polygons

Question 1:

- For the 2D case, what if there were two outputs?
- If this is one of the outputs, what would the other one look like?



Shallow neural networks

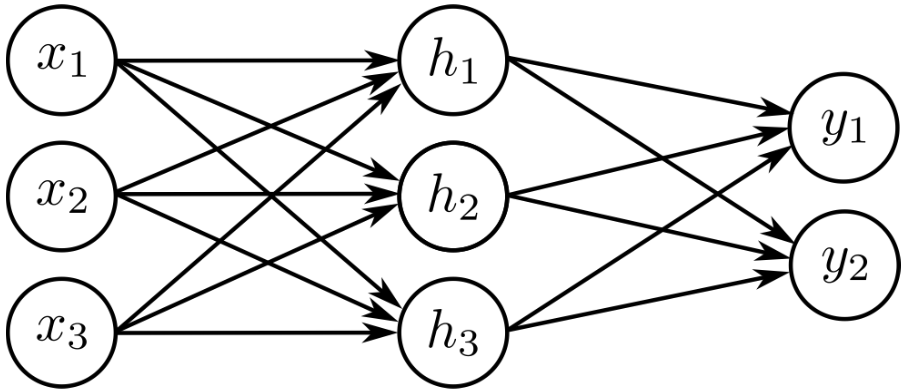
- Example network, 1 input, 1 ouput
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

Arbitrary inputs, hidden units, outputs

- Outputs, *D* hidden units, and inputs

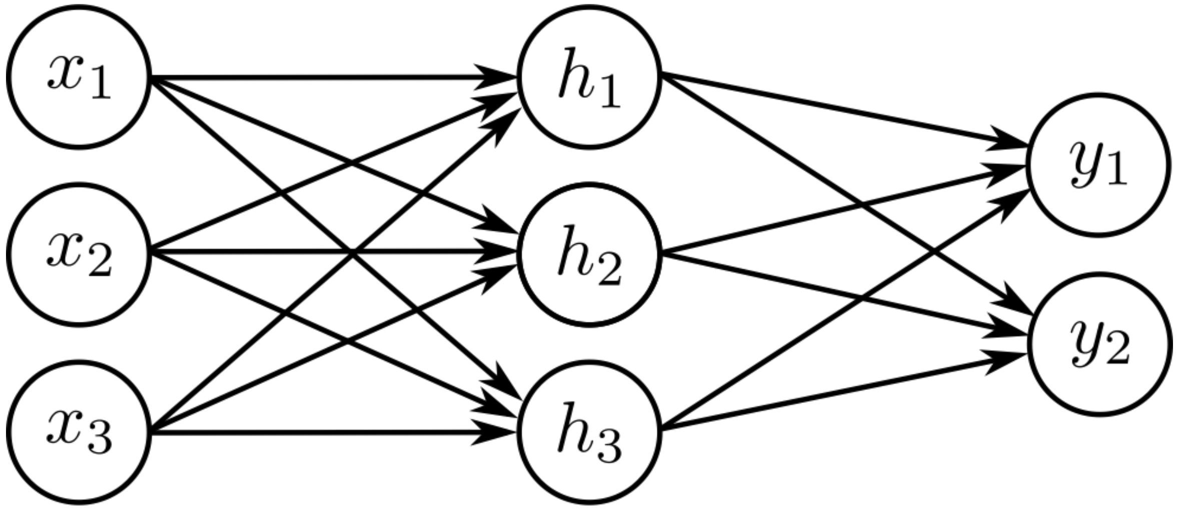
$$h_d = \mathrm{a}\left[\theta_{d0} + \sum_{i=1}^{D_i} \theta_{di} x_i\right] \qquad y_j = \phi_{j0} + \sum_{d=1}^{D} \phi_{jd} h_d$$

- e.g., Three inputs, three hidden units, two outputs



Question 2:

- How many parameters does this model have?

