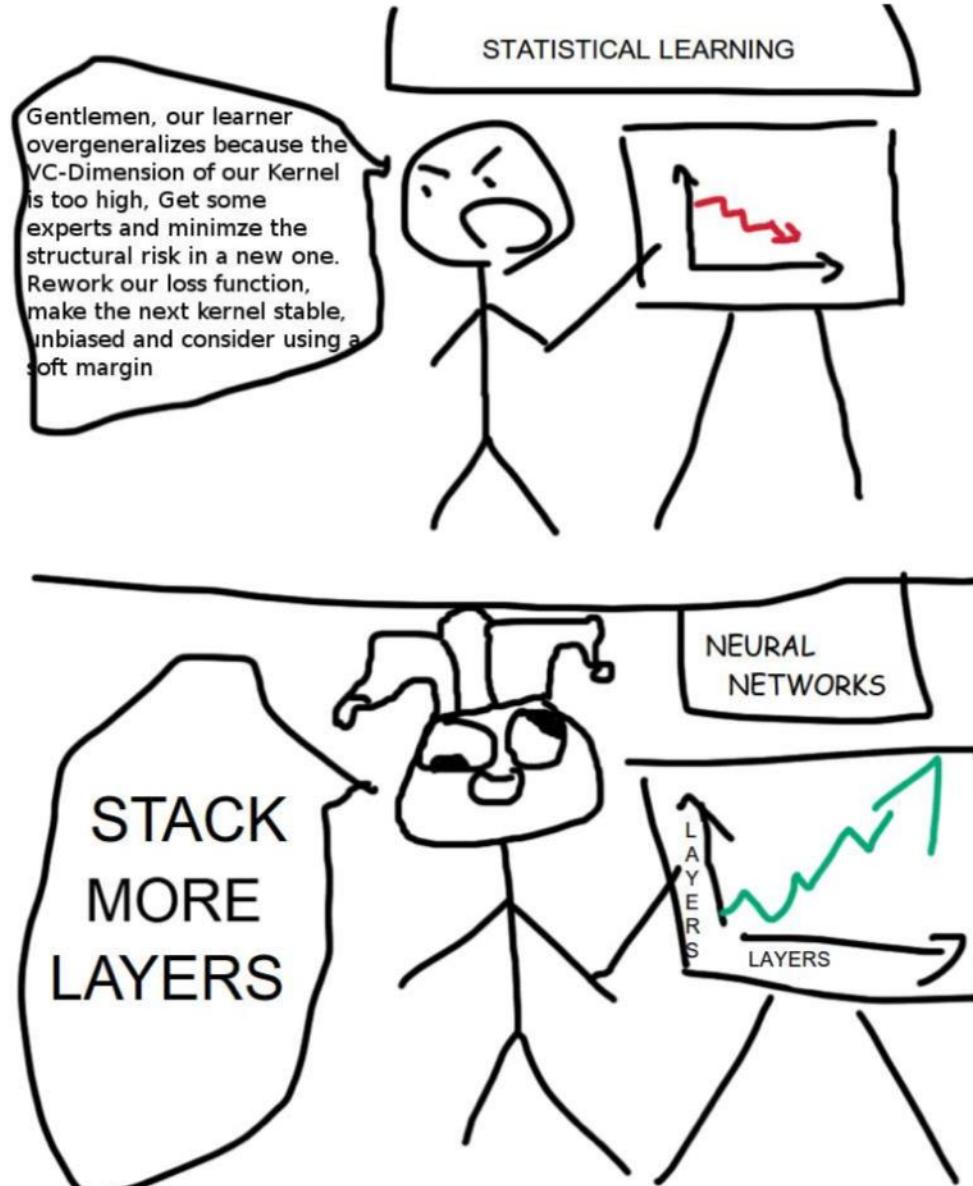


Multi-layer Perceptrons



Miles Cranmer
astroautomata.com

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

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

Example shallow network

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

Example shallow network

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

Example shallow network

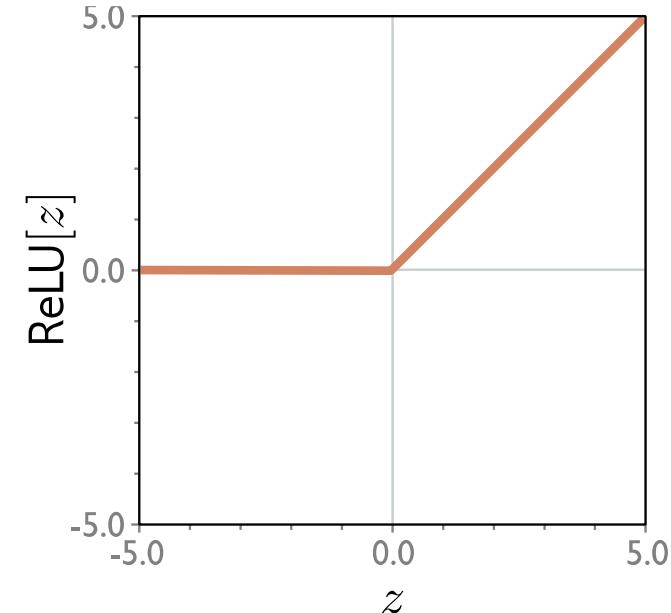
$$y = f[x, \phi]$$

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

Activation function

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

Rectified Linear Unit
(particular kind of activation function)



Rectified Linear Unit
Activation Function
(ReLU)



$\max(0, x)$



Example shallow network

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

This model has 10 parameters:

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

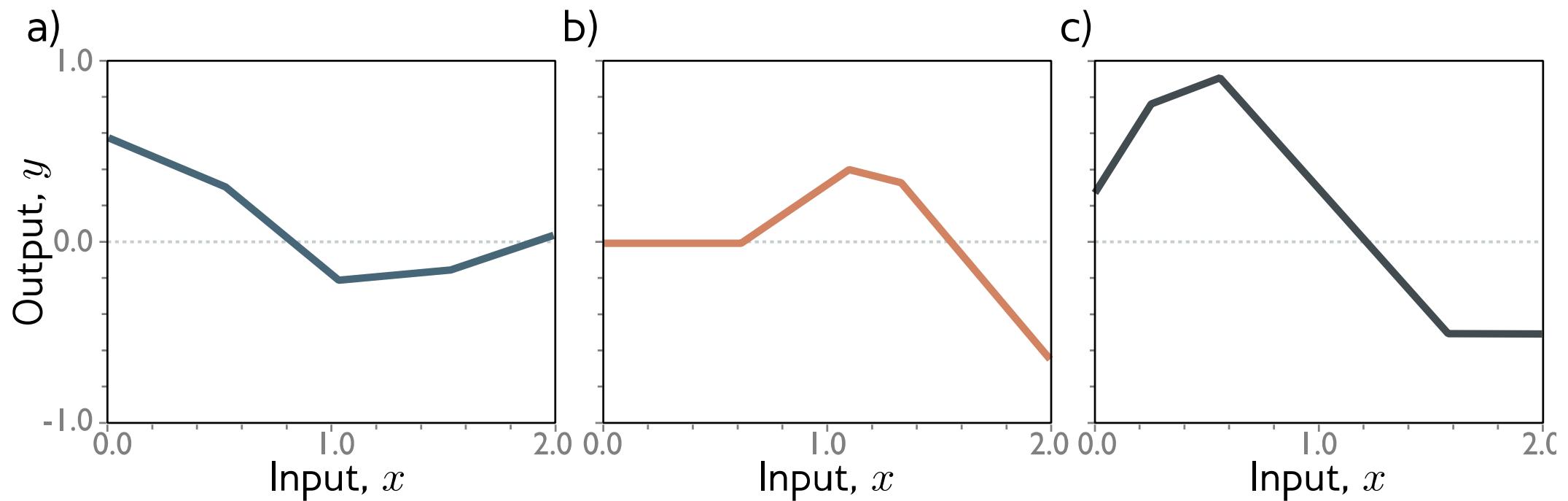
- Represents a family of functions
- Parameters determine particular function
- Given parameters can perform inference (run equation)
- Given training dataset $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^I$
- Define loss function $L[\phi]$ (least squares)
- Change parameters to minimize loss function

Example shallow network

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

Example shallow network

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



Piecewise linear functions with three joints

Hidden units

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

Break down into two parts:

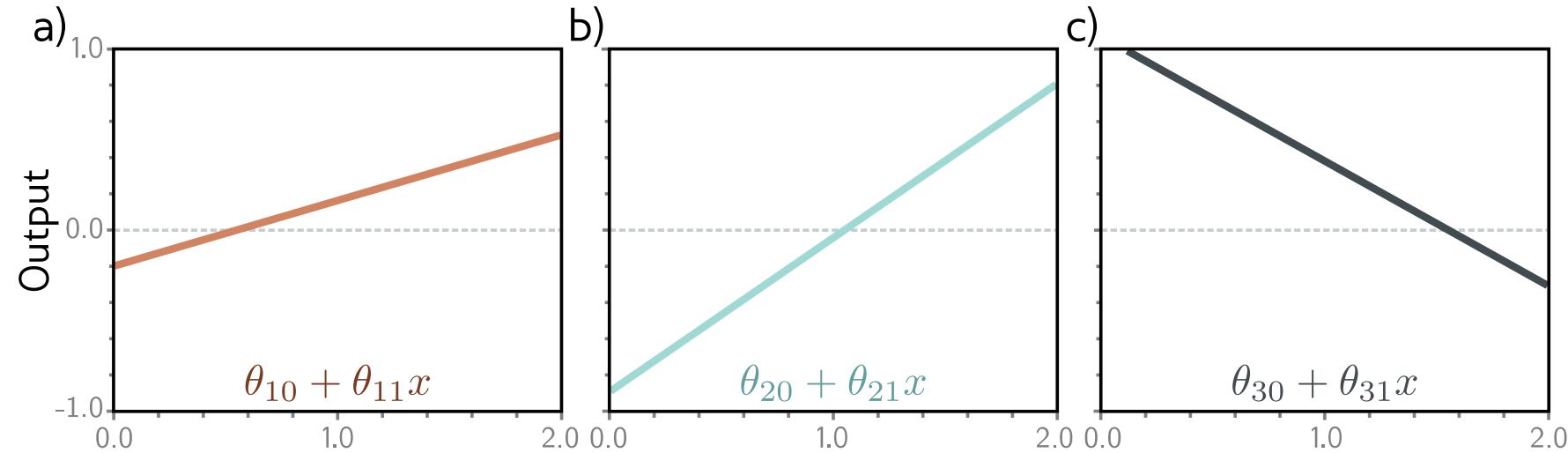
$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$

where:

Hidden units

$$\left\{ \begin{array}{l} h_1 = a[\theta_{10} + \theta_{11}x] \\ h_2 = a[\theta_{20} + \theta_{21}x] \\ h_3 = a[\theta_{30} + \theta_{31}x] \end{array} \right.$$

1. compute three linear functions

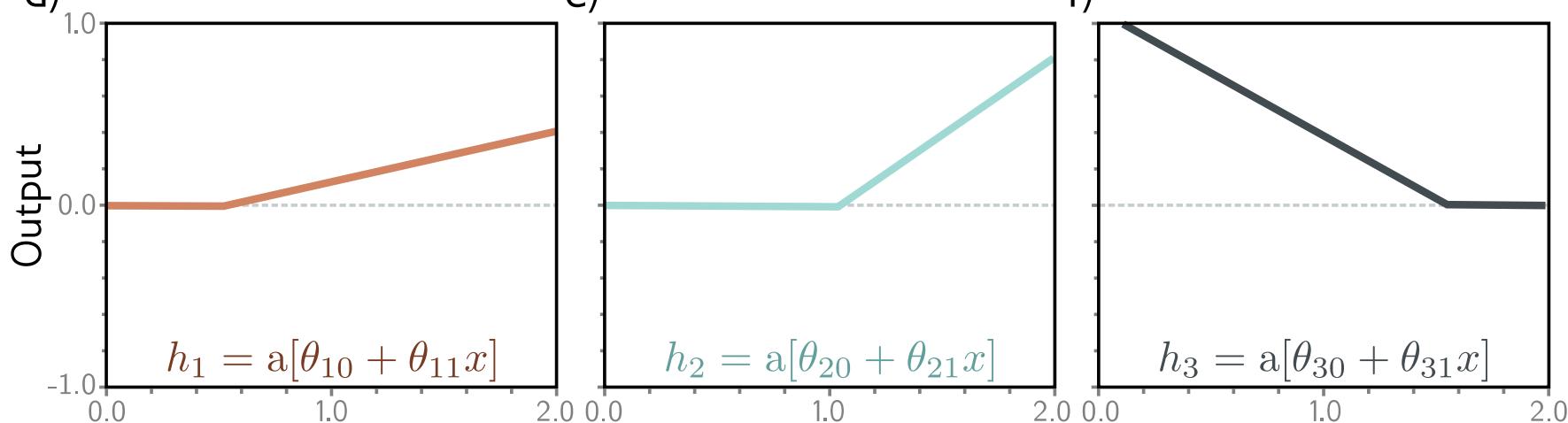
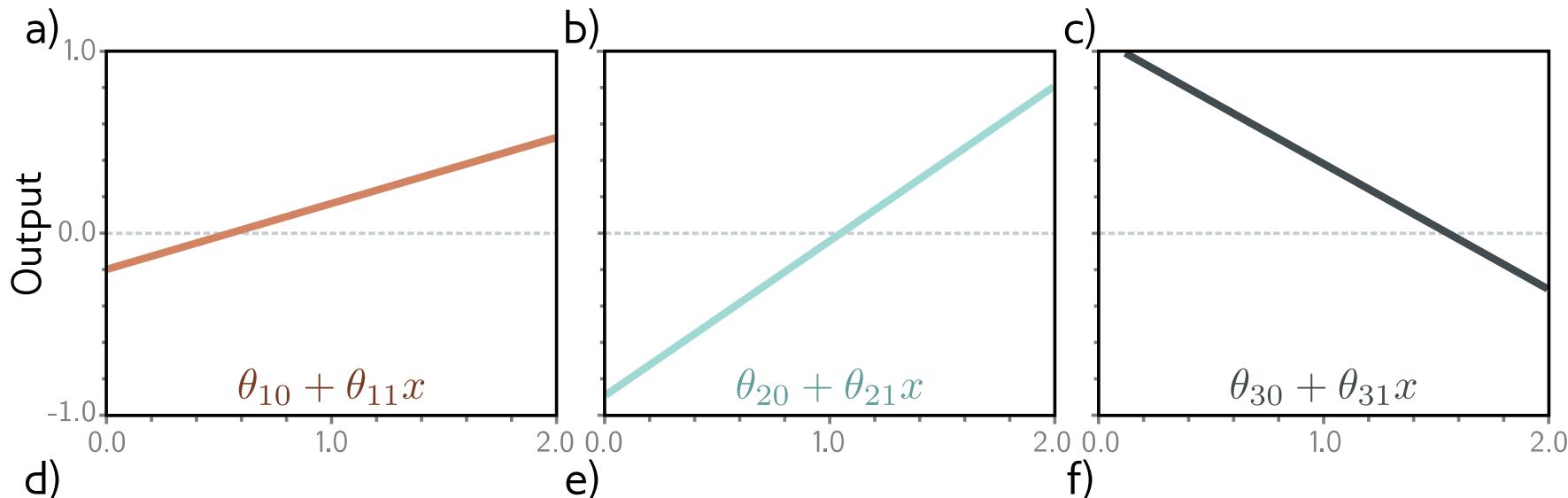


2. Weight the hidden units

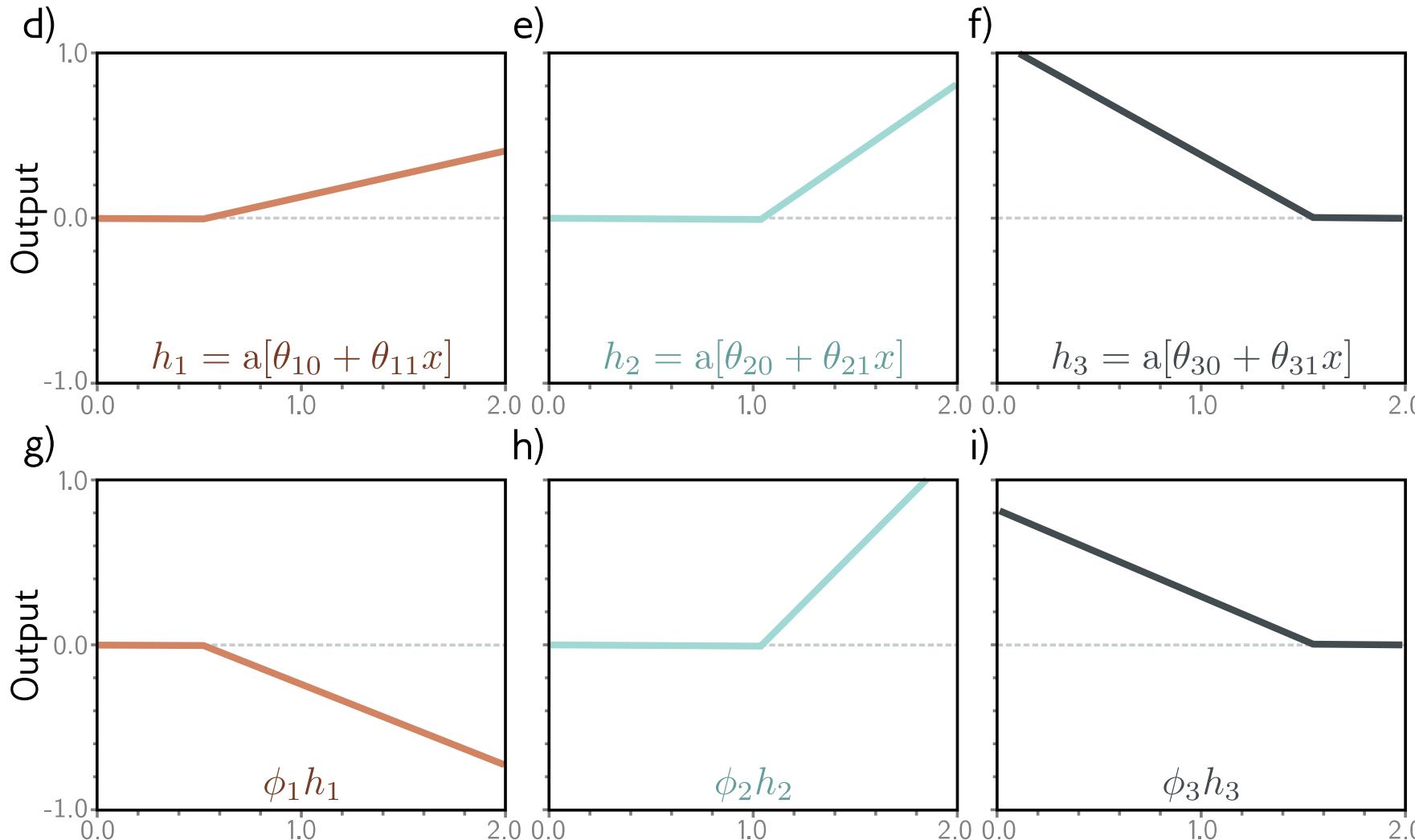
$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x],$$



