Advanced ICT approaches Machine learning

Hands-on deep learning

prof. dr. Bojan Cestnik

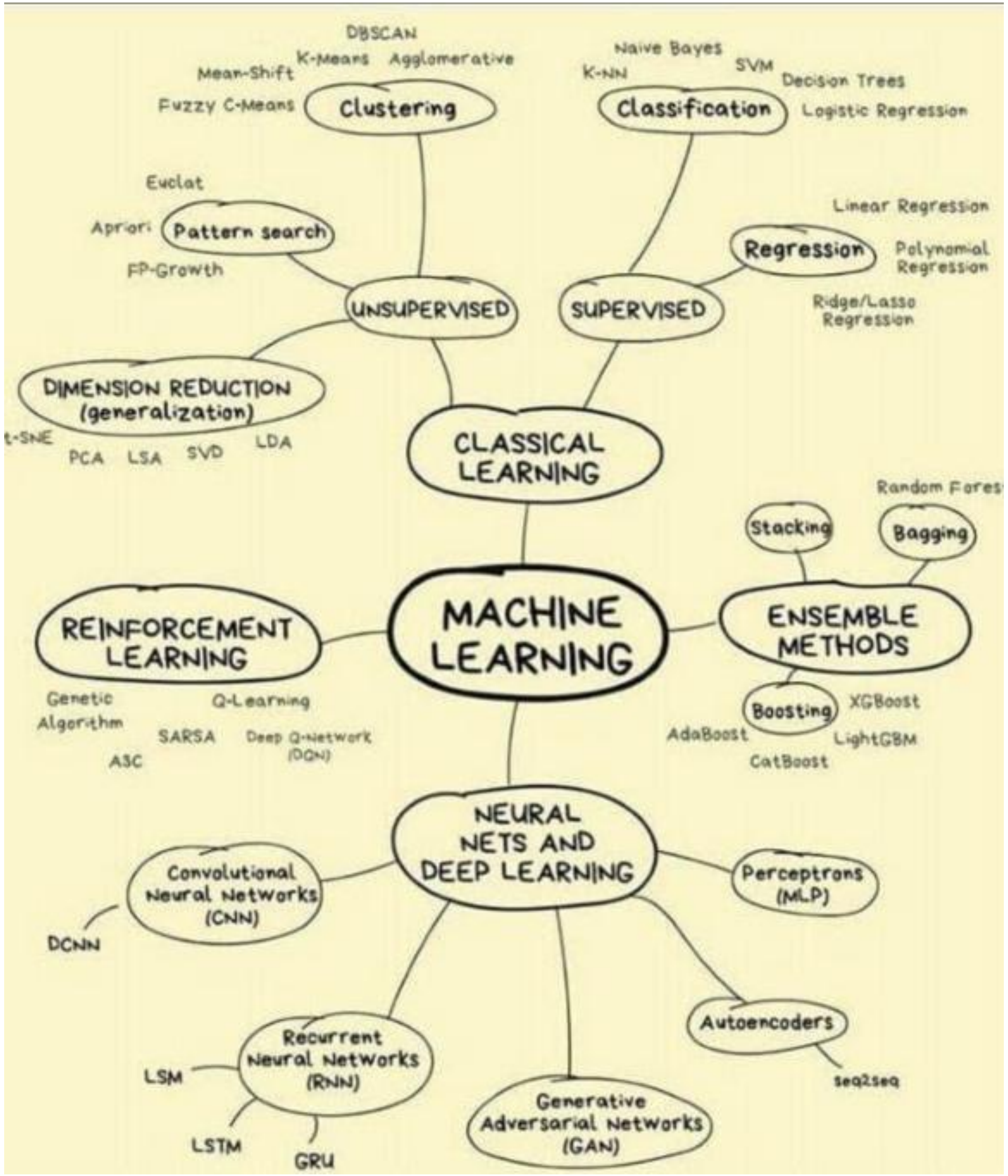
IPS, 19. 11. 2025

Slides from http://udlbook.com, by Simon J.D. Prince

Contents

Machine Learning (ML)