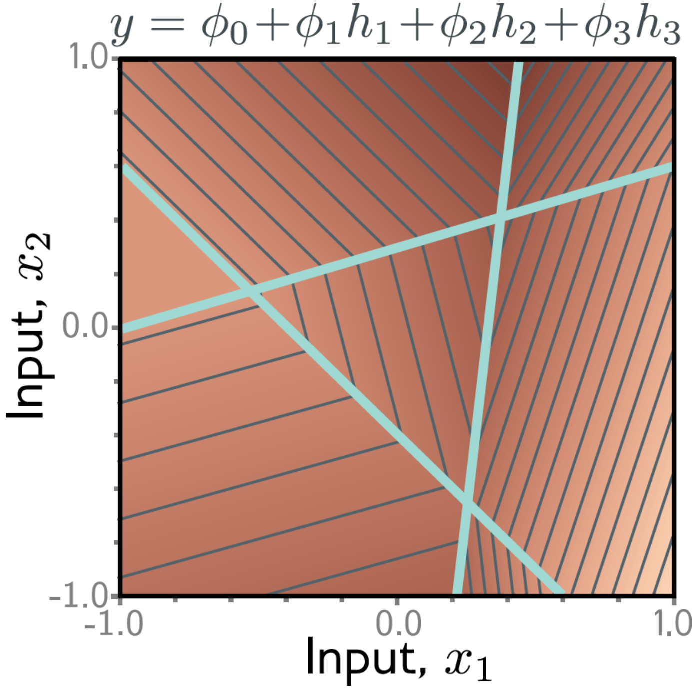


Shallow neural networks

- Example network, 1 input, 1 ouput
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

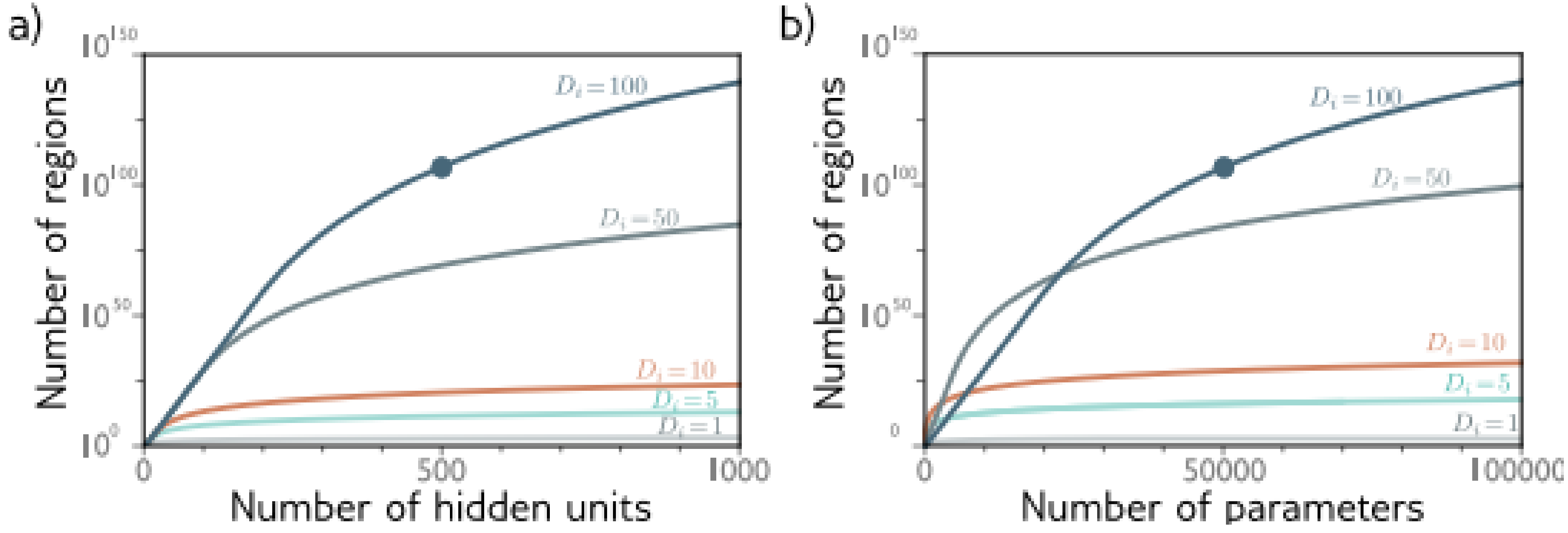
Number of output regions

- In general, each output consists of D dimensional convex polytopes
- With two inputs, and three outputs, we saw there were seven polygons:



Number of output regions

- In general, each output consists of D dimensional convex polytopes
- How many?