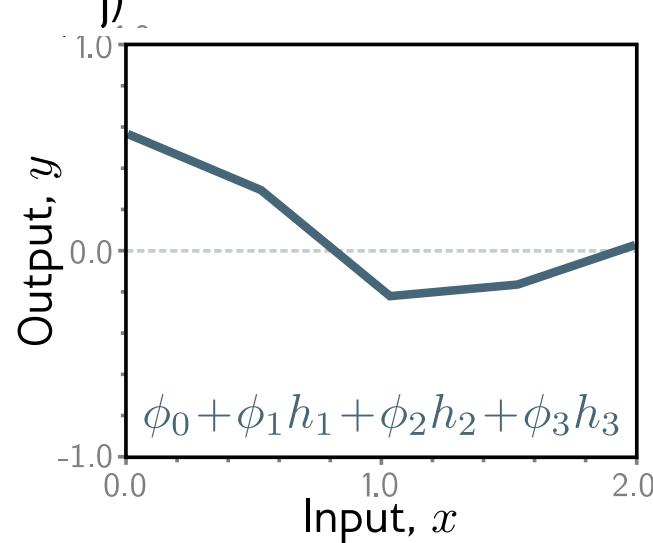
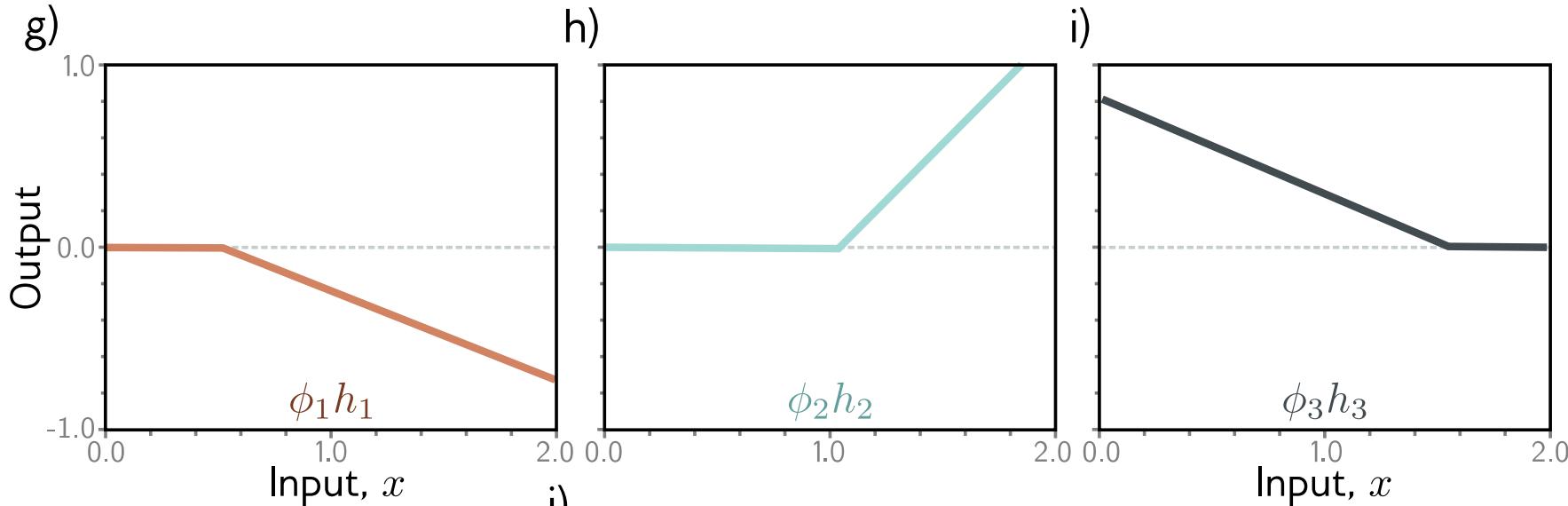
2. Pass through ReLU functions to compute hidden units



2. Pass through ReLU

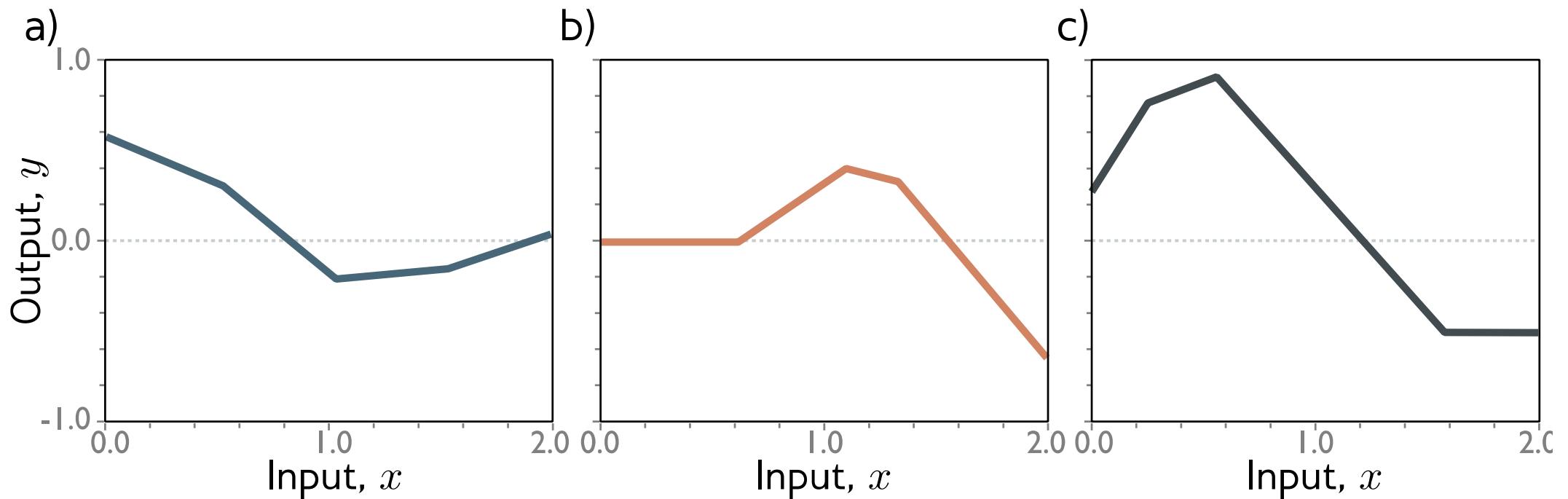
functions to
compute hidden units

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



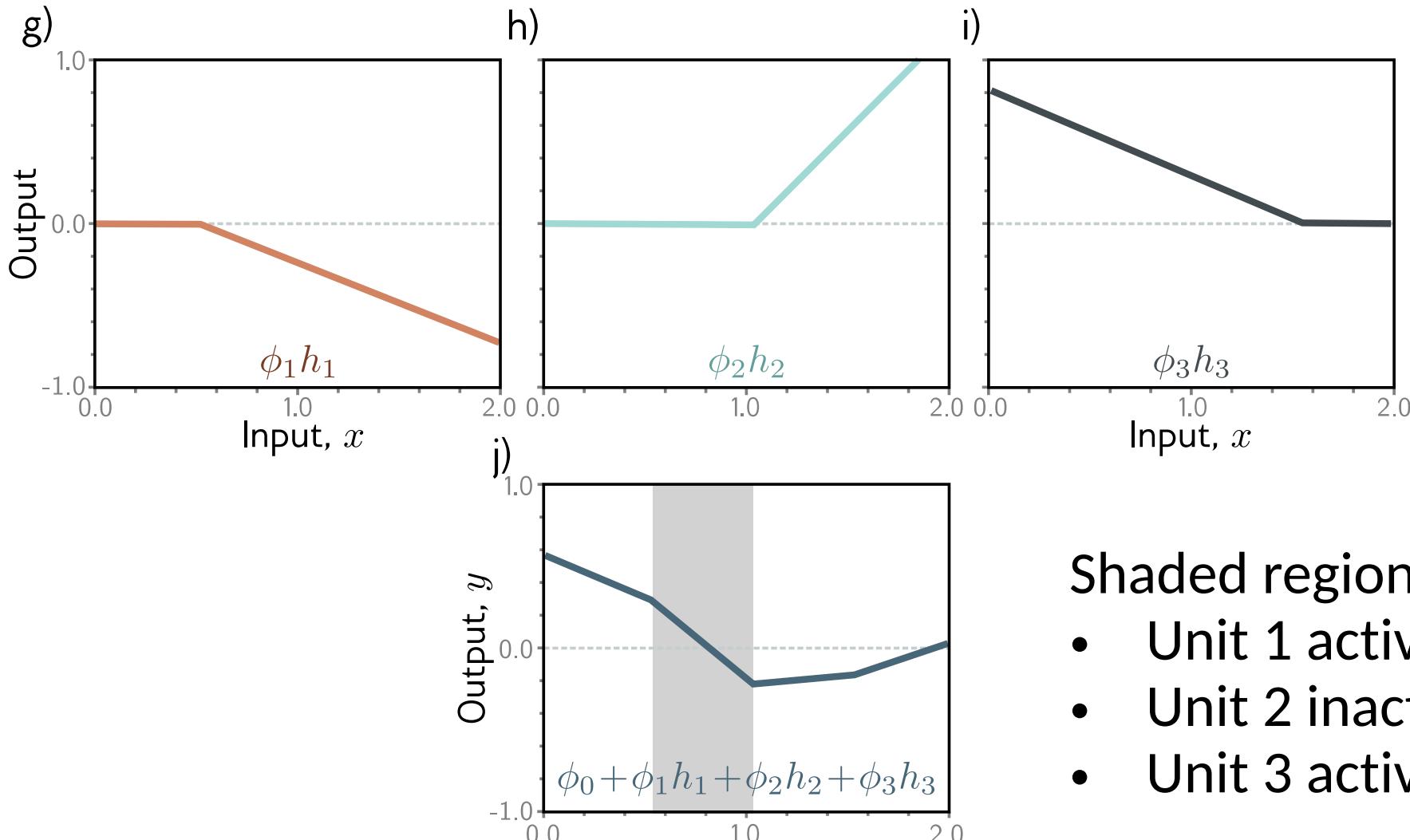
Example shallow network

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



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

Activation pattern = which hidden units are activated



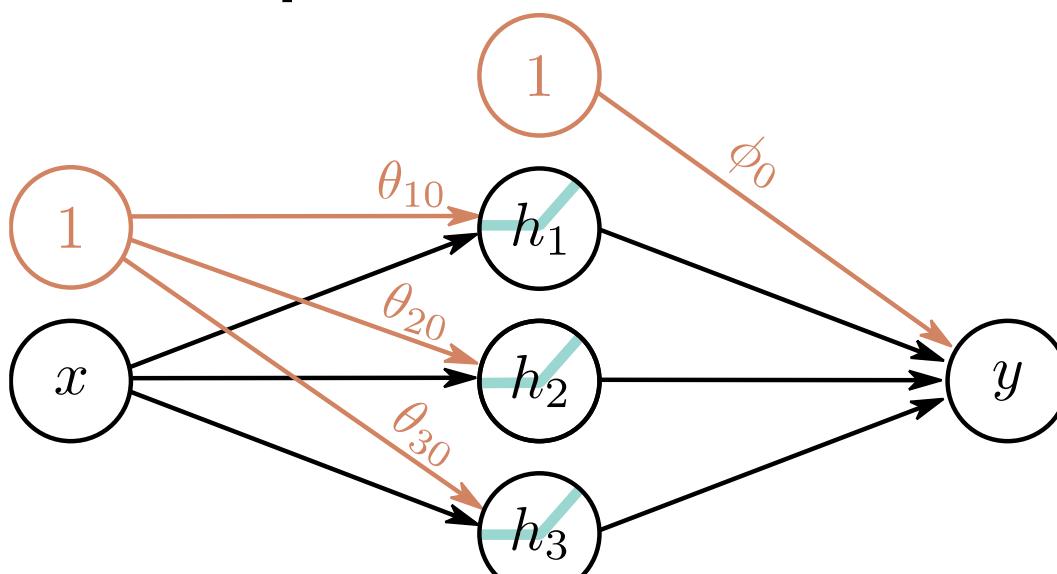
Depicting neural networks

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

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



Each parameter multiplies its source and adds to its target

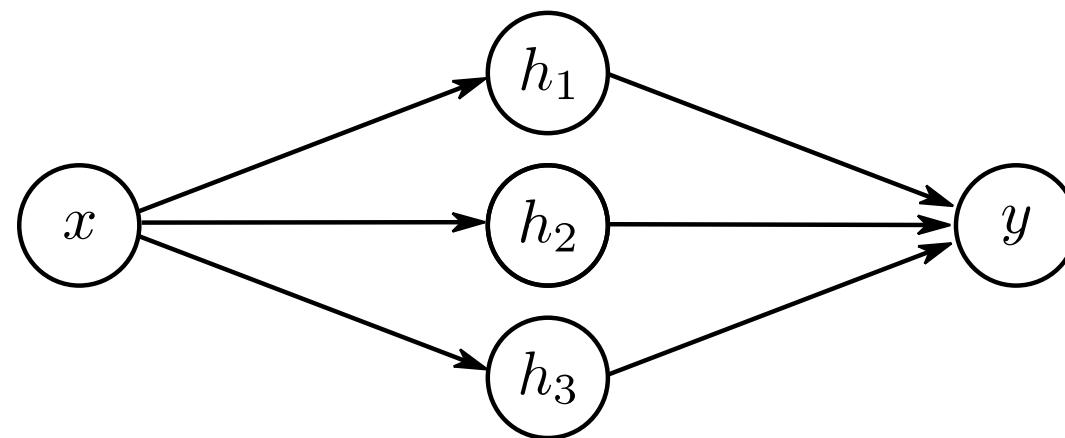
Depicting neural networks

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

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



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:

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

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

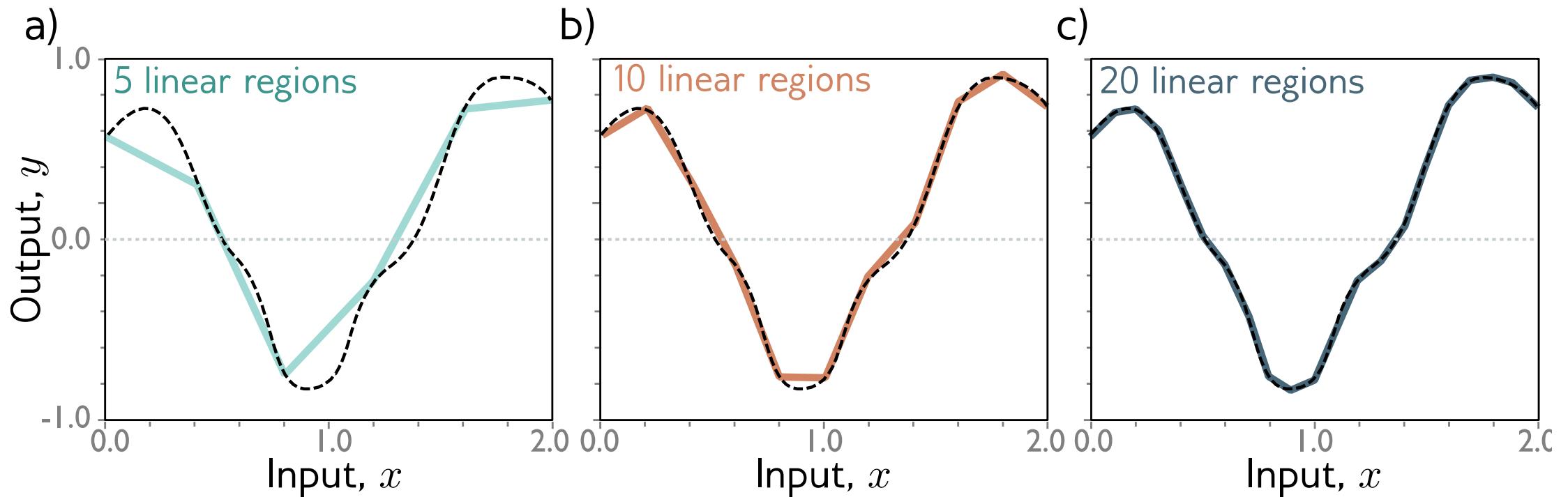
With D hidden units:

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

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

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 output
- 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

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

$$h_4 = a[\theta_{40} + \theta_{41}x]$$

$$y_1 = \phi_{10} + \phi_{11}h_1 + \phi_{12}h_2 + \phi_{13}h_3 + \phi_{14}h_4$$

$$y_2 = \phi_{20} + \phi_{21}h_1 + \phi_{22}h_2 + \phi_{23}h_3 + \phi_{24}h_4$$

Two outputs

- 1 input, 4 hidden units, 2 outputs

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

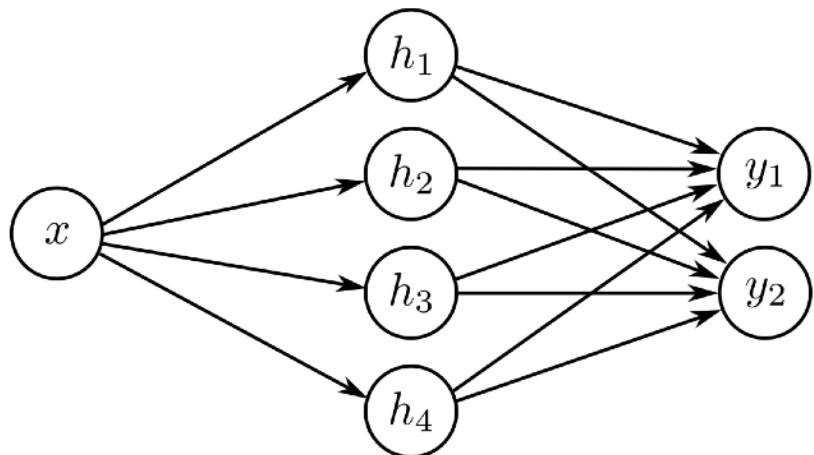
$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

$$h_4 = a[\theta_{40} + \theta_{41}x]$$

$$y_1 = \phi_{10} + \phi_{11}h_1 + \phi_{12}h_2 + \phi_{13}h_3 + \phi_{14}h_4$$

$$y_2 = \phi_{20} + \phi_{21}h_1 + \phi_{22}h_2 + \phi_{23}h_3 + \phi_{24}h_4$$



Two outputs

- 1 input, 4 hidden units, 2 outputs

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

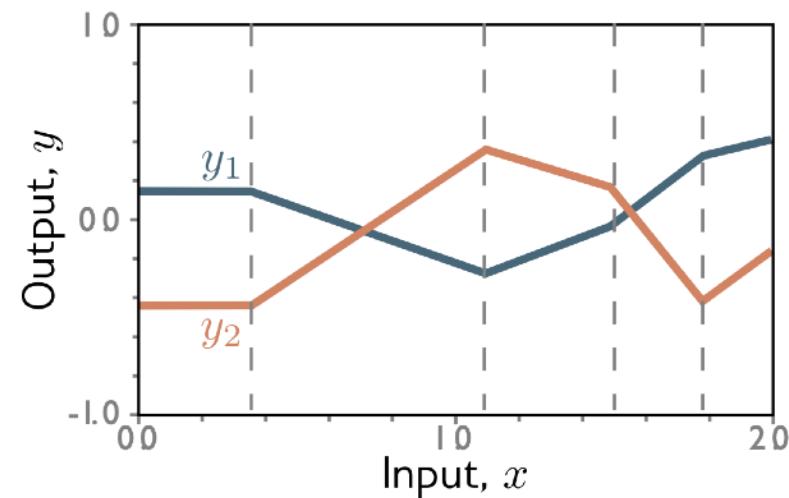
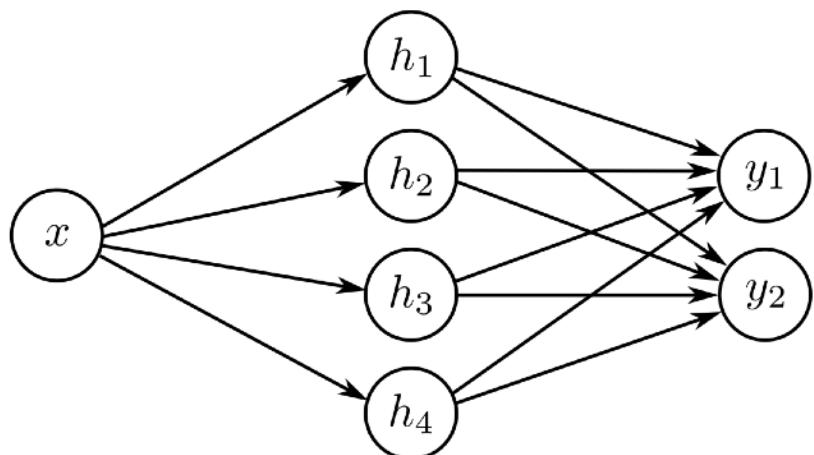
$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