- Developing algorithms and models that enable machines to learn from data
- Key concepts:
  - **Supervised learning**: ML algorithms learn from labeled data (input-output pairs) to make predictions about new, unseen data
  - **Unsupervised learning**: ML algorithms learn from unlabeled data to identify patterns, relationships, or structures in the data
  - **Reinforcement learning**: ML algorithms learn by interacting with an environment and receiving feedback as rewards or penalties
  - **Transfer learning**: ML algorithms focus on storing knowledge gained while solving one problem and applying it to a different but related problem



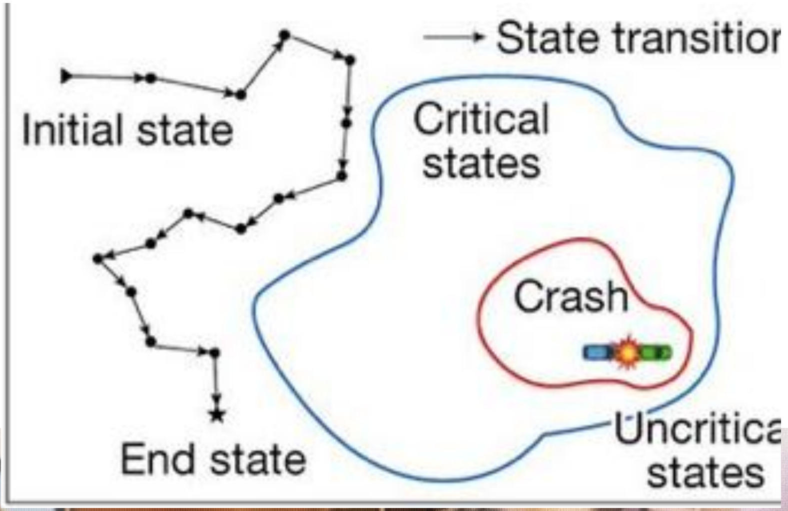
Transfer learning

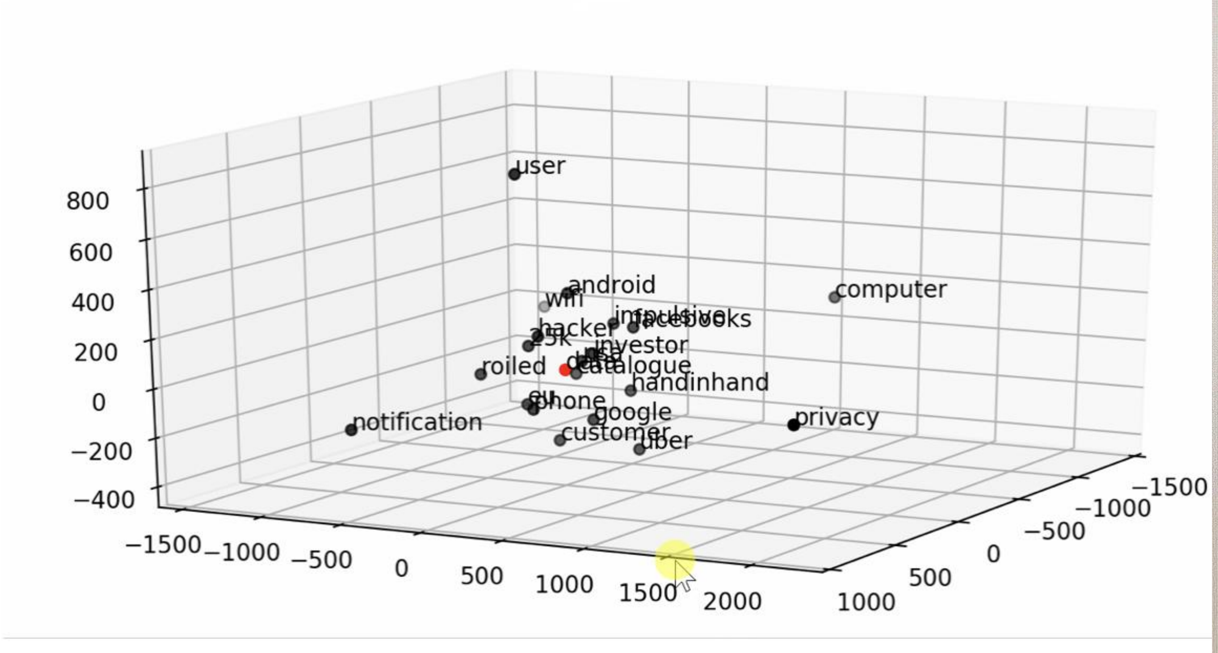
Data 1 → Trained model 1 → Predictions 1