Highlighted point = 500 hidden units or 51,001 parameters

Number of regions:

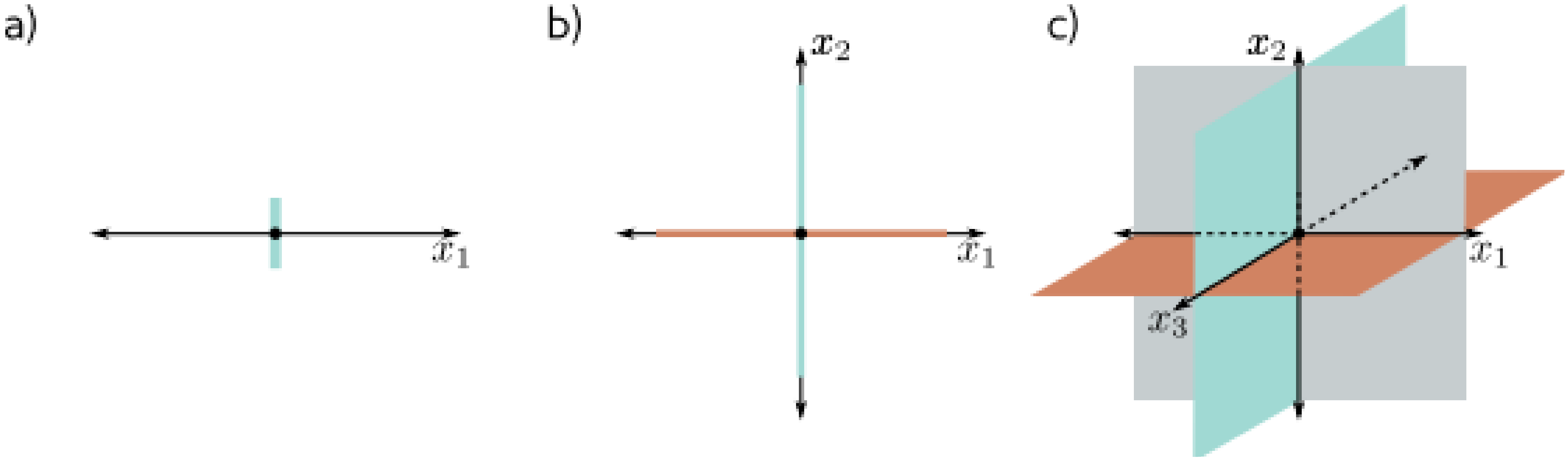
- Number of regions created by *D* > planes in dimensions was proved by Zavlasky (1975) to be:

$$\sum_{j=0}^{D_i} \binom{D}{j}$$

Binomial coefficients!

- How big is this? It's greater than but less than .

Proof that bigger than larger than



1D input with 1 hidden unit creates two regions (one joint)

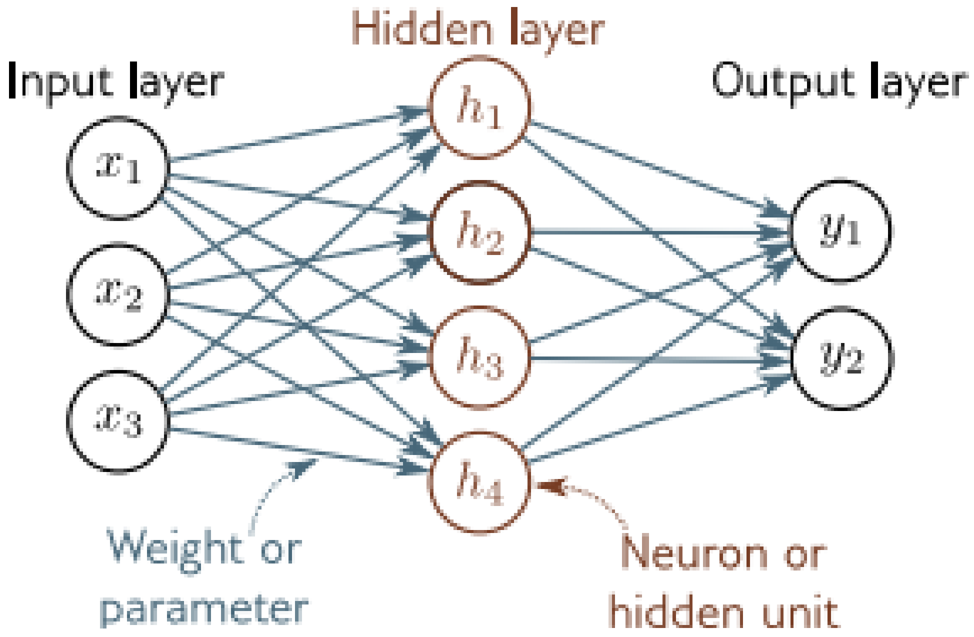
2D input with 2 hidden units creates four regions (two lines)

3D input with D hidden units creates eight regions (three planes)