$$h_4 = a[\theta_{40} + \theta_{41}x]$$

$$y_1 = \phi_{10} + \phi_{11}h_1 + \phi_{12}h_2 + \phi_{13}h_3 + \phi_{14}h_4$$

$$y_2 = \phi_{20} + \phi_{21}h_1 + \phi_{22}h_2 + \phi_{23}h_3 + \phi_{24}h_4$$



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

Two inputs

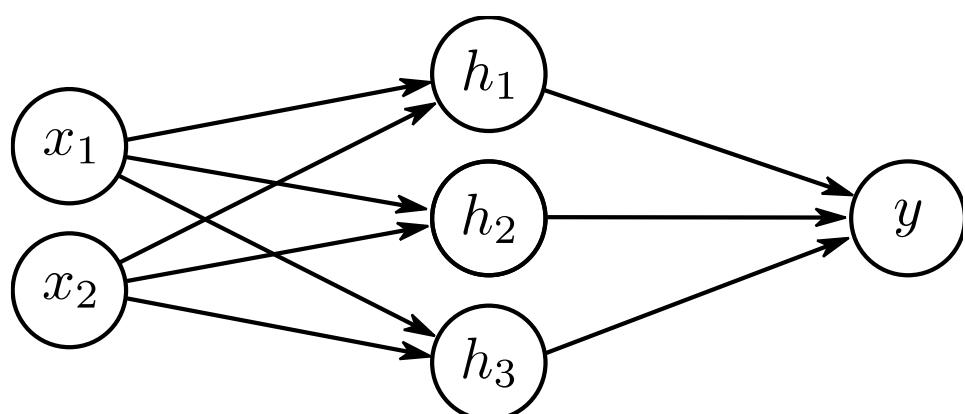
- 2 inputs, 3 hidden units, 1 output

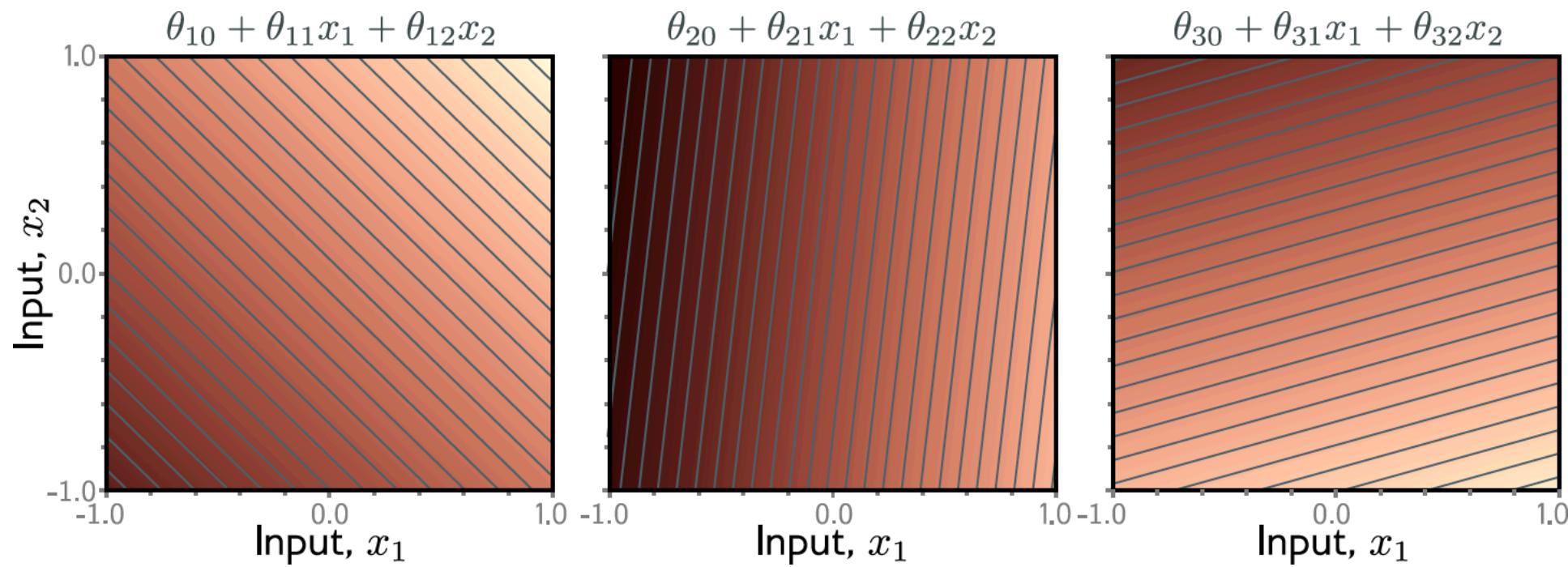
$$h_1 = a[\theta_{10} + \theta_{11}x_1 + \theta_{12}x_2]$$

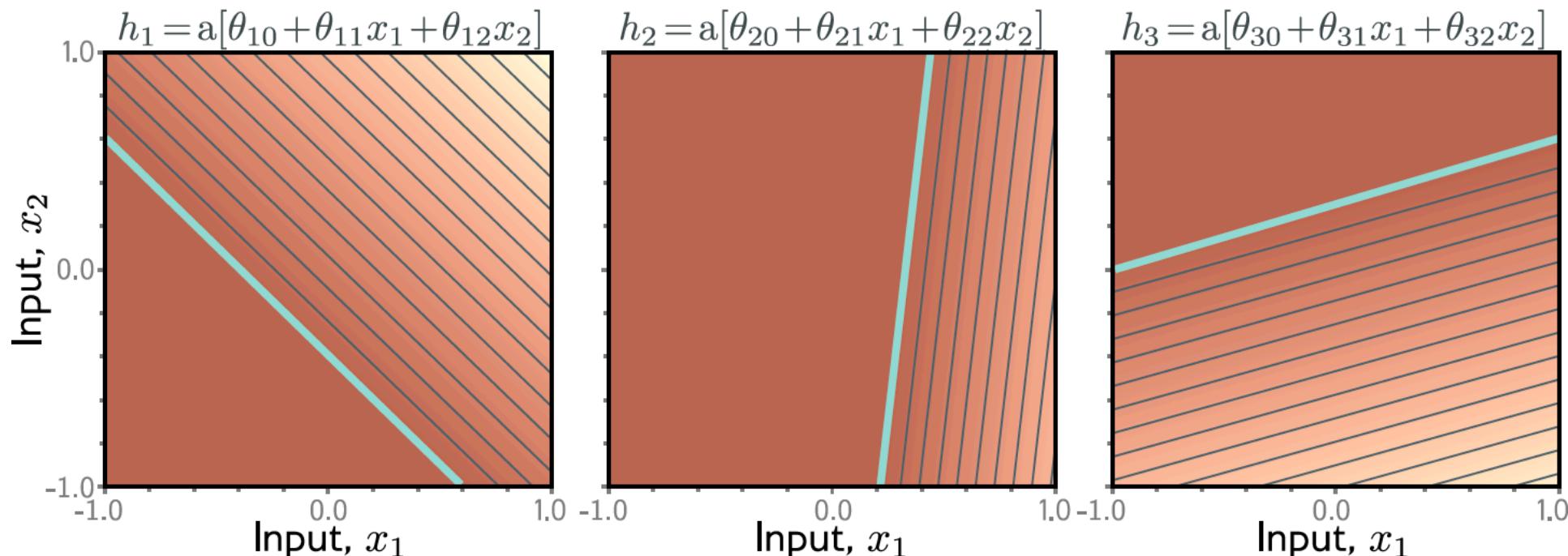
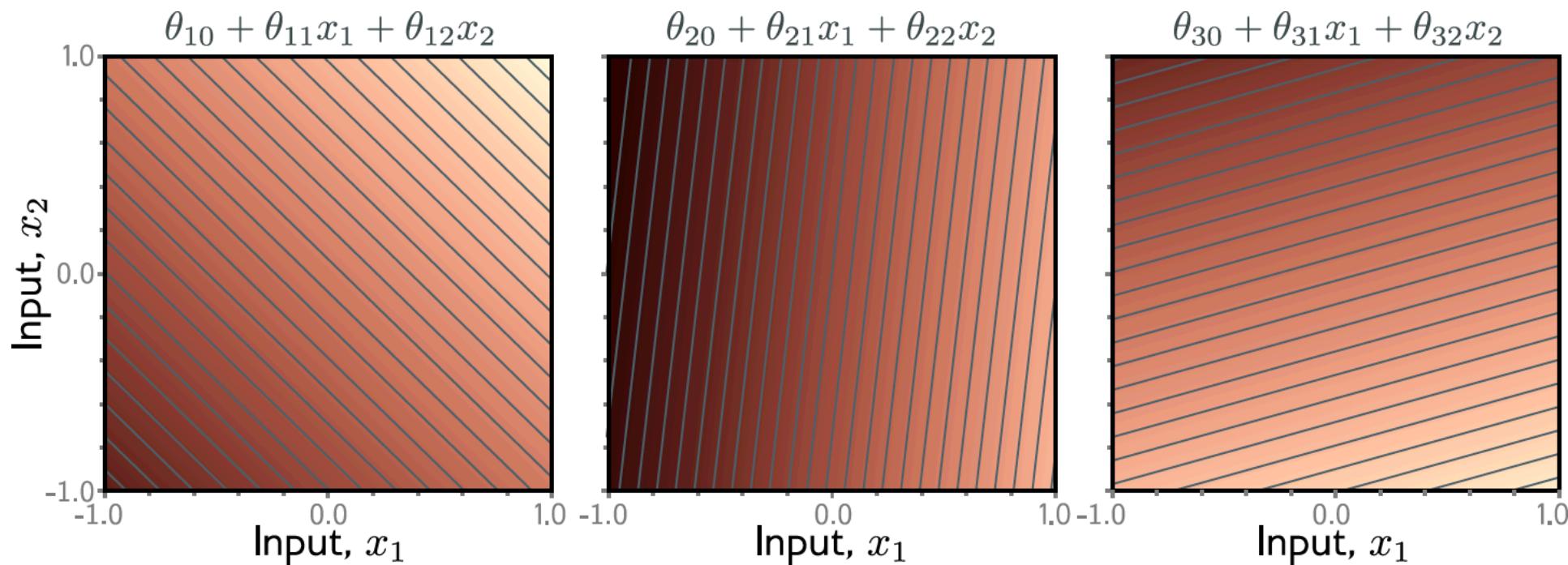
$$h_2 = a[\theta_{20} + \theta_{21}x_1 + \theta_{22}x_2]$$

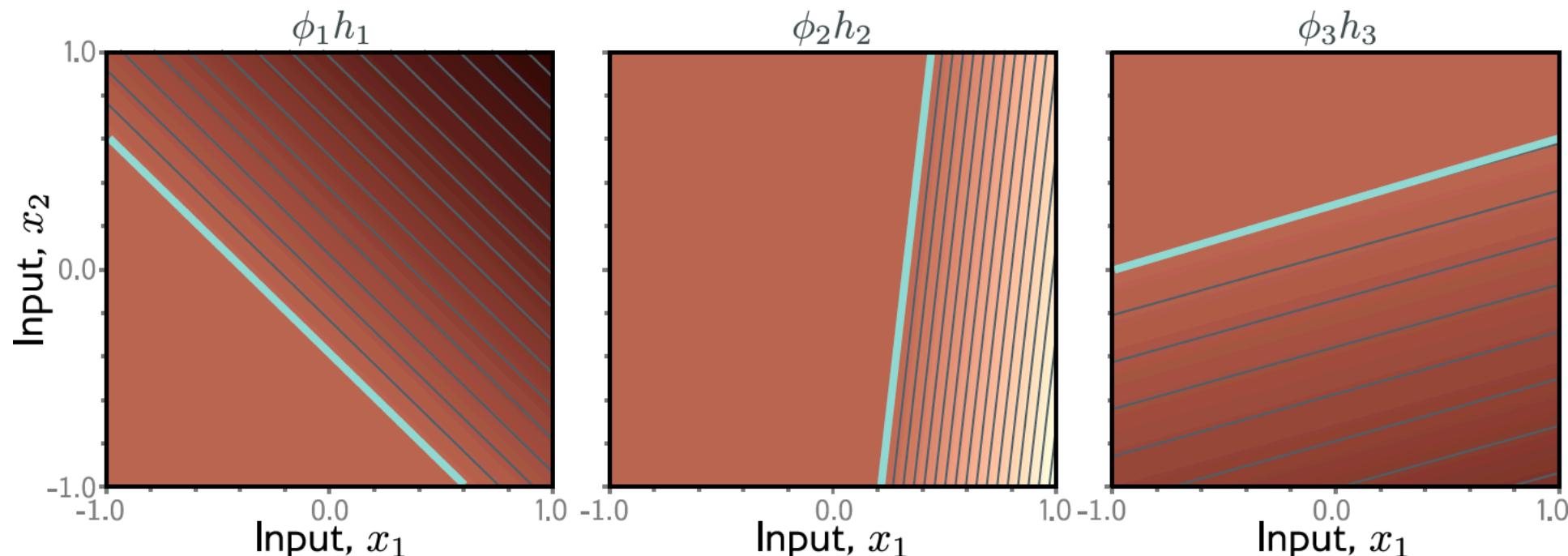
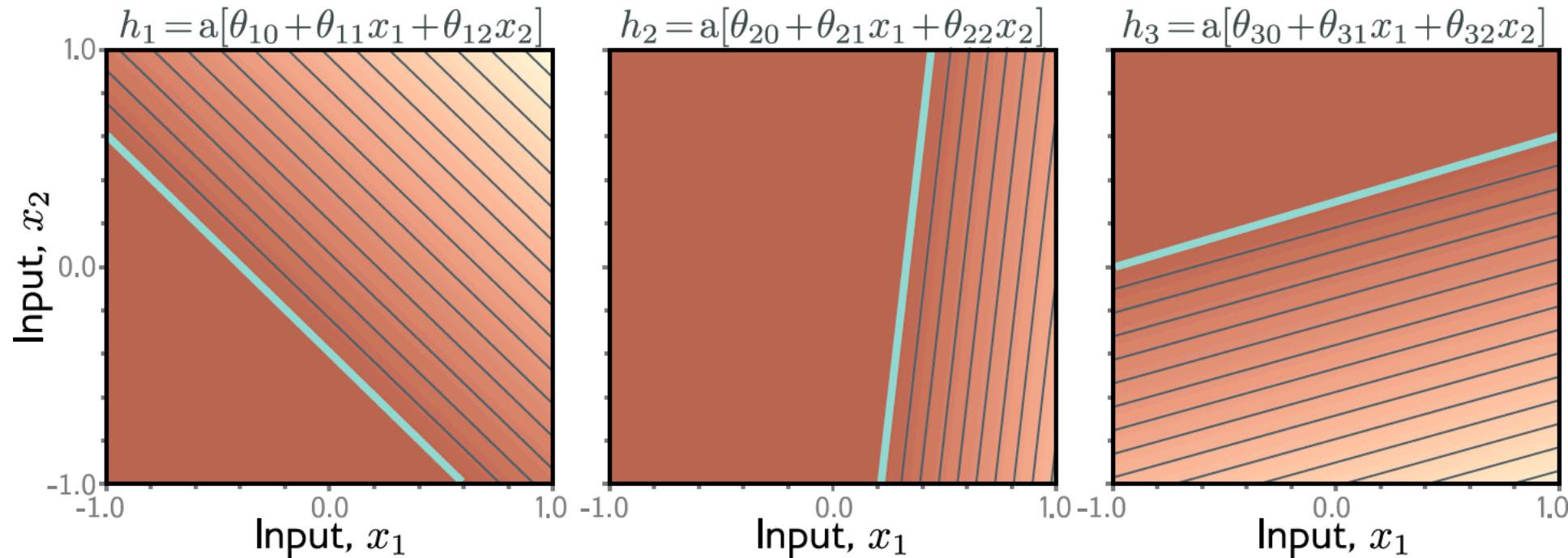
$$h_3 = a[\theta_{30} + \theta_{31}x_1 + \theta_{32}x_2]$$

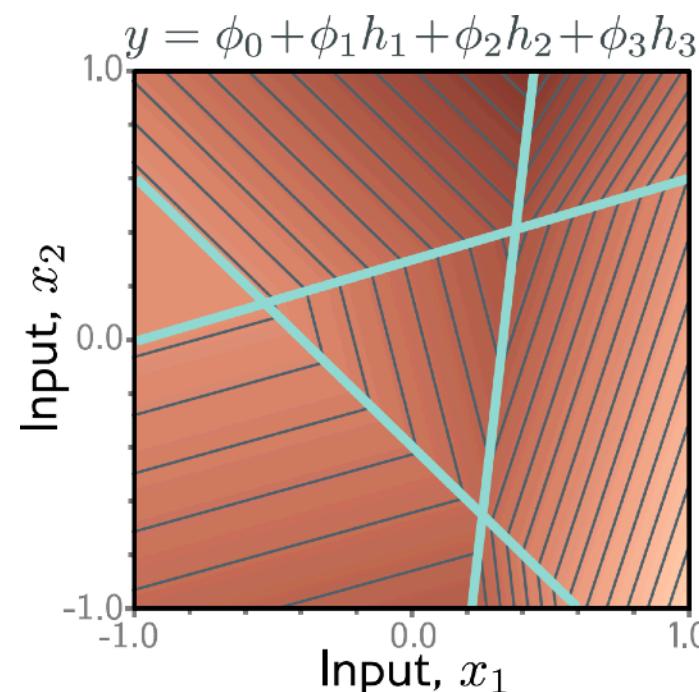
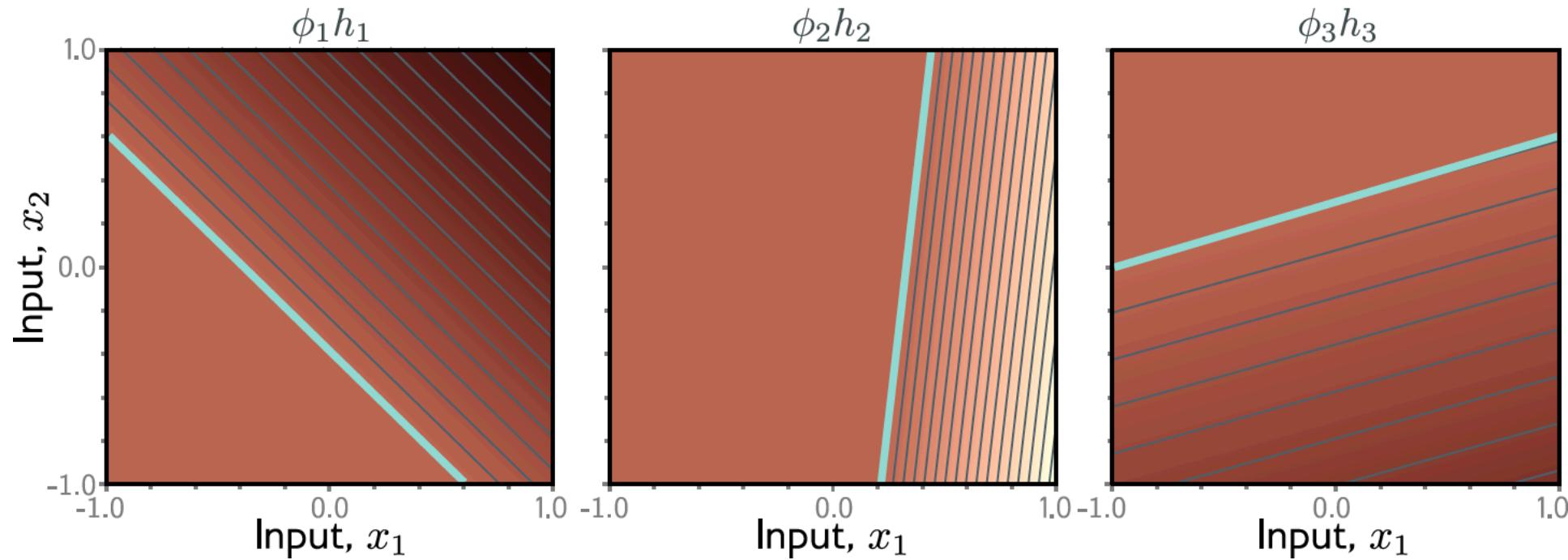
$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$