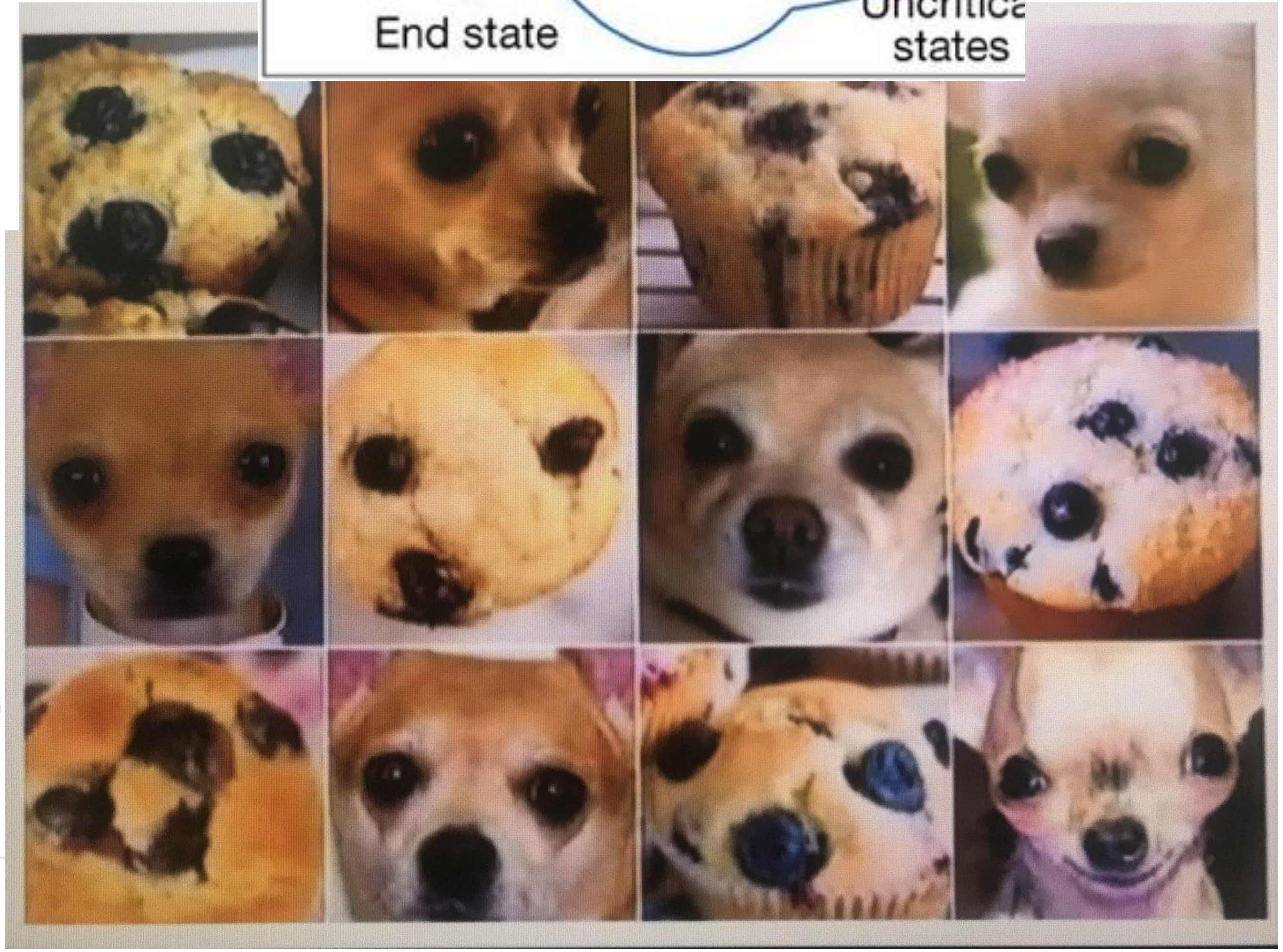
↓ Knowledge transfer

Data 2 → Trained model 2 → Predictions 2


NLP
natural
language
processing
learning
text
automatic
linguistics
interaction
public
processed
understanding
download
process
computer
retrieval
tag
typo
design
discourse
analysis
job
word
communicate
simulation
keywords
telecommunications
output
operating
typography
information
human
systems
input
layout
data
evolution
cloud
science
coreference
programming technology