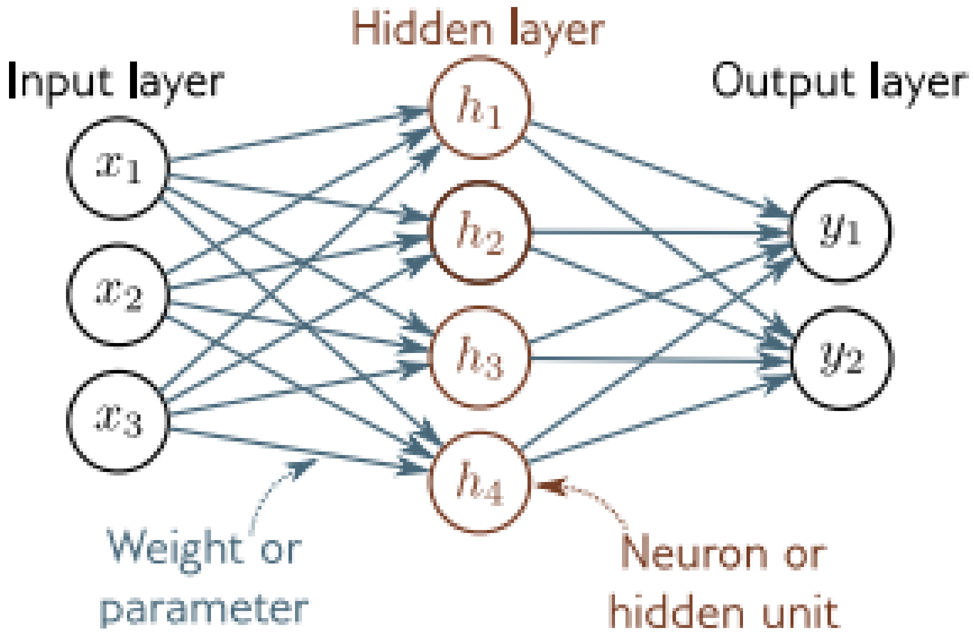
Shallow neural networks

- Example network, 1 input, 1 ouput
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