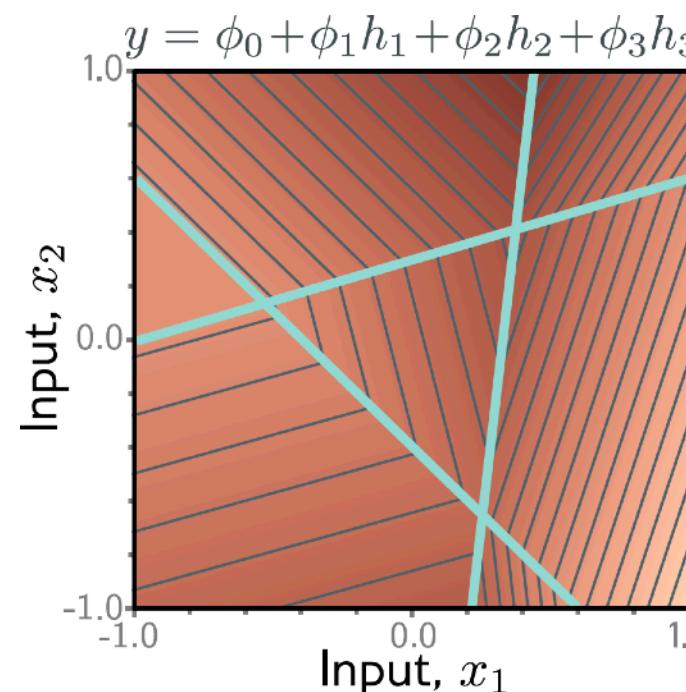
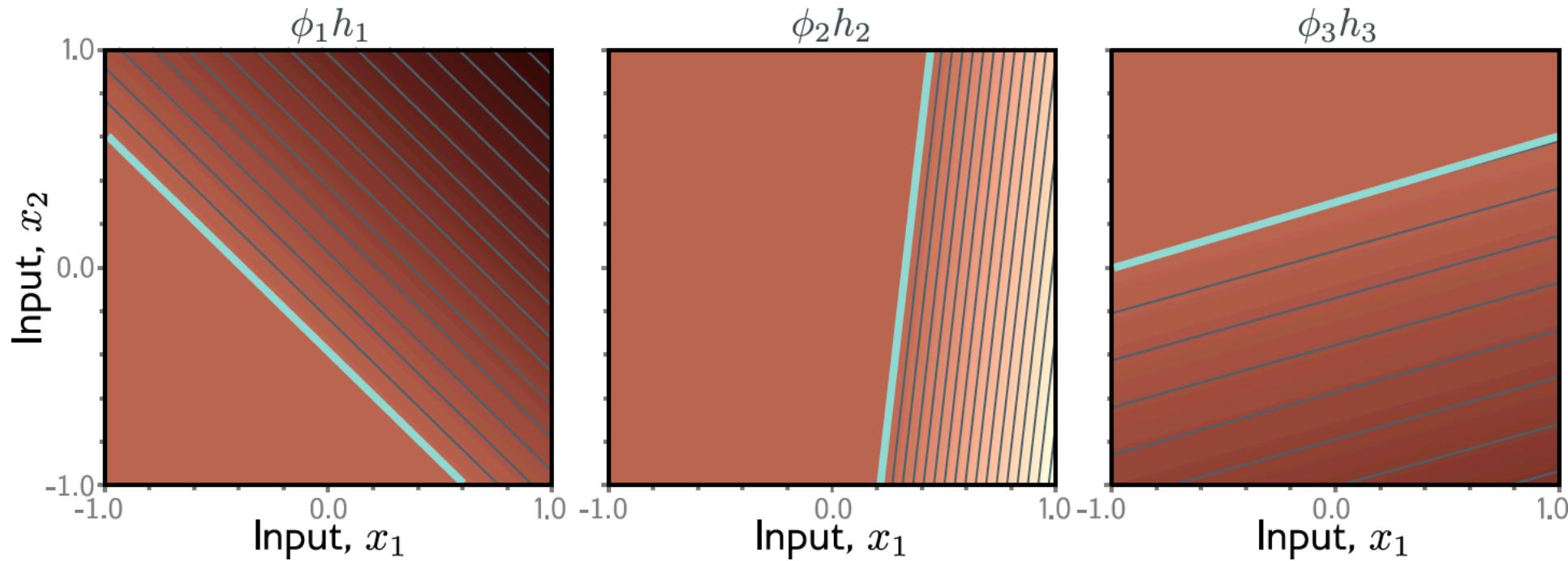








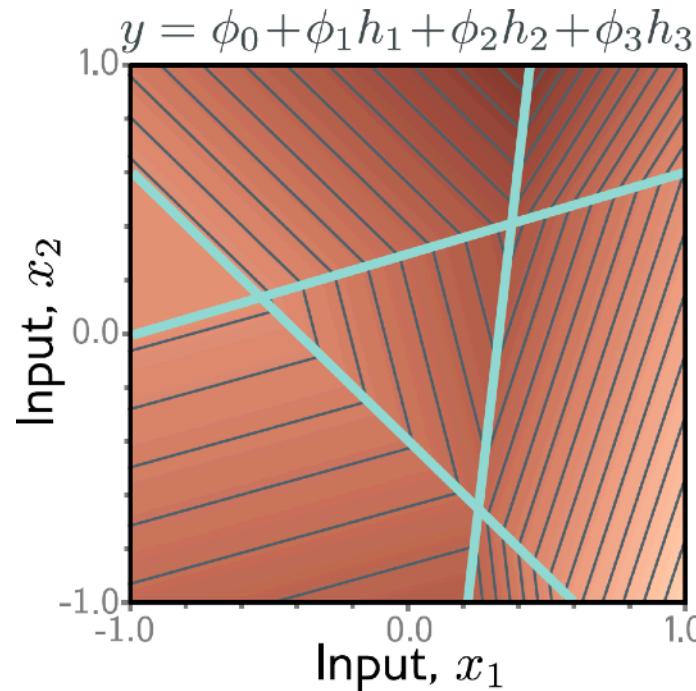




Convex polygons

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

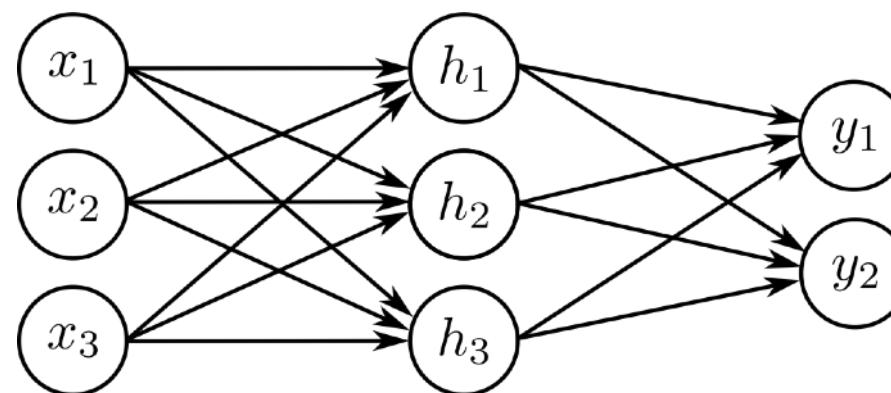
- Example network, 1 input, 1 output
- Universal approximation theorem
- More than one output
- More than one input
- General case
- Number of regions
- Terminology

Arbitrary inputs, hidden units, outputs

- D_o Outputs, D hidden units, and D_i inputs

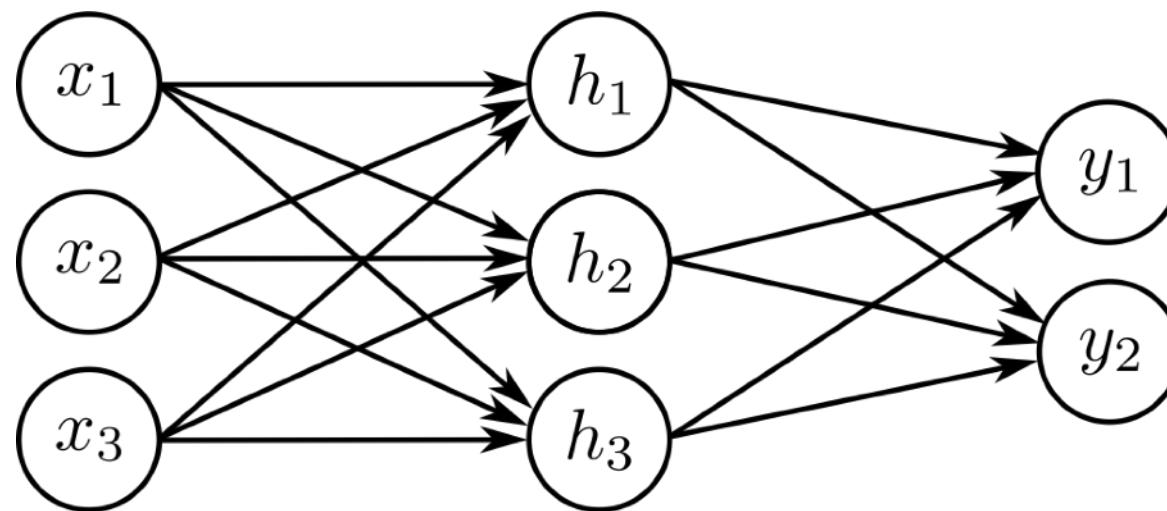
$$h_d = a \left[\theta_{d0} + \sum_{i=1}^{D_i} \theta_{di} x_i \right] \quad y_j = \phi_{j0} + \sum_{d=1}^D \phi_{jd} h_d$$

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



Question 2:

- How many parameters does this model have?



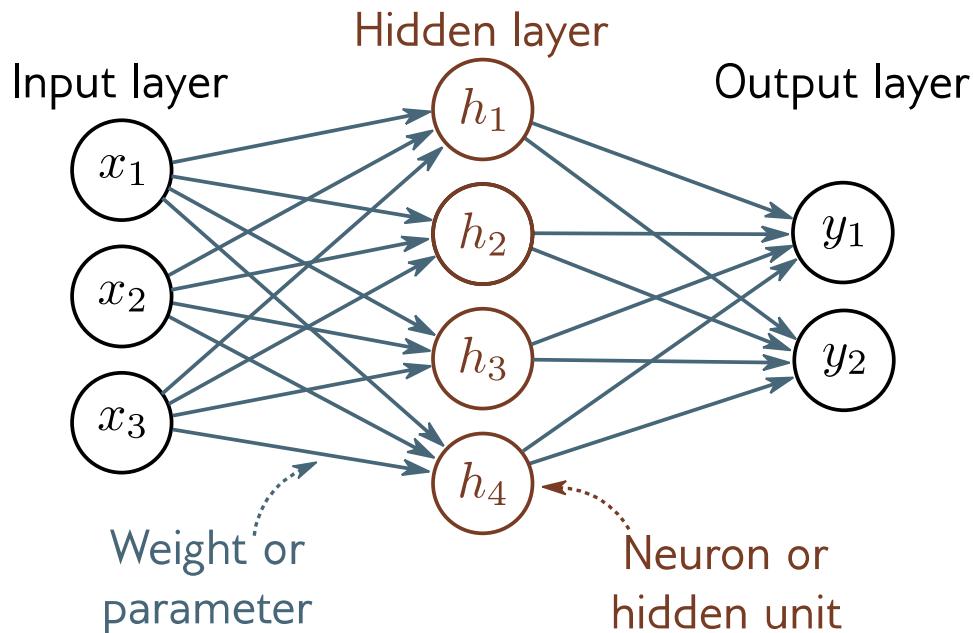
Shallow neural networks

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

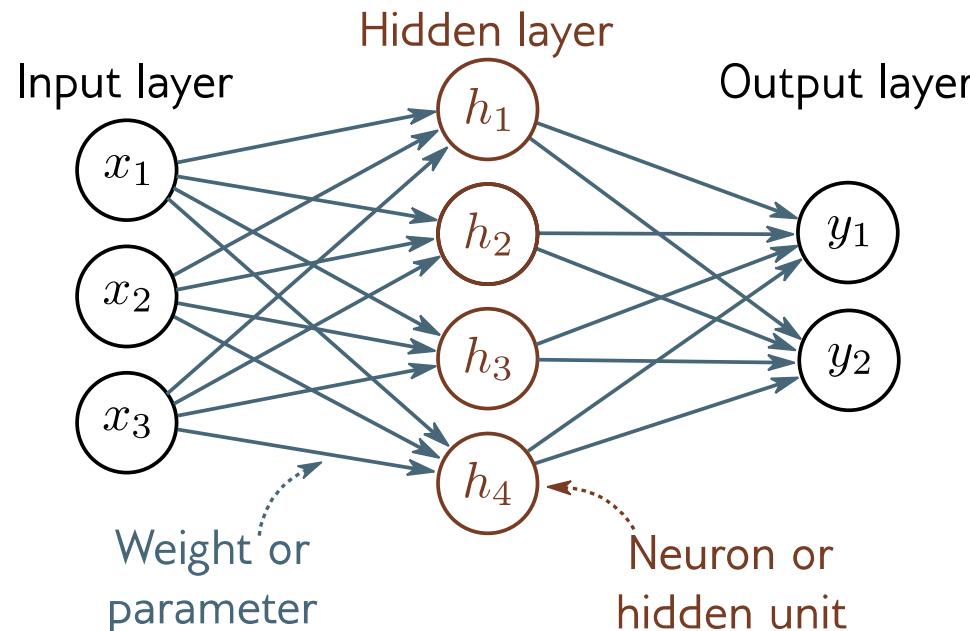
Shallow neural networks

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

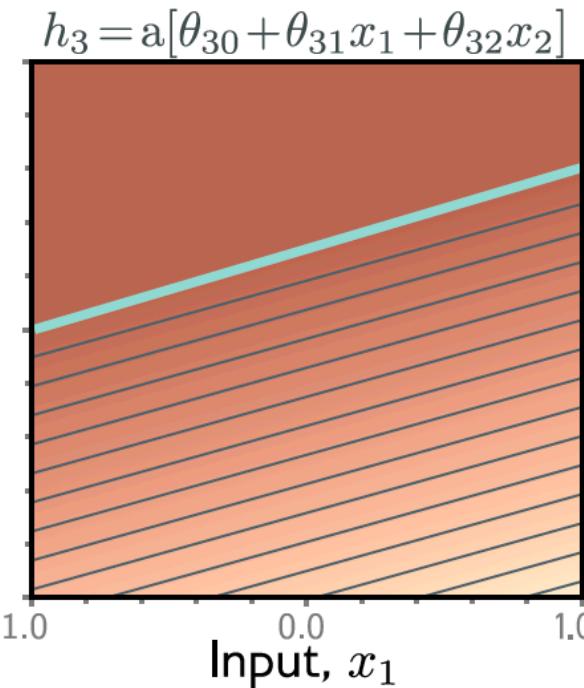
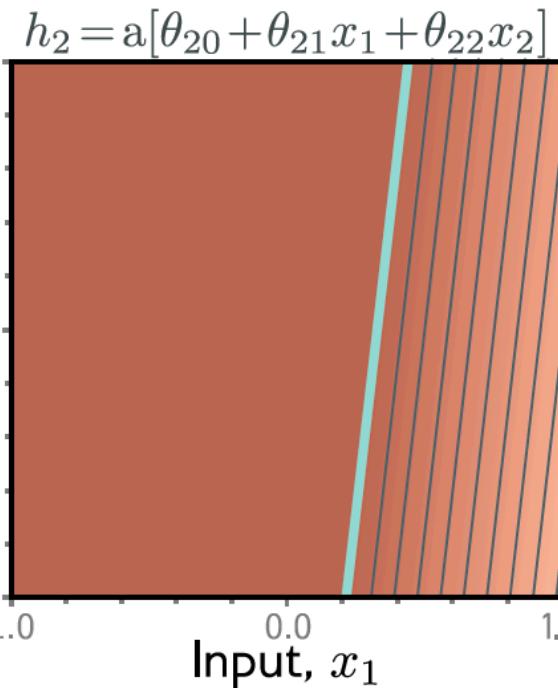
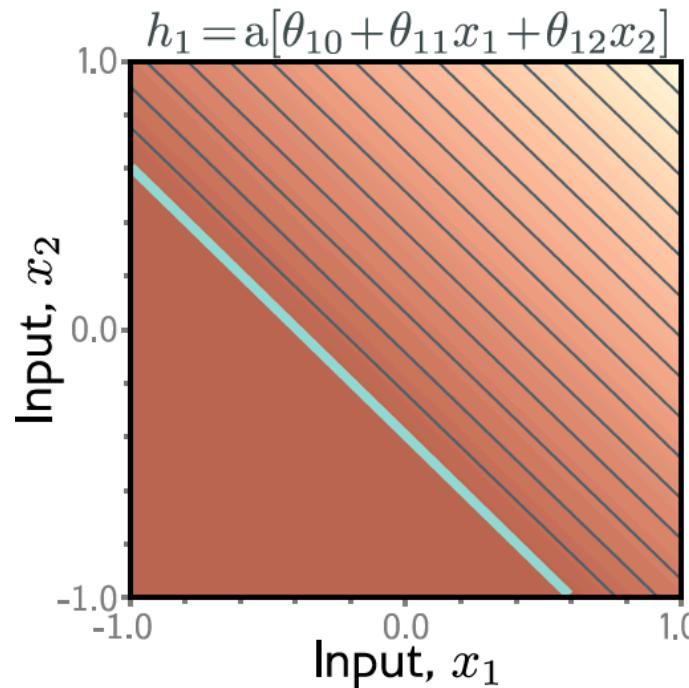
Nomenclature



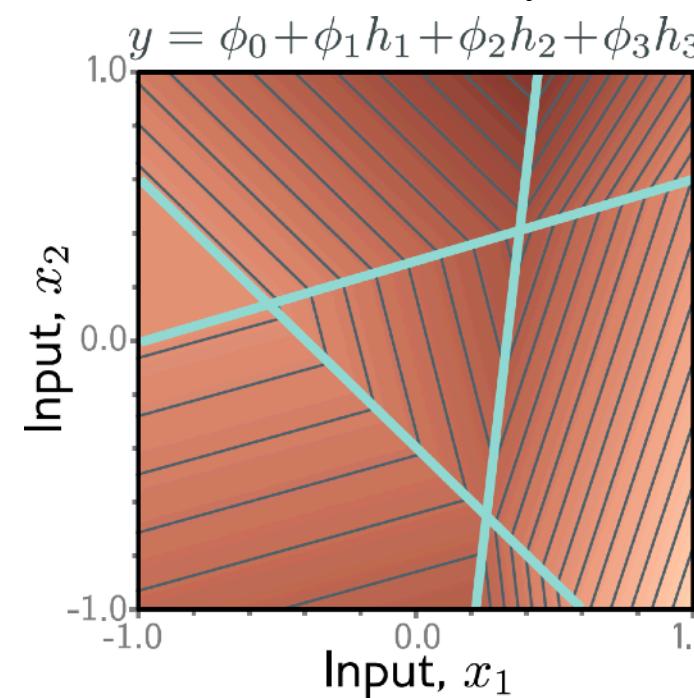
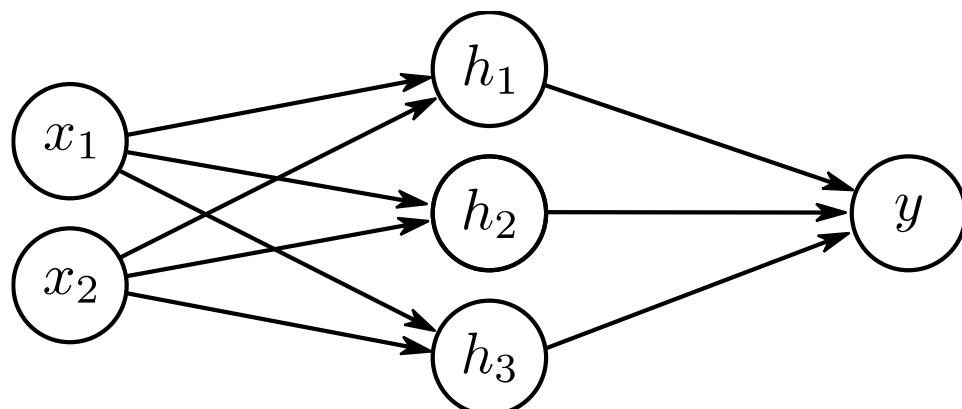
Nomenclature



- 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 \approx **capacity**



Scale each of these hidden units
to get final prediction manifold:



Deep neural networks

- Networks with more than one hidden layer
- Intuition becomes more difficult

Composing two networks.