summarization
networks
machine
automated
evaluation
statistical
artificial

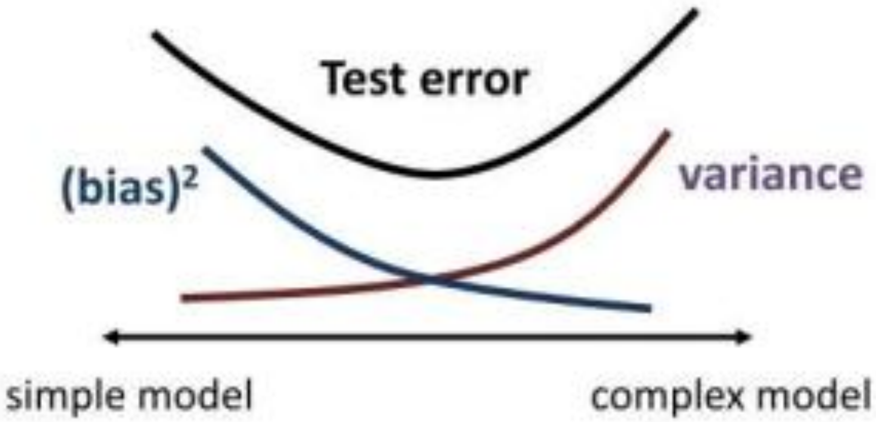

State transition
Initial state
Critical states
Crash
End state
Uncritical states

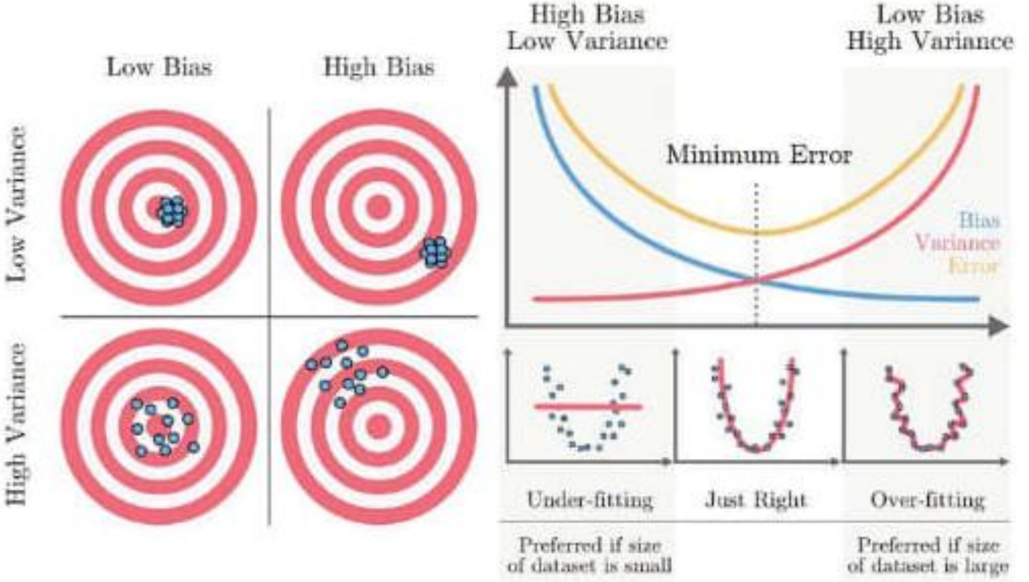

user
android
wifi
computer
hacker
investor
roiled
catalogue
handinhand
notification
phone
google
privacy
customer
uber
800
600
400
200
0
−200
−400
−1500
−1000
−500
0
500
1000
1500
2000
−1500
−1000
−500
0
500
1000