Nomenclature

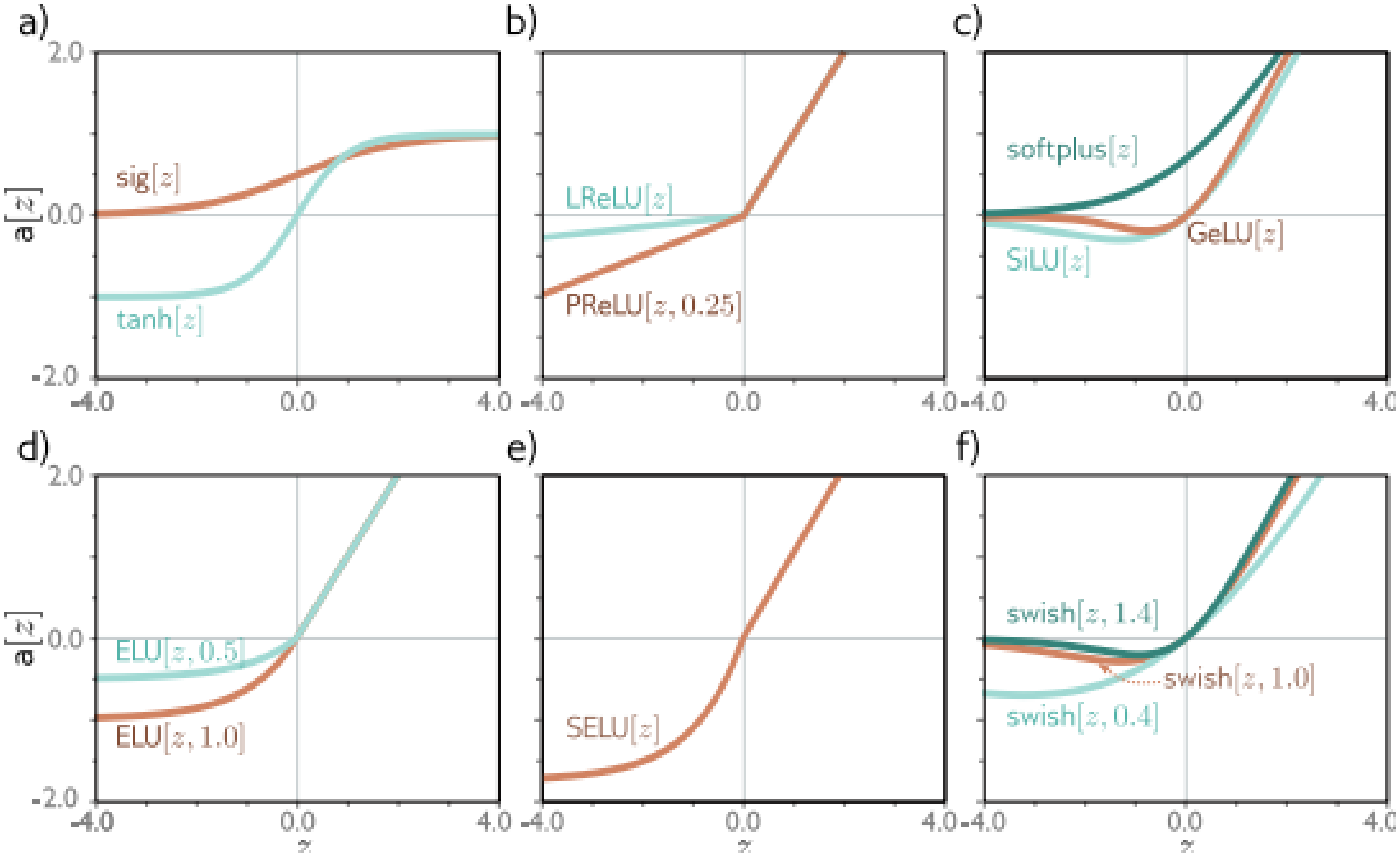


Nomenclature

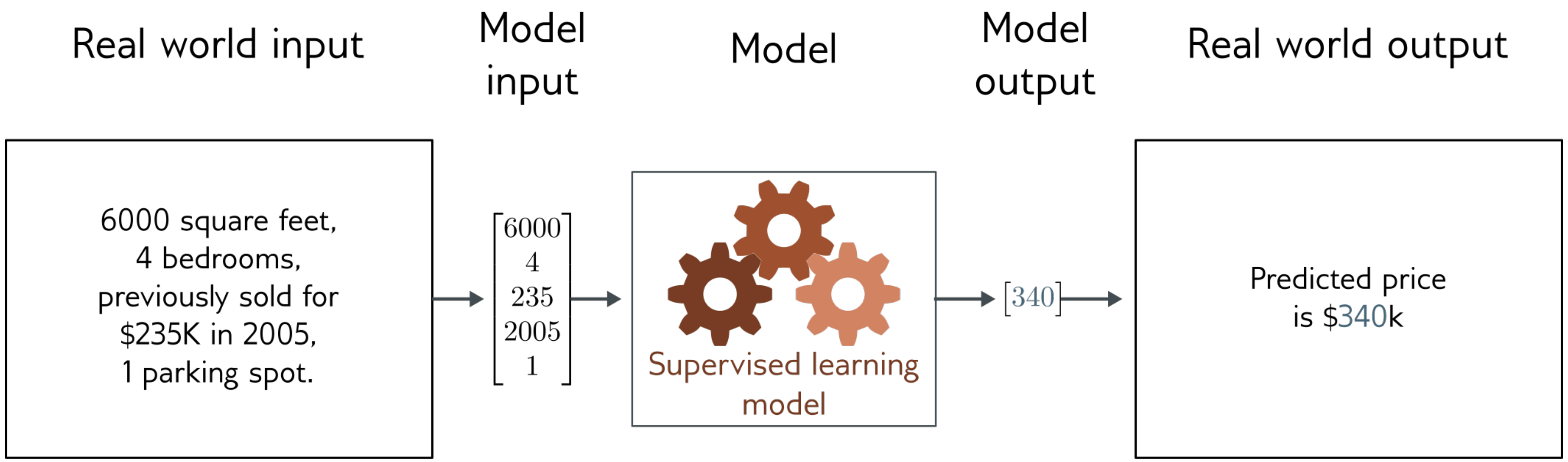


- Y-offsets = biases
- Slopes = weights
- Everything in one layer connected to everything in the next = Fully Connected Network
- No loops = Feedforward network
- Values after ReLU (activation functions) = activations
- Values before ReLU = pre-activations
- One hidden layer = shallow neural network
- More than one hidden layer = deep neural network
- Number of hidden units

Other activation functions



Regression



We have built a model that can:

- take an arbitrary number of inputs
- output an arbitrary number of outputs
- model a function of arbitrary complexity between the two

$$h_d = \mathrm{a}\left[\theta_{d0} + \sum_{i=1}^{D_i} \theta_{di} x_i\right] \qquad y_j = \phi_{j0} + \sum_{d=1}^{D} \phi_{jd} h_d$$

Next time:

- What happens if we feed one neural network into another neural network?