Network 1:

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

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

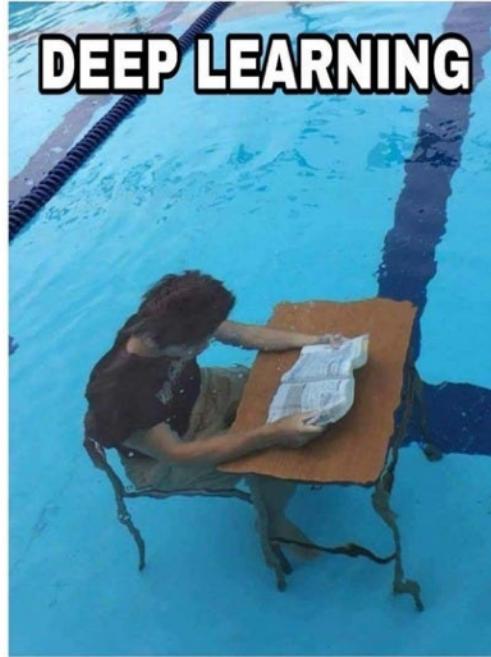
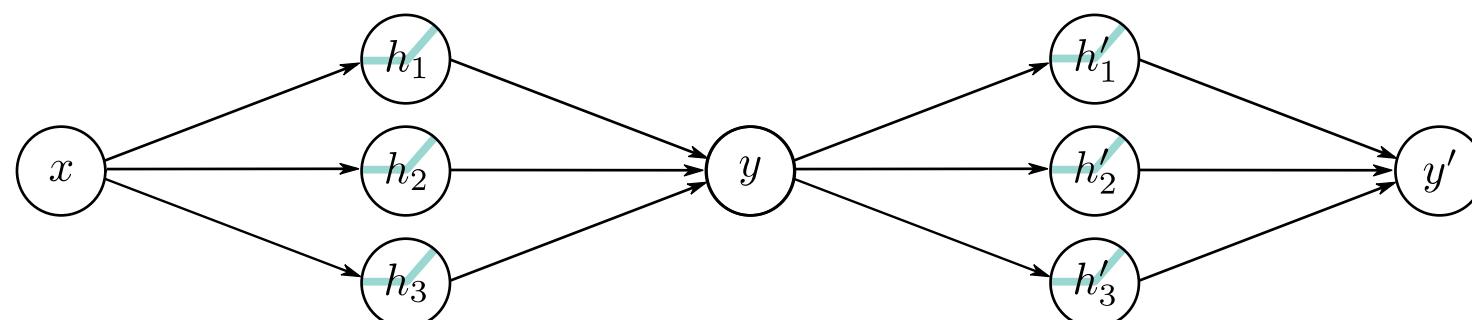
Network 2:

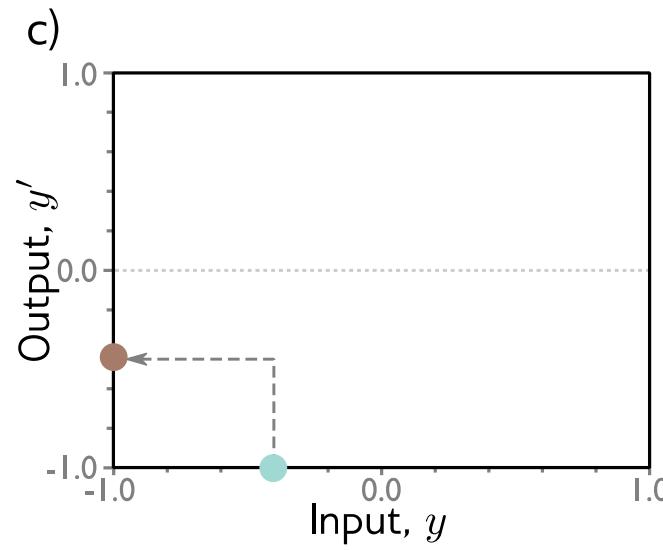
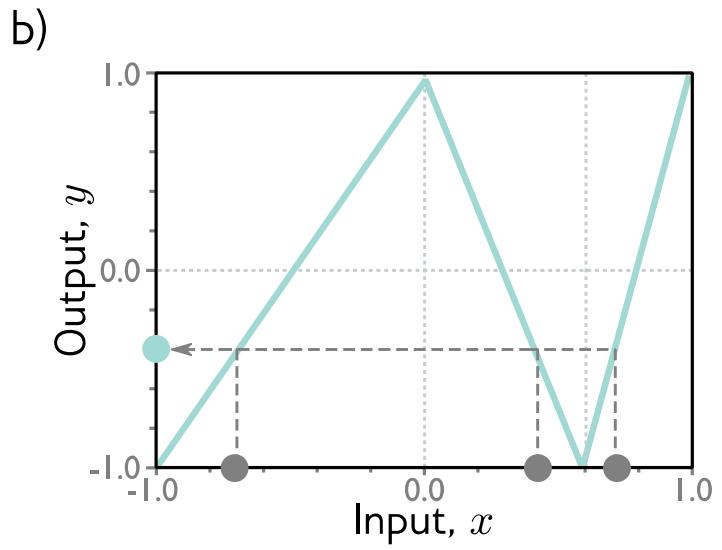
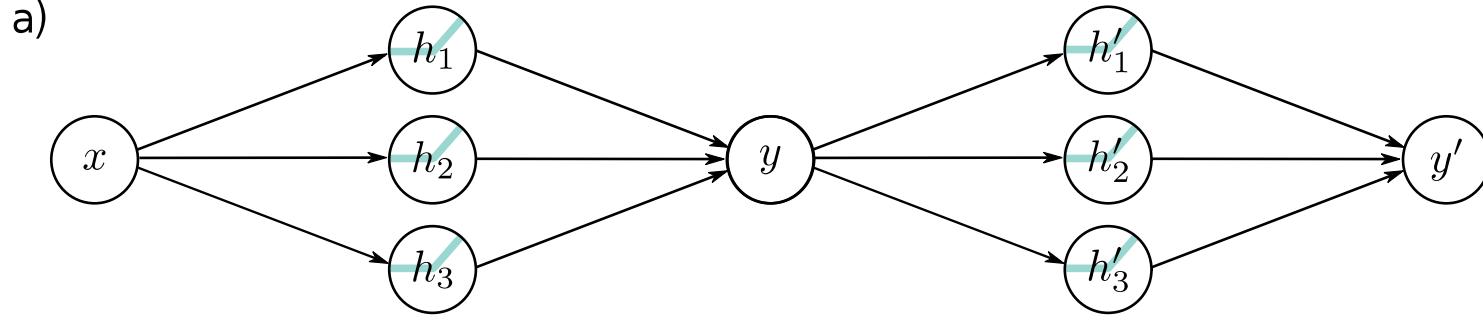
$$h'_1 = a[\theta'_{10} + \theta'_{11}y]$$

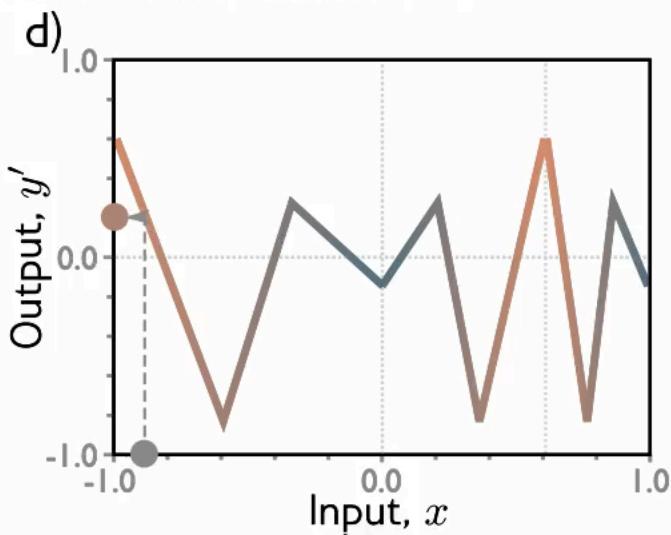
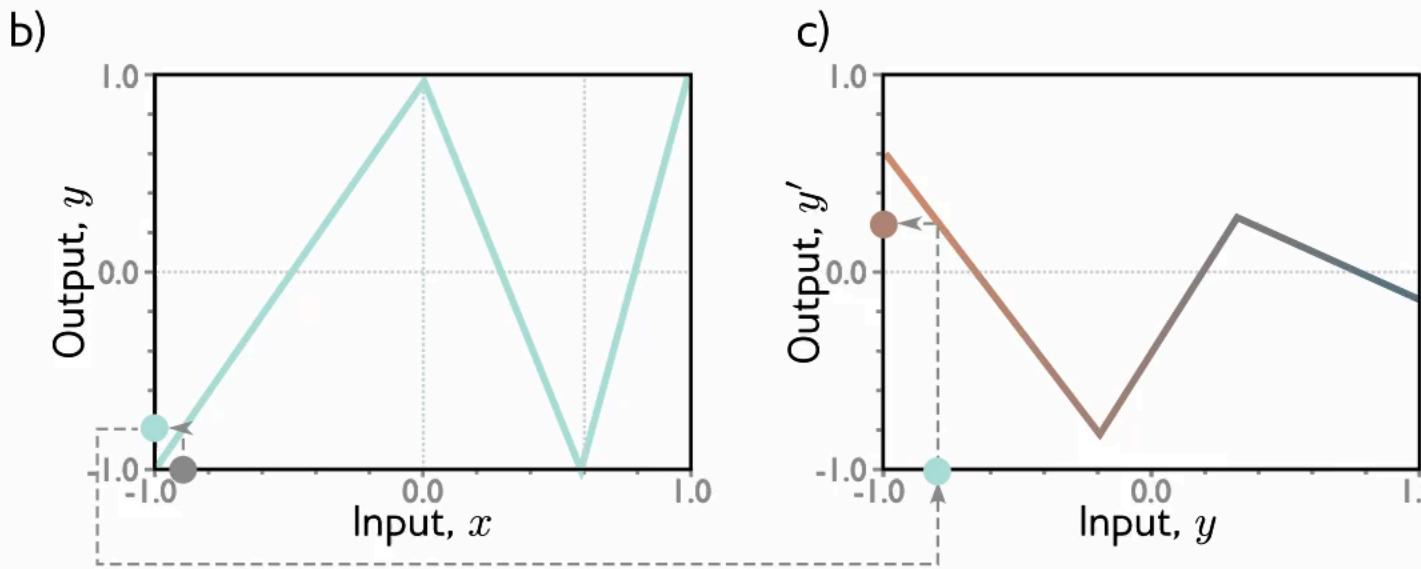
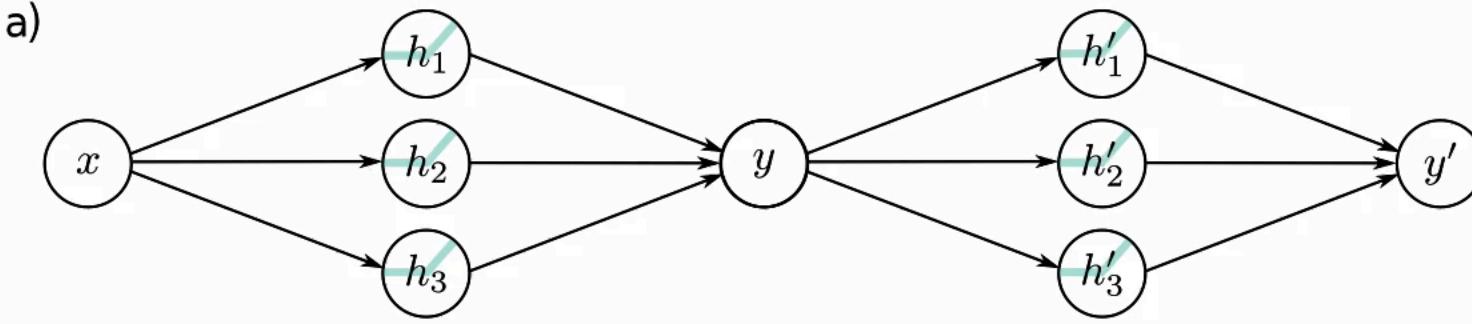
$$h'_2 = a[\theta'_{20} + \theta'_{21}y]$$

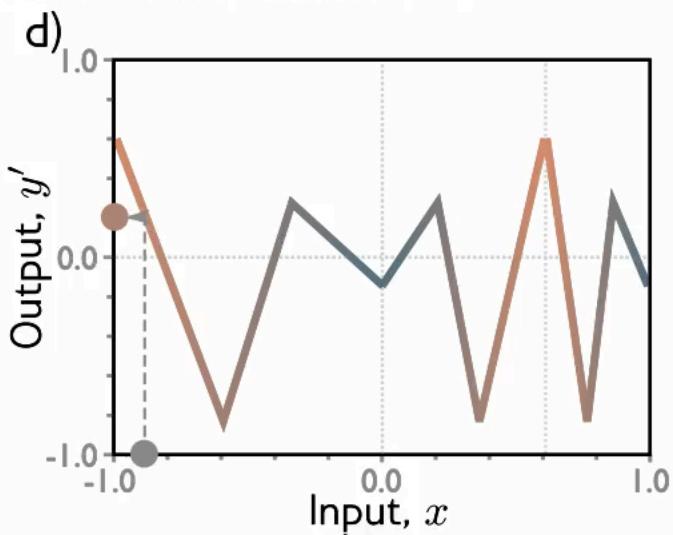
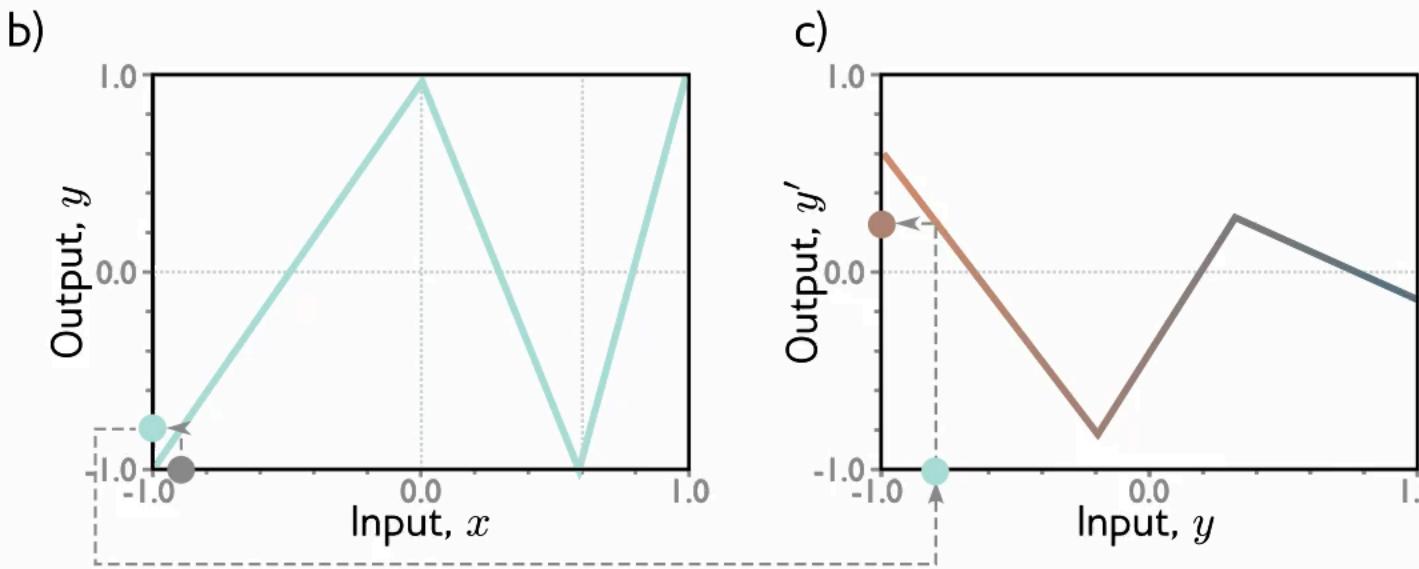
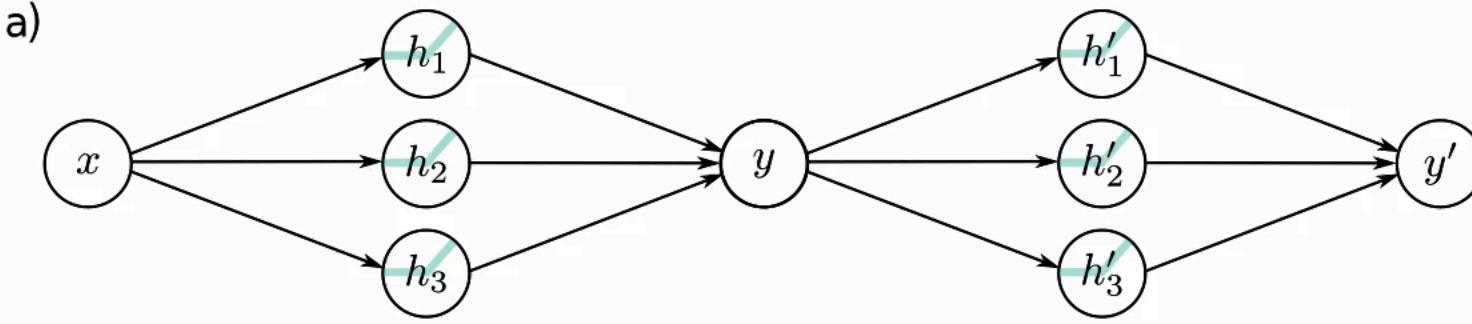
$$h'_3 = a[\theta'_{30} + \theta'_{31}y]$$