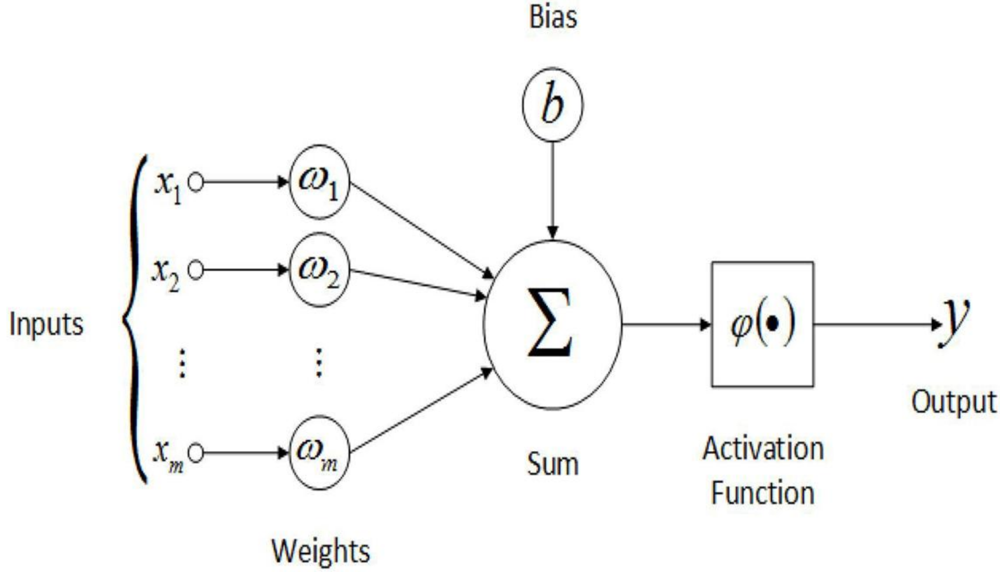
Machine Learning (ML)

- Learning models from training data (generalization)
- Measuring performance on unseen test data
- Model overfitting: too specialized on its training data
- Model underfitting: overly simplistic, inadequate capture of underlying patterns in the data
- Result: poor performance on unseen data