$$y' = \phi'_0 + \phi'_1 h'_1 + \phi'_2 h'_2 + \phi'_3 h'_3$$

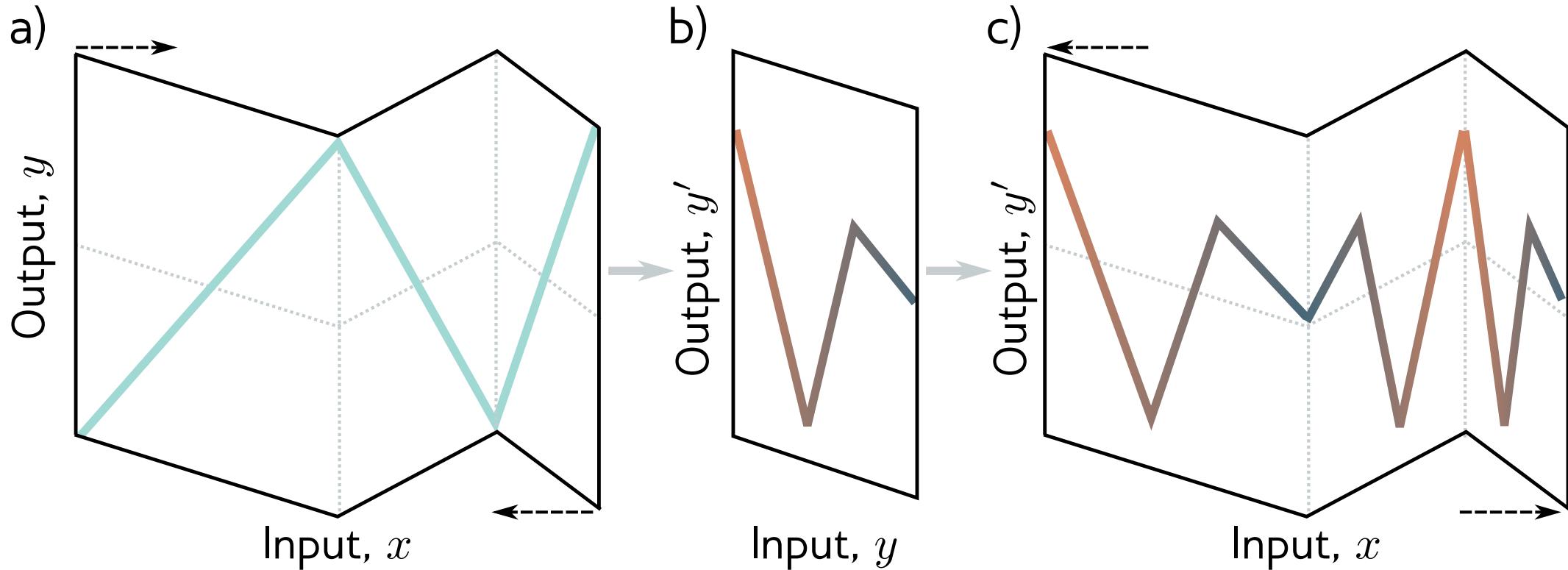




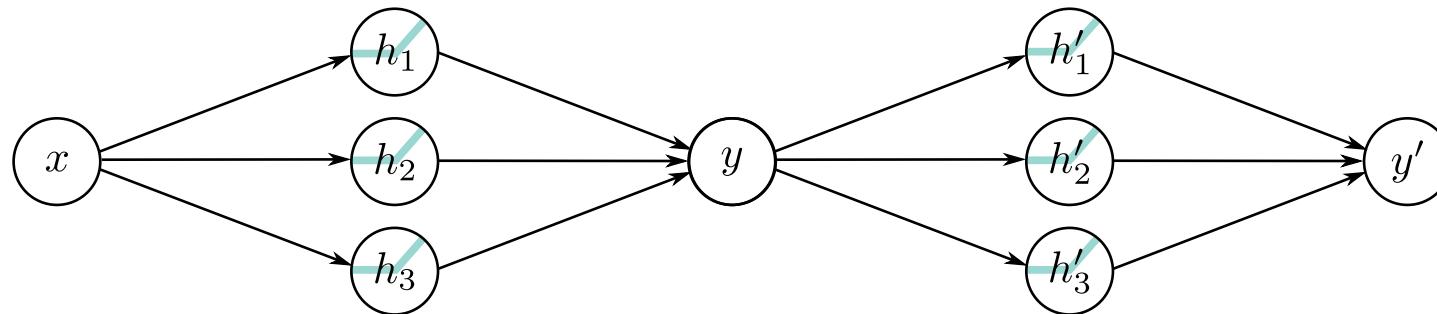




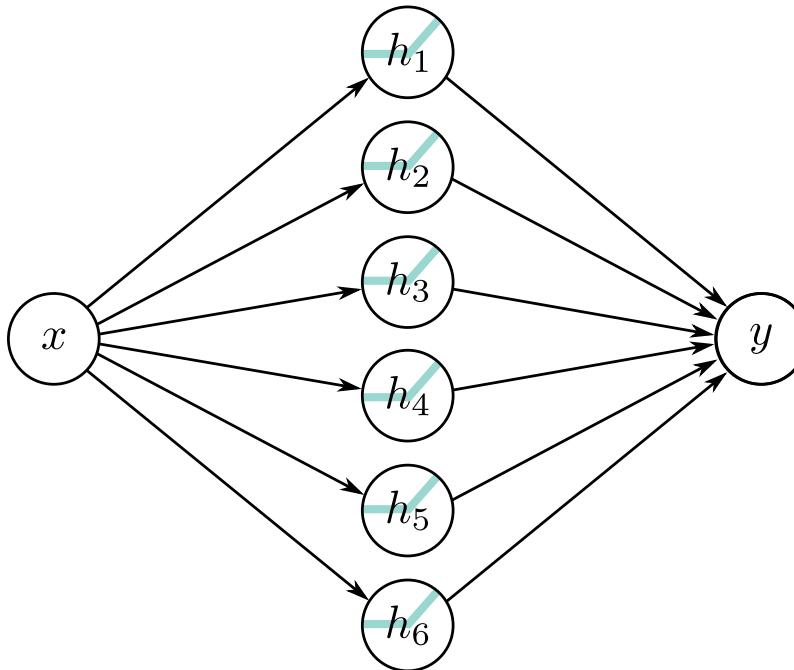
“Folding analogy”



Comparing to shallow with six hidden units

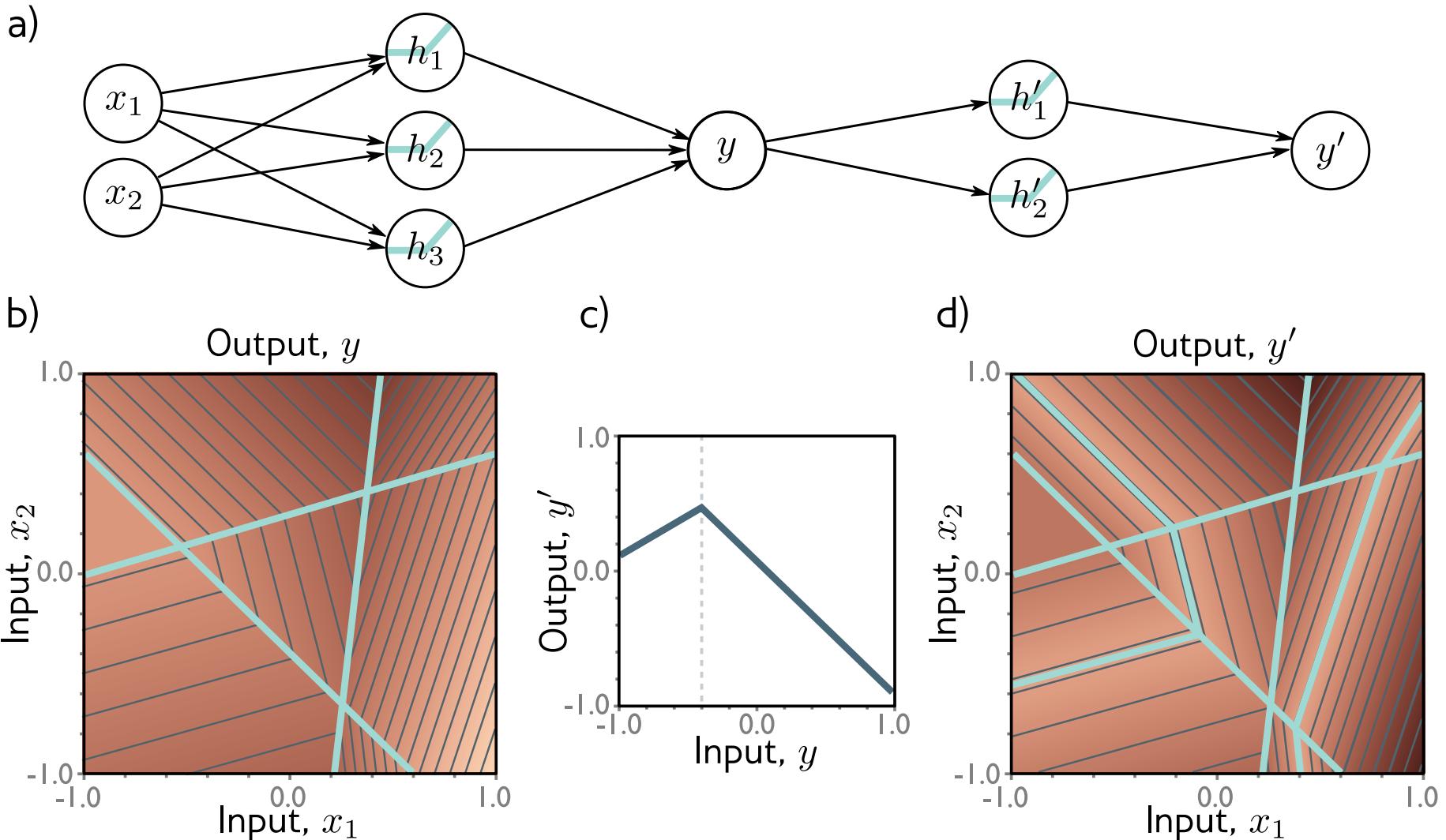


- 20 parameters
- (at least) 9 regions



- 19 parameters
- Max 7 regions

Composing networks in 2D



Combine two networks into one

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

Network 1:

$$\begin{aligned} h_2 &= a[\theta_{20} + \theta_{21}x] & y &= \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3 \\ h_3 &= a[\theta_{30} + \theta_{31}x] \end{aligned}$$

$$h'_1 = a[\theta'_{10} + \theta'_{11}y]$$

Network 2:

$$\begin{aligned} h'_2 &= a[\theta'_{20} + \theta'_{21}y] & y' &= \phi'_0 + \phi'_1 h'_1 + \phi'_2 h'_2 + \phi'_3 h'_3 \\ h'_3 &= a[\theta'_{30} + \theta'_{31}y] \end{aligned}$$

Hidden units of second network in terms of first:

$$h'_1 = a[\theta'_{10} + \theta'_{11}y] = a[\theta'_{10} + \theta'_{11}\phi_0 + \theta'_{11}\phi_1 h_1 + \theta'_{11}\phi_2 h_2 + \theta'_{11}\phi_3 h_3]$$

$$h'_2 = a[\theta'_{20} + \theta'_{21}y] = a[\theta'_{20} + \theta'_{21}\phi_0 + \theta'_{21}\phi_1 h_1 + \theta'_{21}\phi_2 h_2 + \theta'_{21}\phi_3 h_3]$$

$$h'_3 = a[\theta'_{30} + \theta'_{31}y] = a[\theta'_{30} + \theta'_{31}\phi_0 + \theta'_{31}\phi_1 h_1 + \theta'_{31}\phi_2 h_2 + \theta'_{31}\phi_3 h_3]$$

Create new variables

$$h'_1 = a[\theta'_{10} + \theta'_{11}y] = a[\theta'_{10} + \theta'_{11}\phi_0 + \theta'_{11}\phi_1h_1 + \theta'_{11}\phi_2h_2 + \theta'_{11}\phi_3h_3]$$

$$h'_2 = a[\theta'_{20} + \theta'_{21}y] = a[\theta'_{20} + \theta'_{21}\phi_0 + \theta'_{21}\phi_1h_1 + \theta'_{21}\phi_2h_2 + \theta'_{21}\phi_3h_3]$$

$$h'_3 = a[\theta'_{30} + \theta'_{31}y] = a[\theta'_{30} + \theta'_{31}\phi_0 + \theta'_{31}\phi_1h_1 + \theta'_{31}\phi_2h_2 + \theta'_{31}\phi_3h_3]$$

$$h'_1 = a[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$

$$h'_2 = a[\psi_{20} + \psi_{21}h_1 + \psi_{22}h_2 + \psi_{23}h_3]$$

$$h'_3 = a[\psi_{30} + \psi_{31}h_1 + \psi_{32}h_2 + \psi_{33}h_3]$$

Two-layer network

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

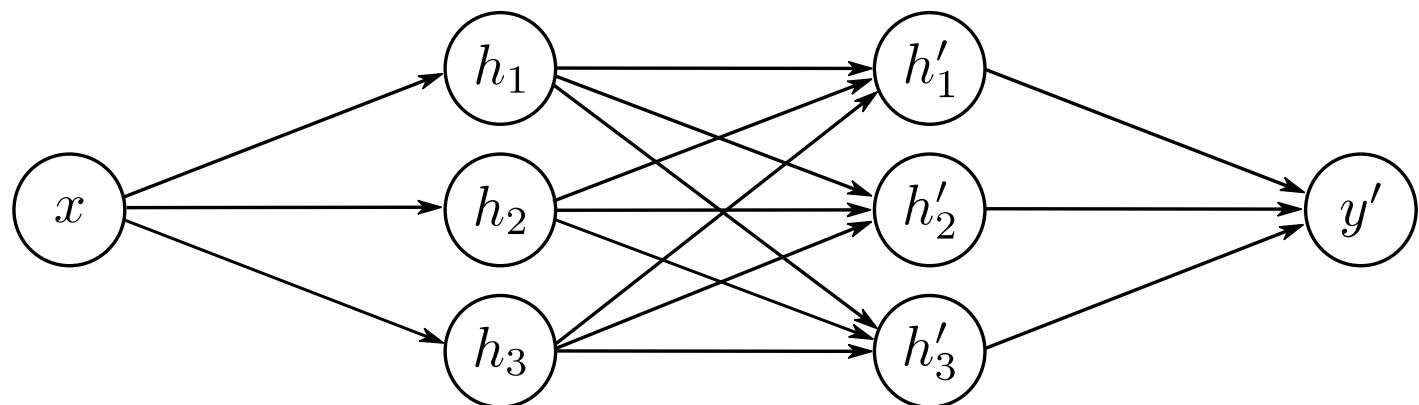
$$h_3 = a[\theta_{30} + \theta_{31}x]$$

$$h'_1 = a[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$

$$h'_2 = a[\psi_{20} + \psi_{21}h_2 + \psi_{22}h_2 + \psi_{23}h_3]$$

$$h'_3 = a[\psi_{30} + \psi_{31}h_2 + \psi_{32}h_2 + \psi_{33}h_3]$$

$$y' = \phi'_0 + \phi'_1 h'_1 + \phi'_2 h'_2 + \phi'_3 h'_3$$



Two-layer network as one equation

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

$$h'_1 = a[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$

$$h'_2 = a[\psi_{20} + \psi_{21}h_1 + \psi_{22}h_2 + \psi_{23}h_3]$$

$$h'_3 = a[\psi_{30} + \psi_{31}h_1 + \psi_{32}h_2 + \psi_{33}h_3]$$

$$y' = \phi'_0 + \phi'_1 h'_1 + \phi'_2 h'_2 + \phi'_3 h'_3$$

$$\begin{aligned} y' = & \phi'_0 + \phi'_1 a [\psi_{10} + \psi_{11}a[\theta_{10} + \theta_{11}x] + \psi_{12}a[\theta_{20} + \theta_{21}x] + \psi_{13}a[\theta_{30} + \theta_{31}x]] \\ & + \phi'_2 a [\psi_{20} + \psi_{21}a[\theta_{10} + \theta_{11}x] + \psi_{22}a[\theta_{20} + \theta_{21}x] + \psi_{23}a[\theta_{30} + \theta_{31}x]] \\ & + \phi'_3 a [\psi_{30} + \psi_{31}a[\theta_{10} + \theta_{11}x] + \psi_{32}a[\theta_{20} + \theta_{21}x] + \psi_{33}a[\theta_{30} + \theta_{31}x]] \end{aligned}$$

Networks as composing functions

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

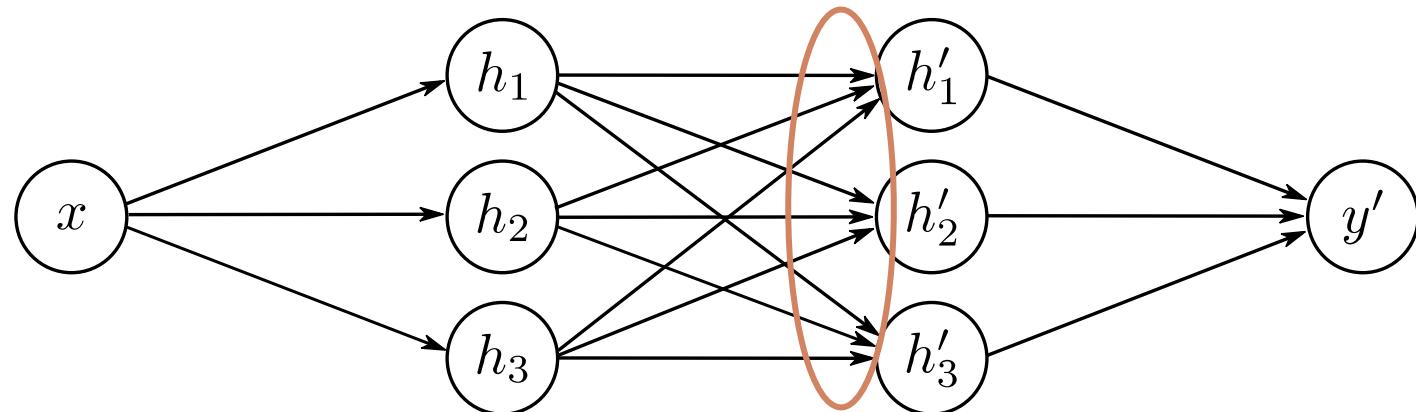
$$h'_1 = a[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$

$$h'_2 = a[\psi_{20} + \psi_{21}h_1 + \psi_{22}h_2 + \psi_{23}h_3]$$

$$h'_3 = a[\psi_{30} + \psi_{31}h_1 + \psi_{32}h_2 + \psi_{33}h_3]$$

Consider the pre-activations at the second hidden units

At this point, it's a one-layer network with three outputs



Networks as composing functions

$$h_1 = a[\theta_{10} + \theta_{11}x]$$

$$h_2 = a[\theta_{20} + \theta_{21}x]$$

$$h_3 = a[\theta_{30} + \theta_{31}x]$$

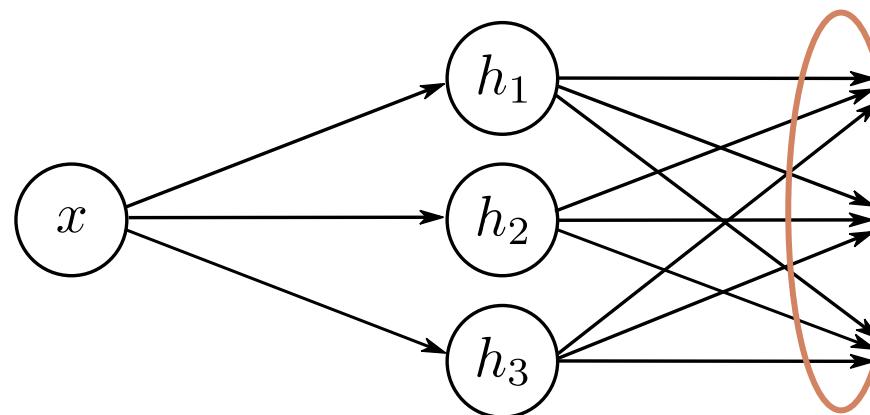
$$h'_1 = a[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$