Neural Networks

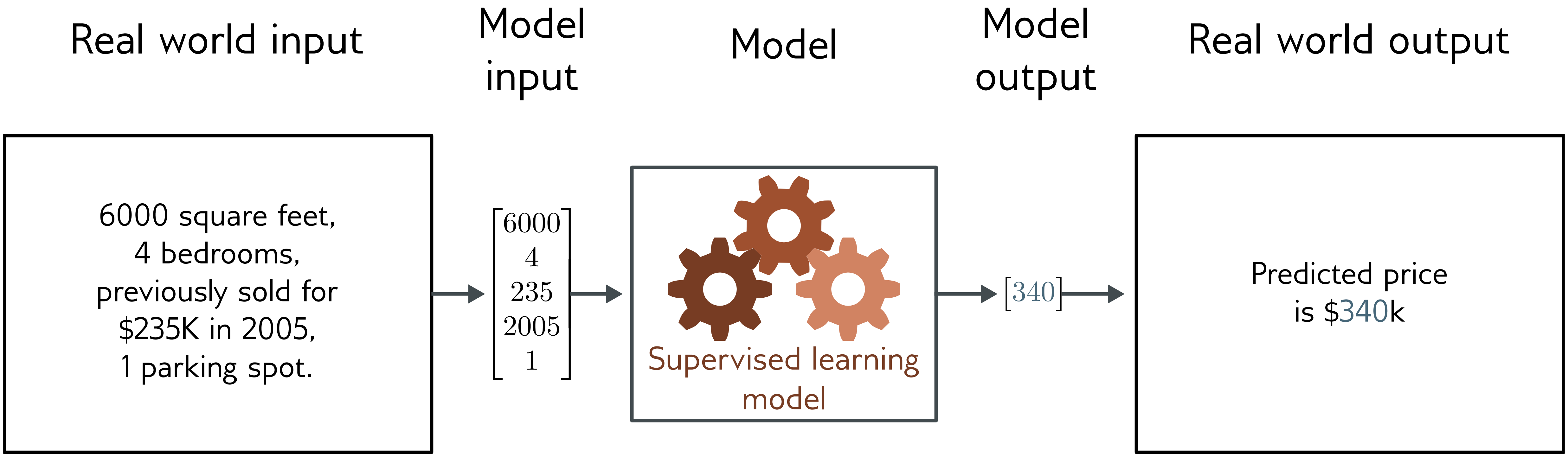


- **Inspired by the structure and function of biological neurons and neural systems**
- Designed to process, learn, and represent complex patterns in data
- Key concepts:
  - **Neurons**: the basic building blocks of neural networks, which receive input signals, process them, and produce output signals
  - **Activation functions**: nonlinear functions applied to the weighted sum of a neuron's inputs to determine its output signal
  - **Backpropagation**: an algorithm used to train neural networks by minimizing the error between predicted outputs and actual outputs through gradient descent and weight adjustments

1. Linear regression

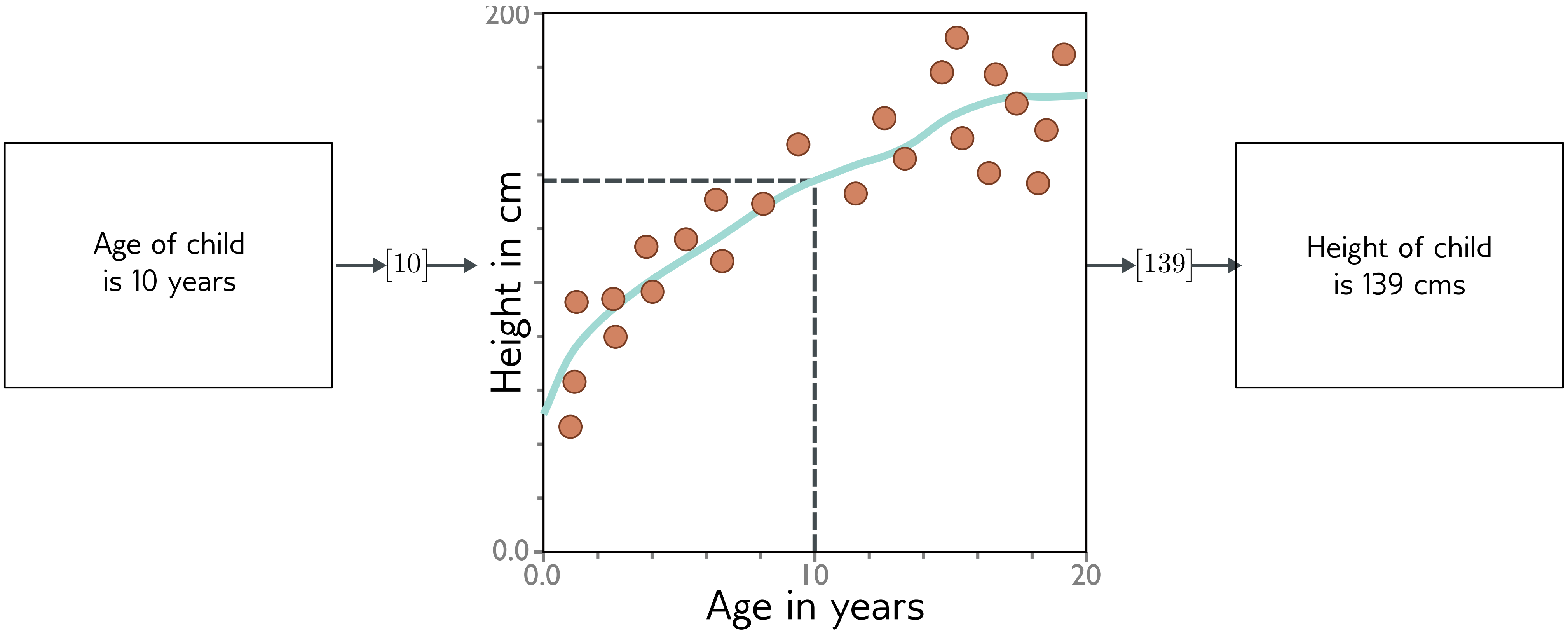
- Data points (x_1, y_1), (x_2, y_2), (x_3, y_3), … (x_{n-1}, y_{n-1}), (x_n, y_n)
- $y = k * x + n$
- Loss function
- Estimation of k and n from data

Regression



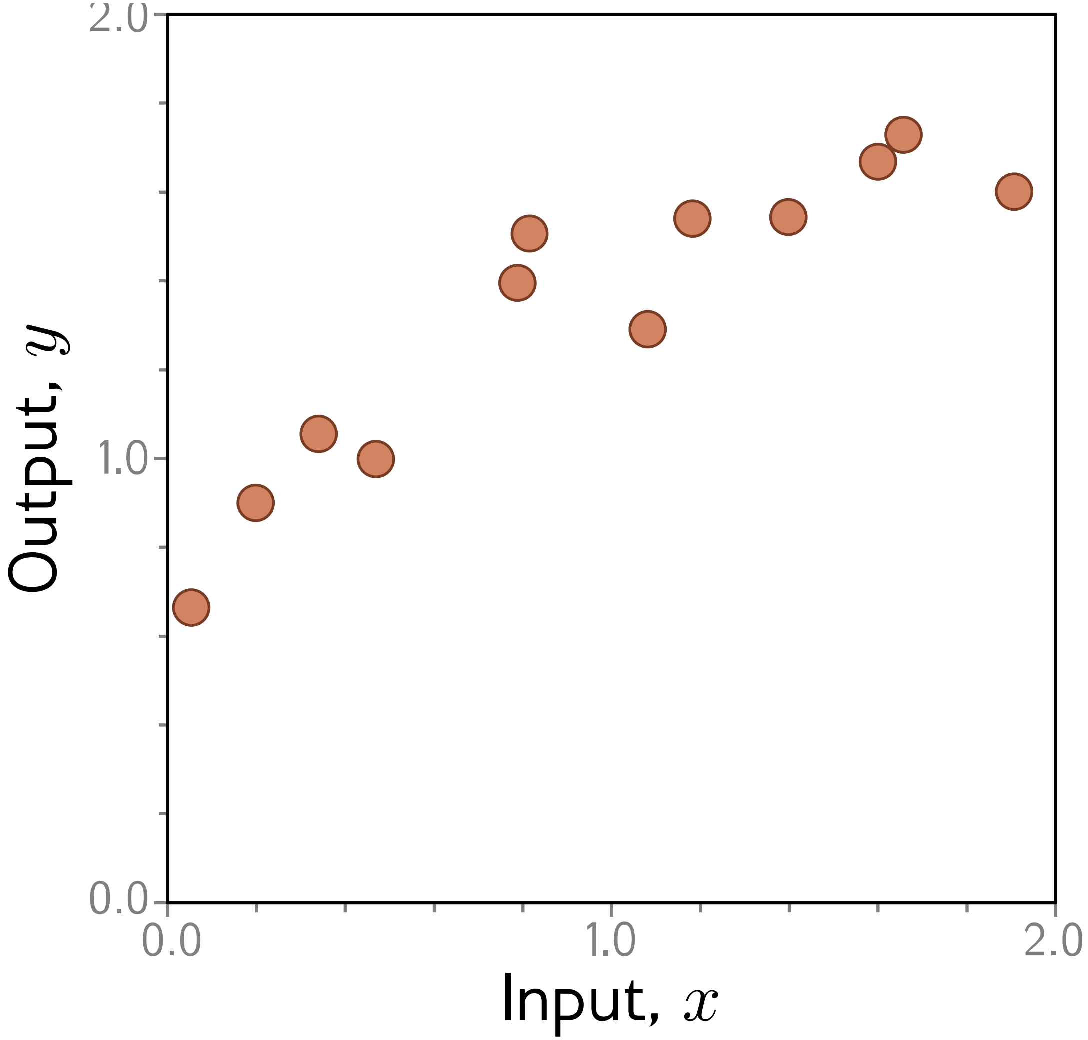
- Univariate regression problem (one output, real value)
- Fully connected network

What is a supervised learning model?