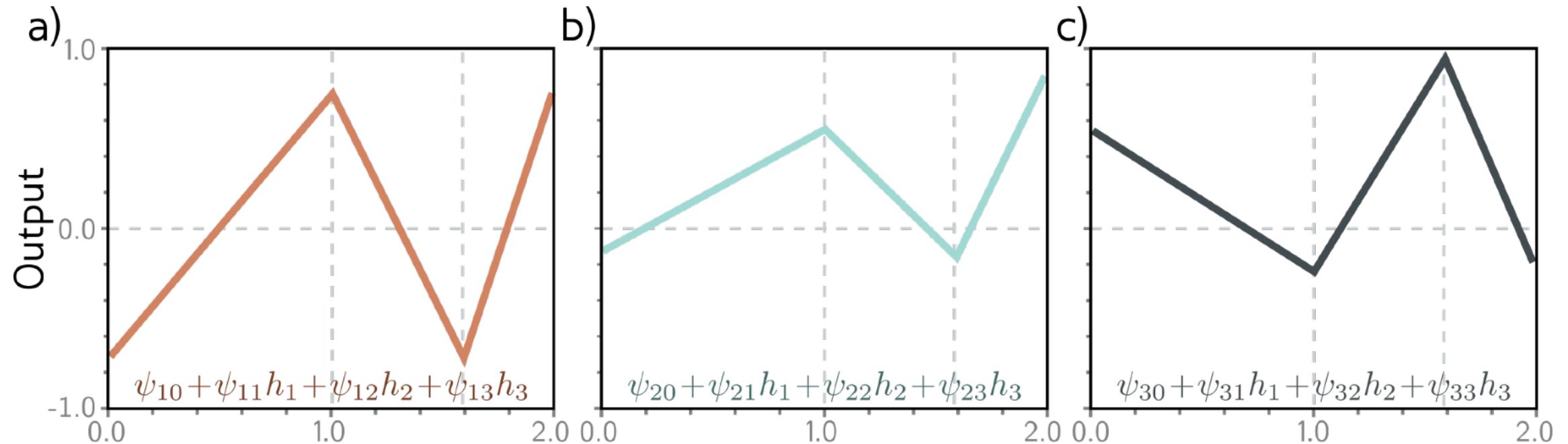
$$h'_2 = a[\psi_{20} + \psi_{21}h_1 + \psi_{22}h_2 + \psi_{23}h_3]$$

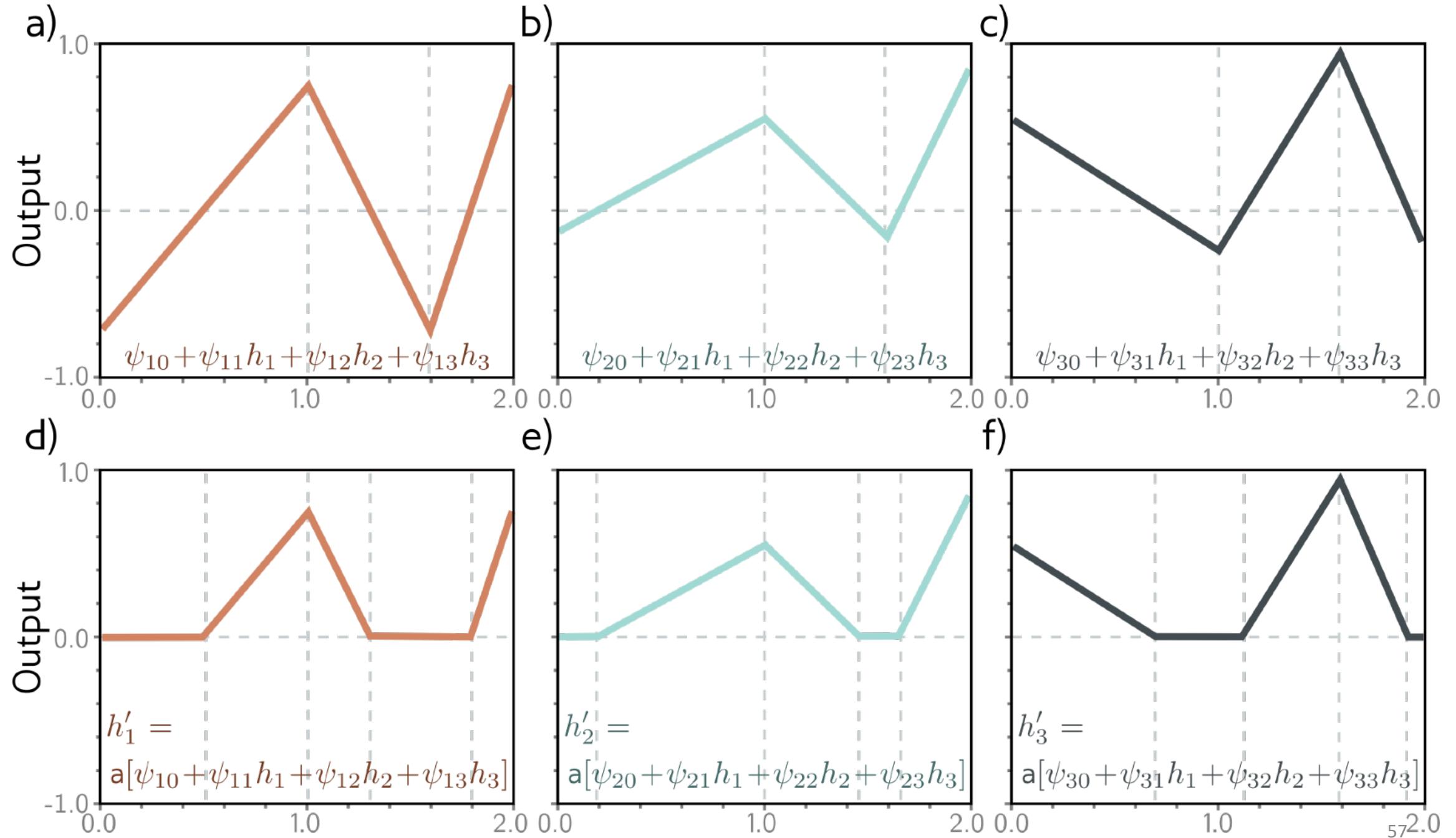
$$h'_3 = a[\psi_{30} + \psi_{31}h_1 + \psi_{32}h_2 + \psi_{33}h_3]$$

Consider the pre-activations at the second hidden units

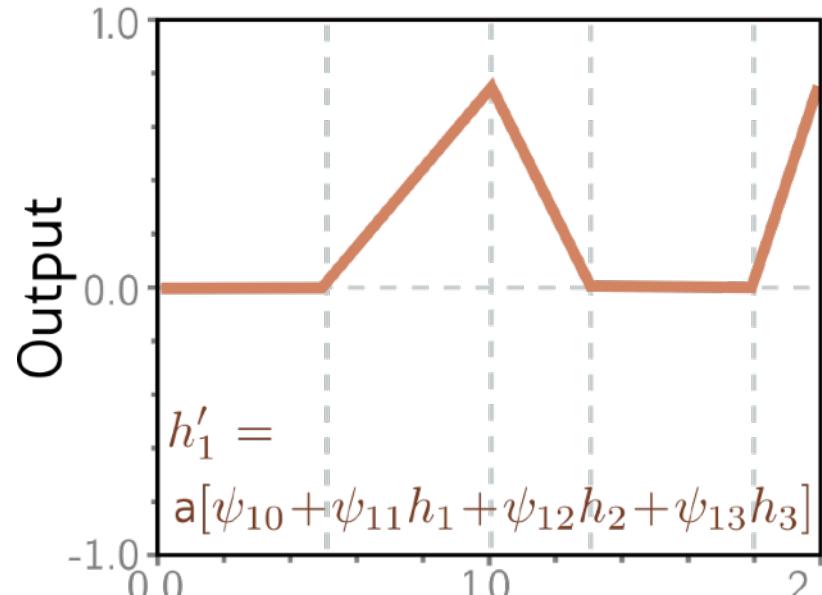
At this point, it's a one-layer network with three outputs



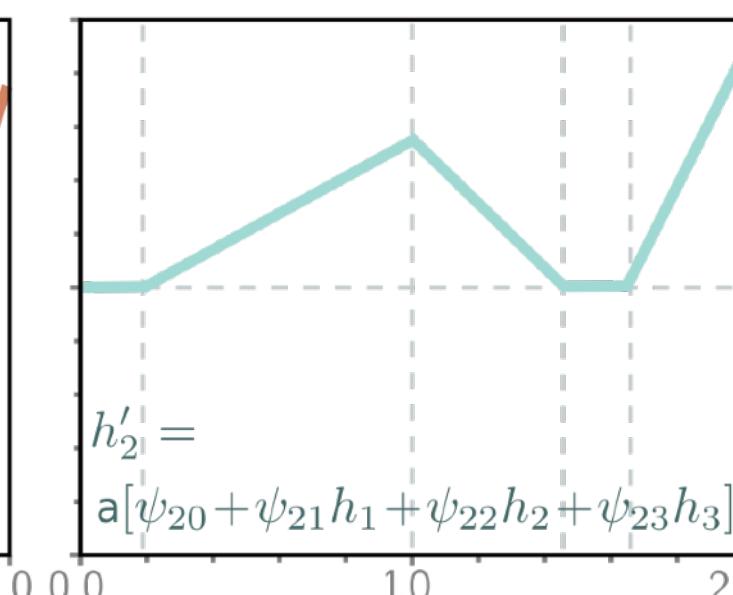




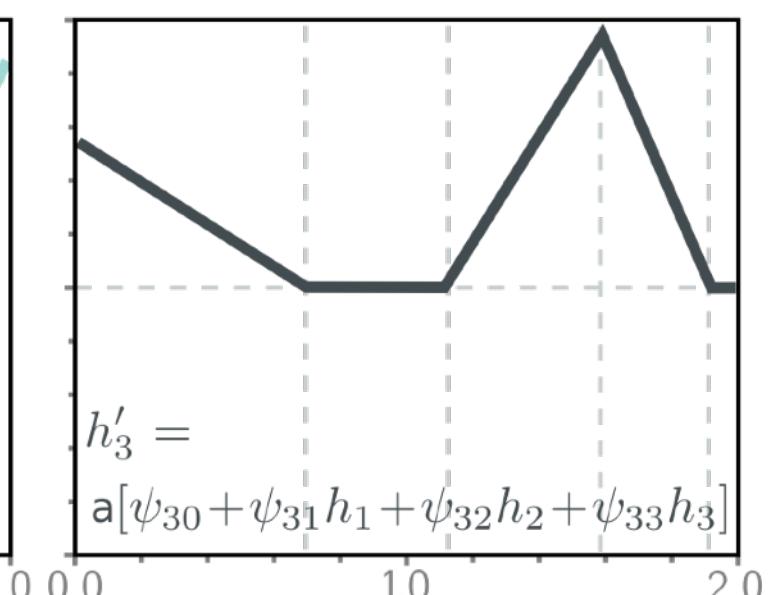
d)



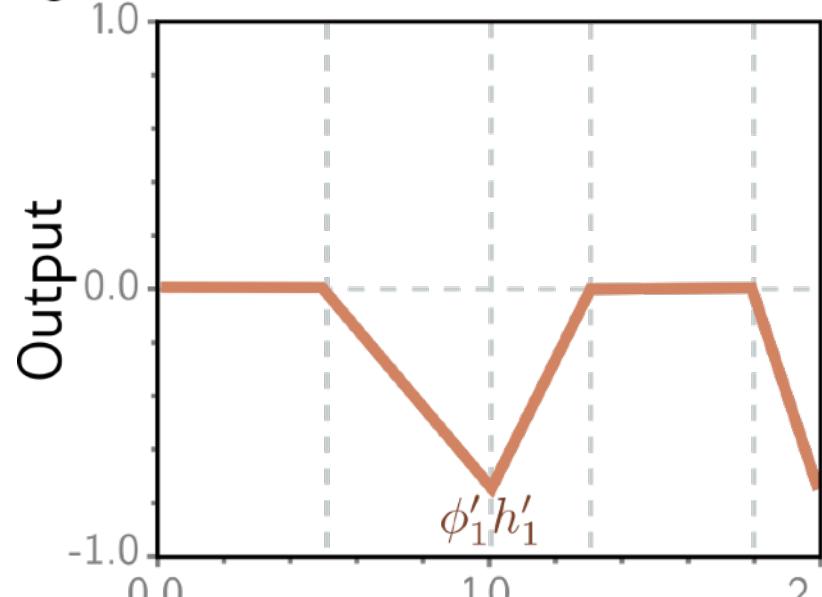
e)



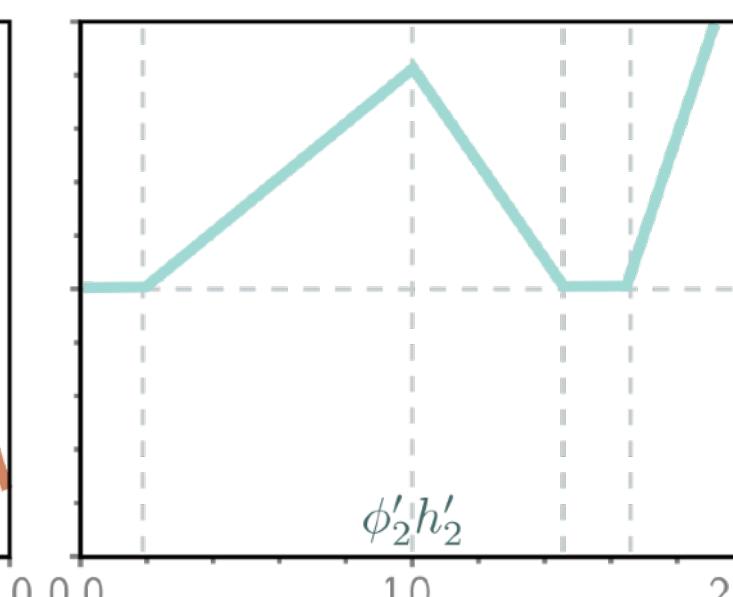
f)



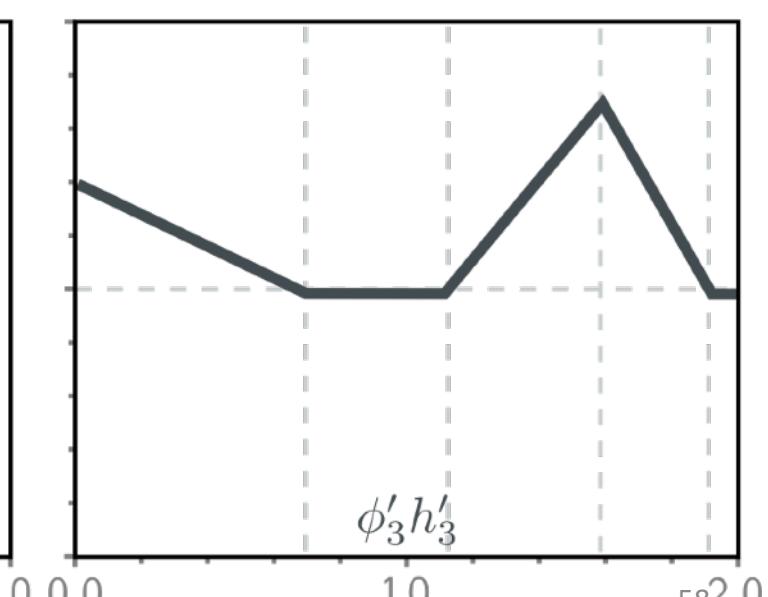
g)

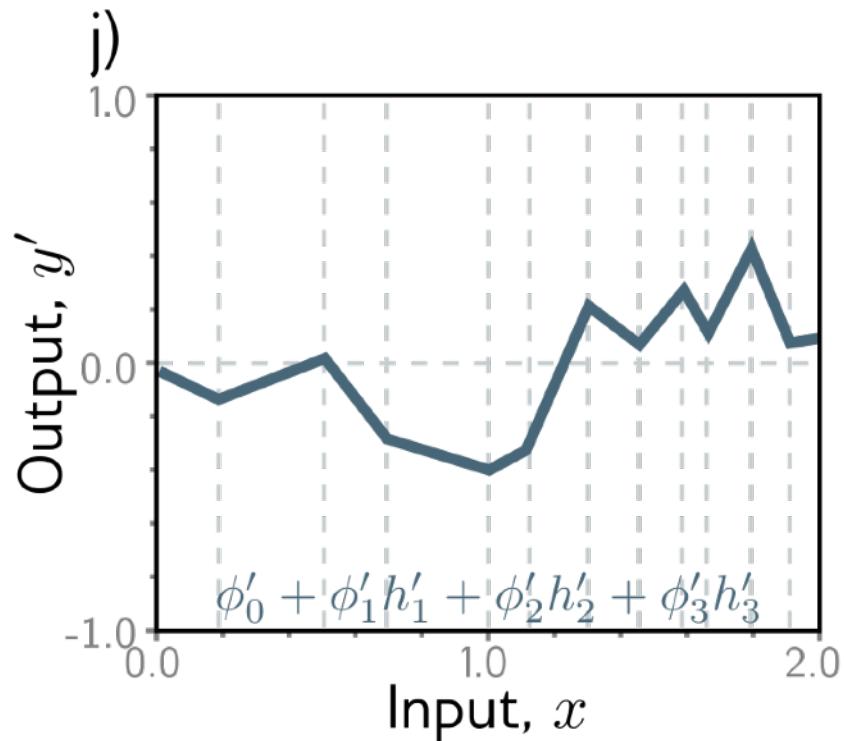
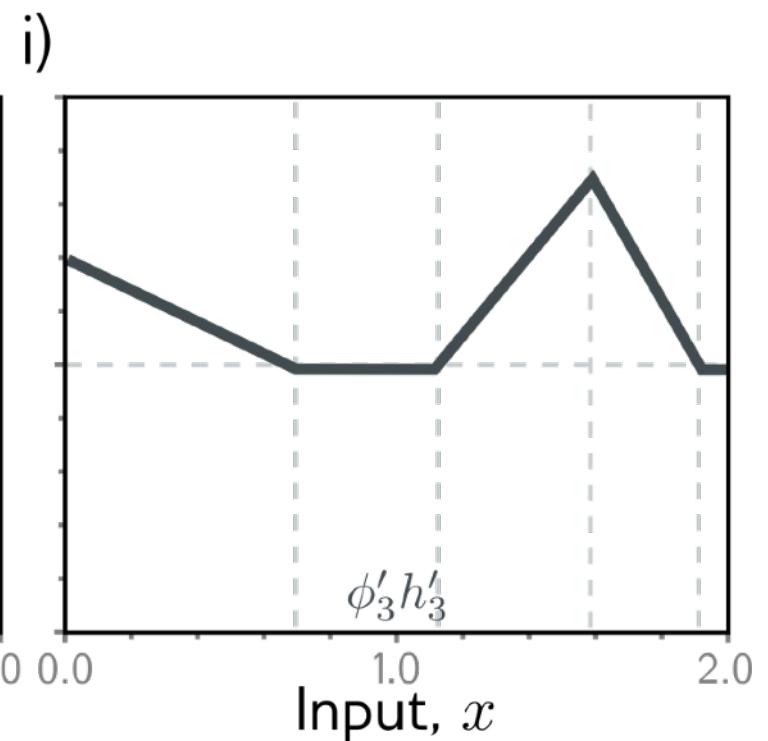
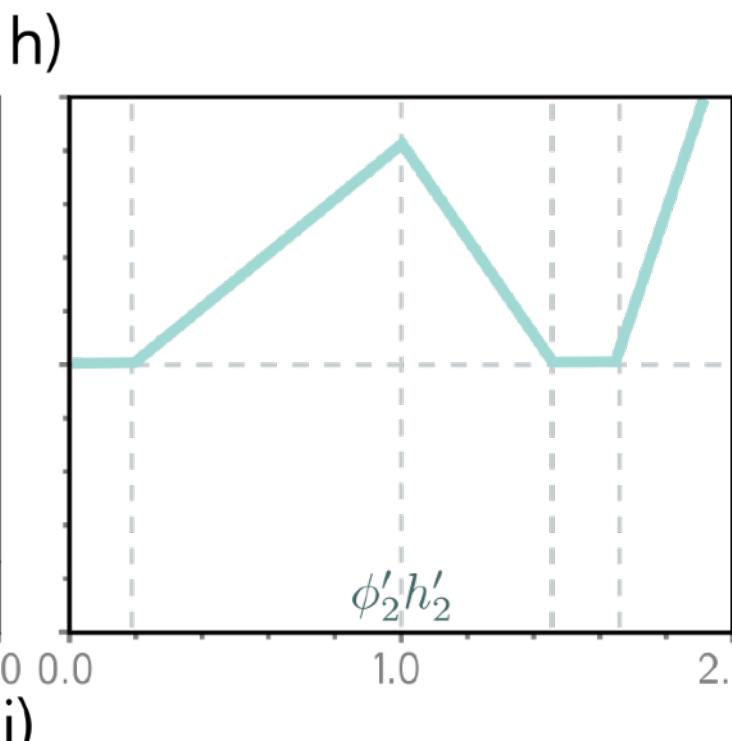
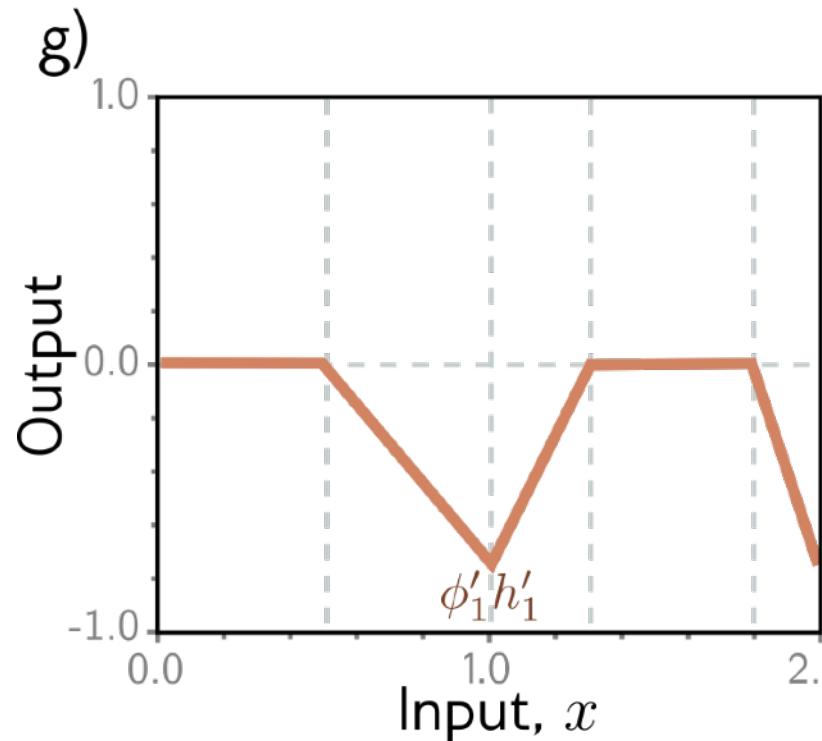


h)



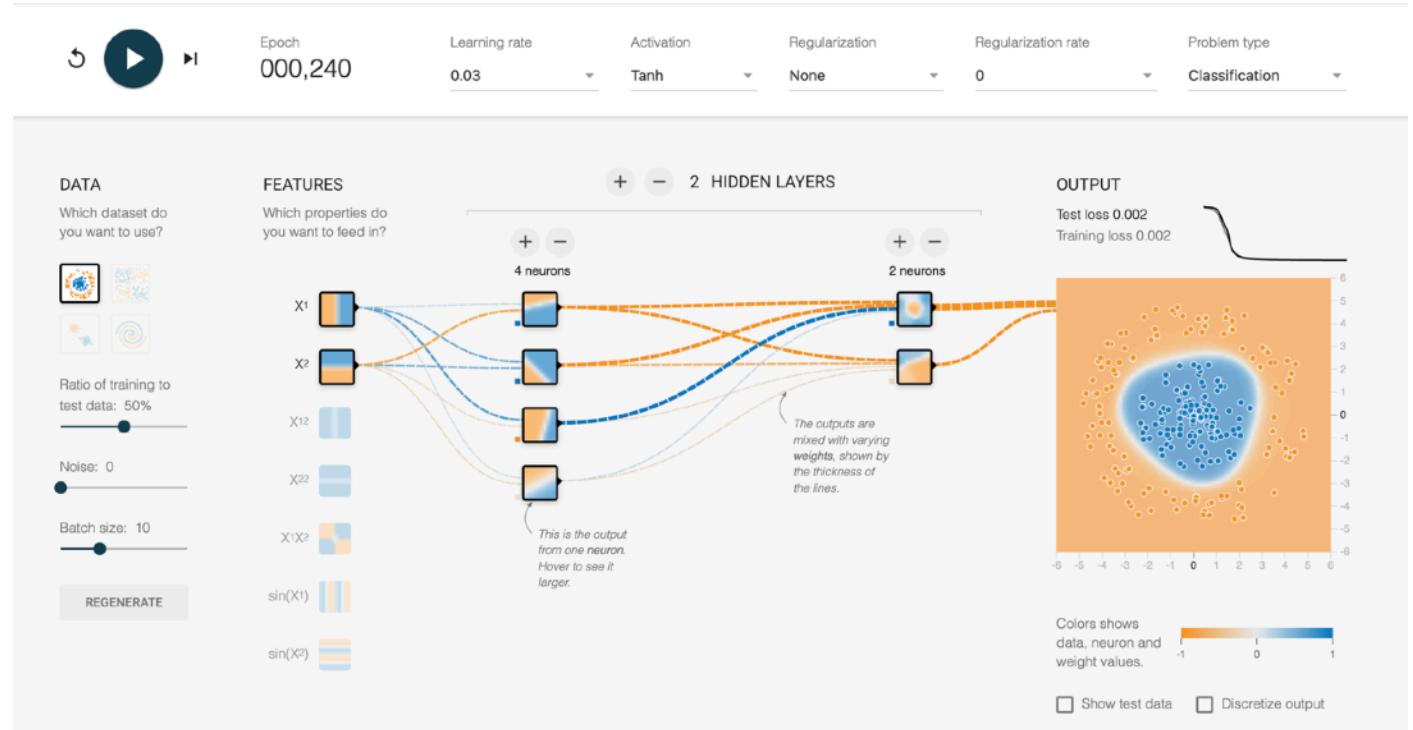
i)



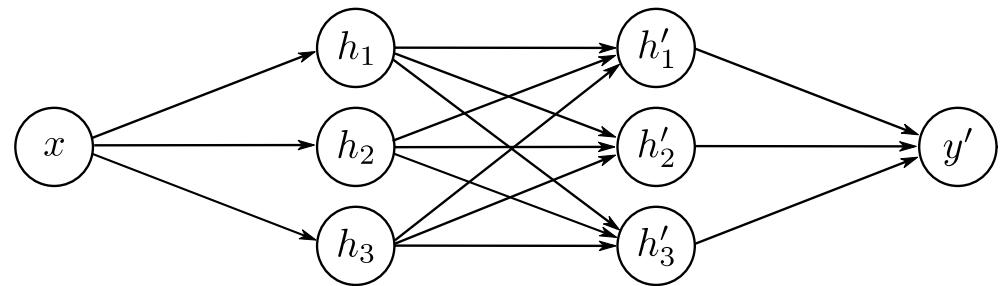


Interactive visualisation

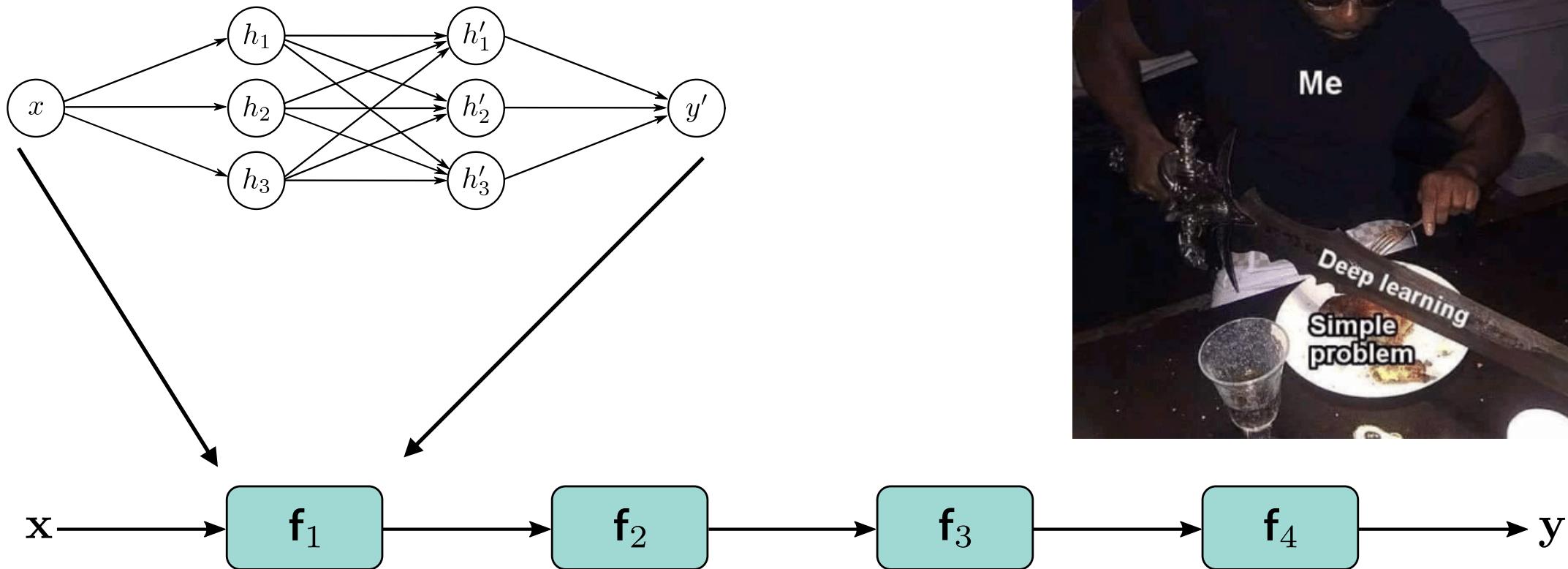
<https://playground.tensorflow.org/>



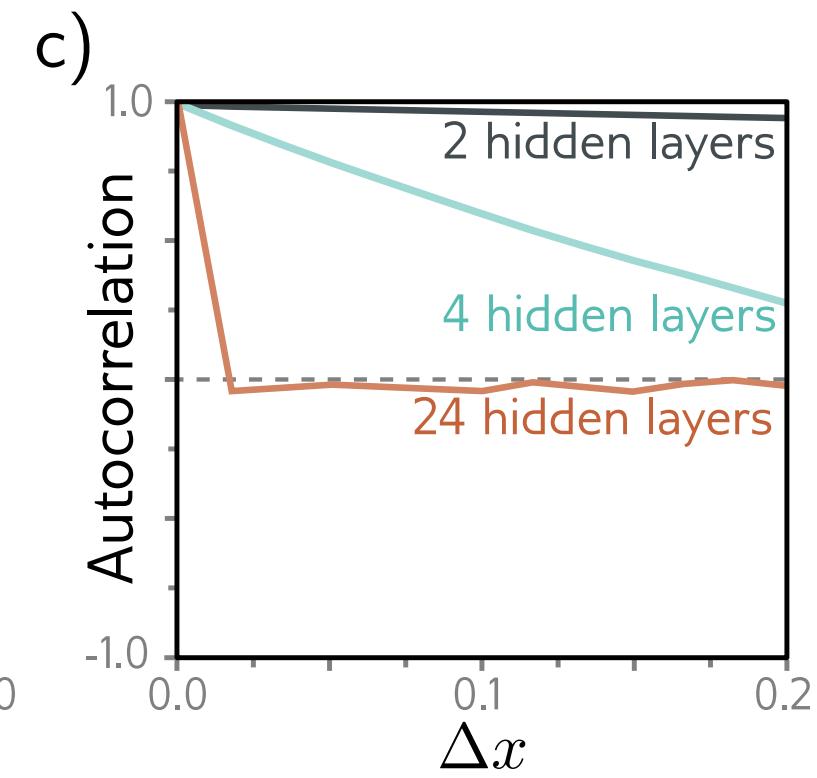
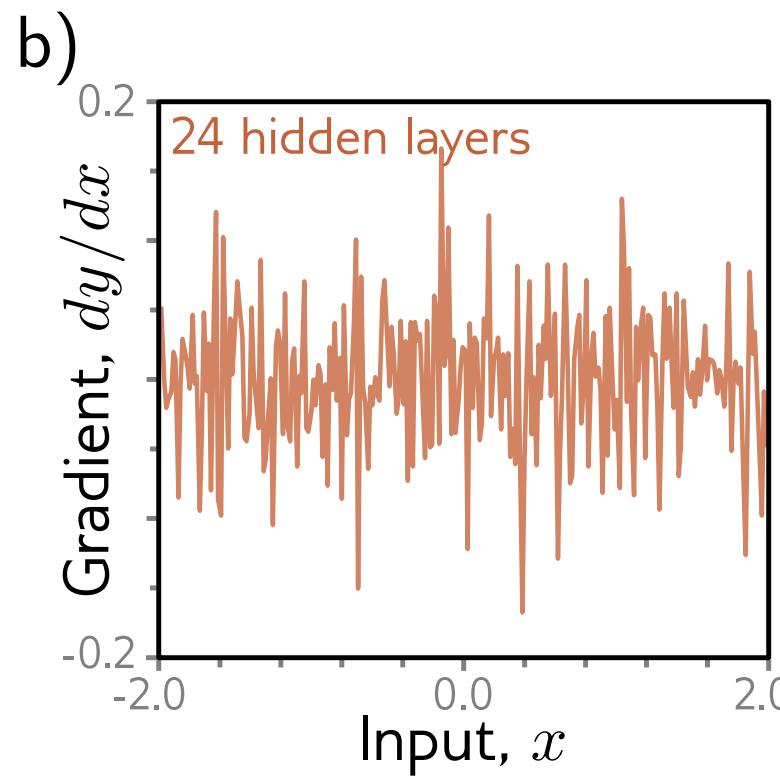
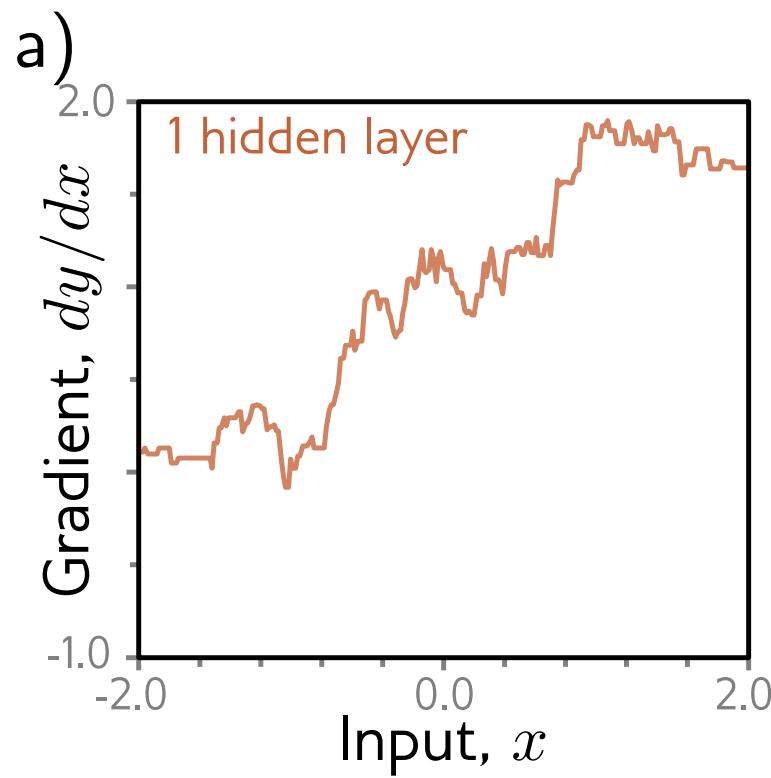
Multi-layer perceptrons as a “building block”

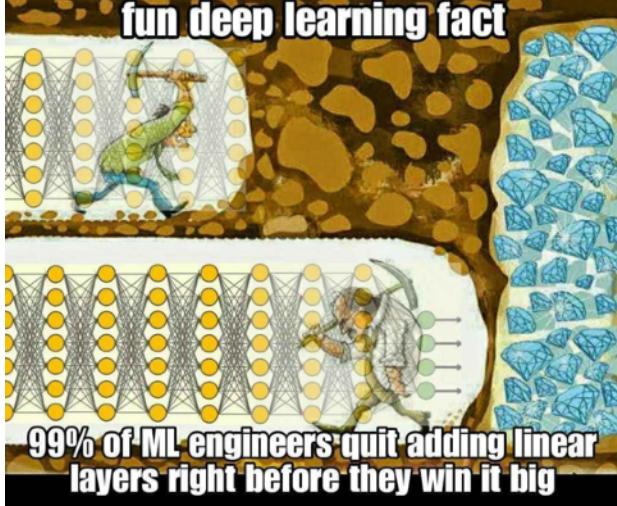


Multi-layer perceptrons as a “building block”

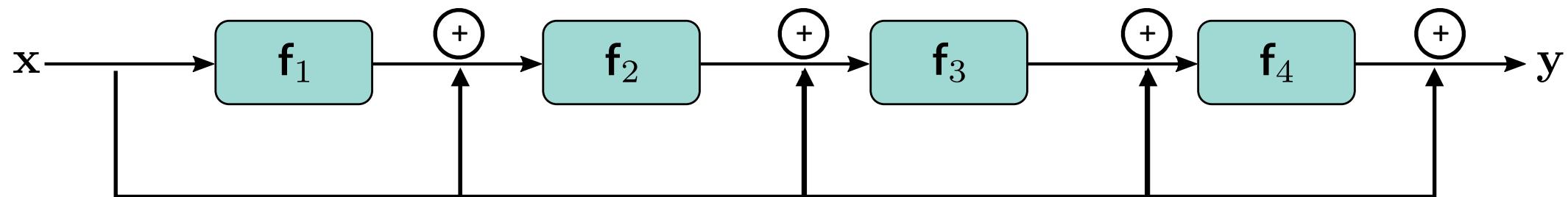
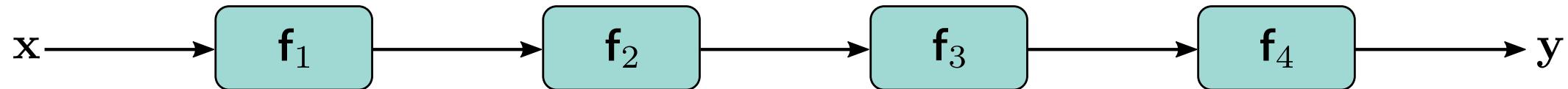


Very very deep networks





Strategy: “residual connections”



This “stabilises” very deep networks