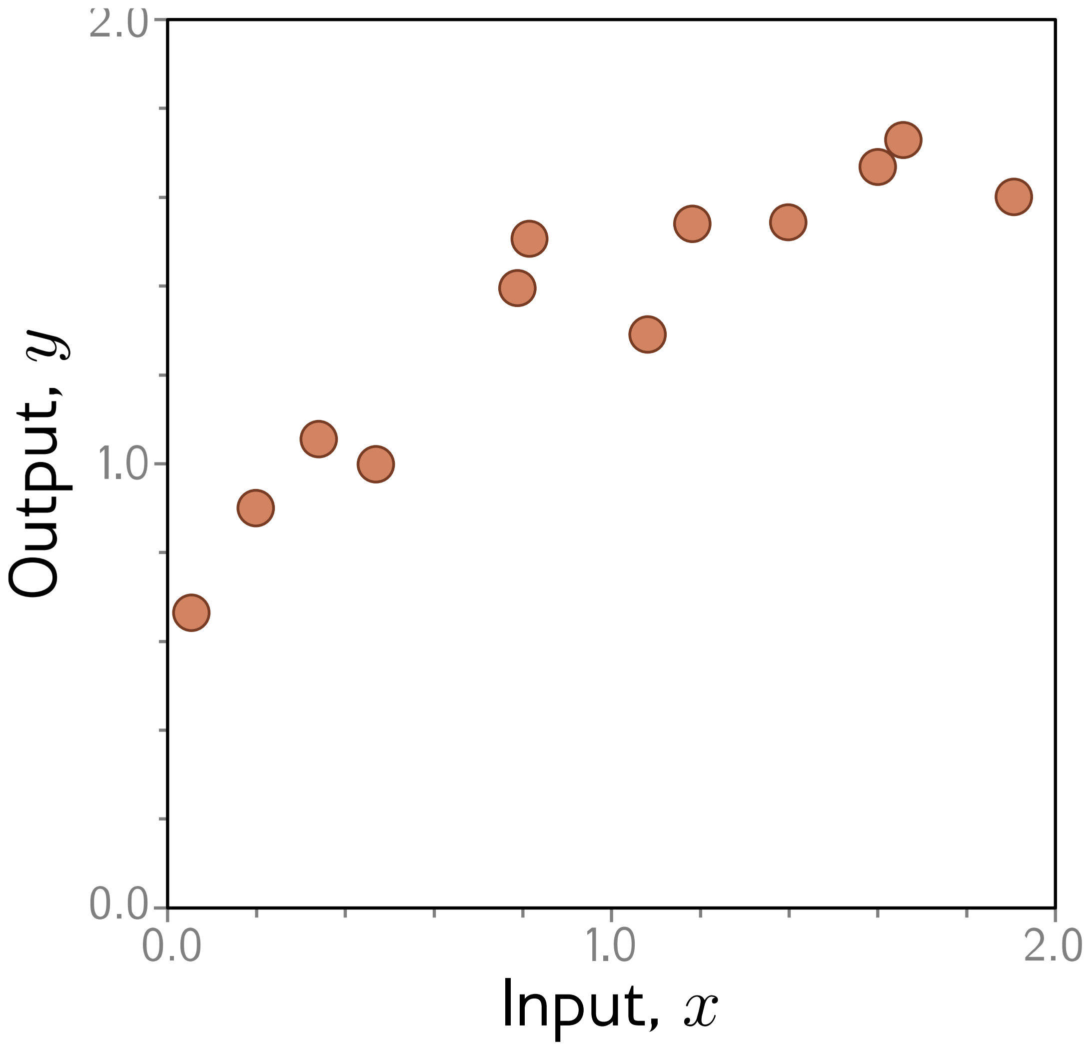


- An equation relating input (age) to output (height)
- Search through family of possible equations to find one that fits training data well

Example: 1D Linear regression training data



Example: 1D Linear regression training data



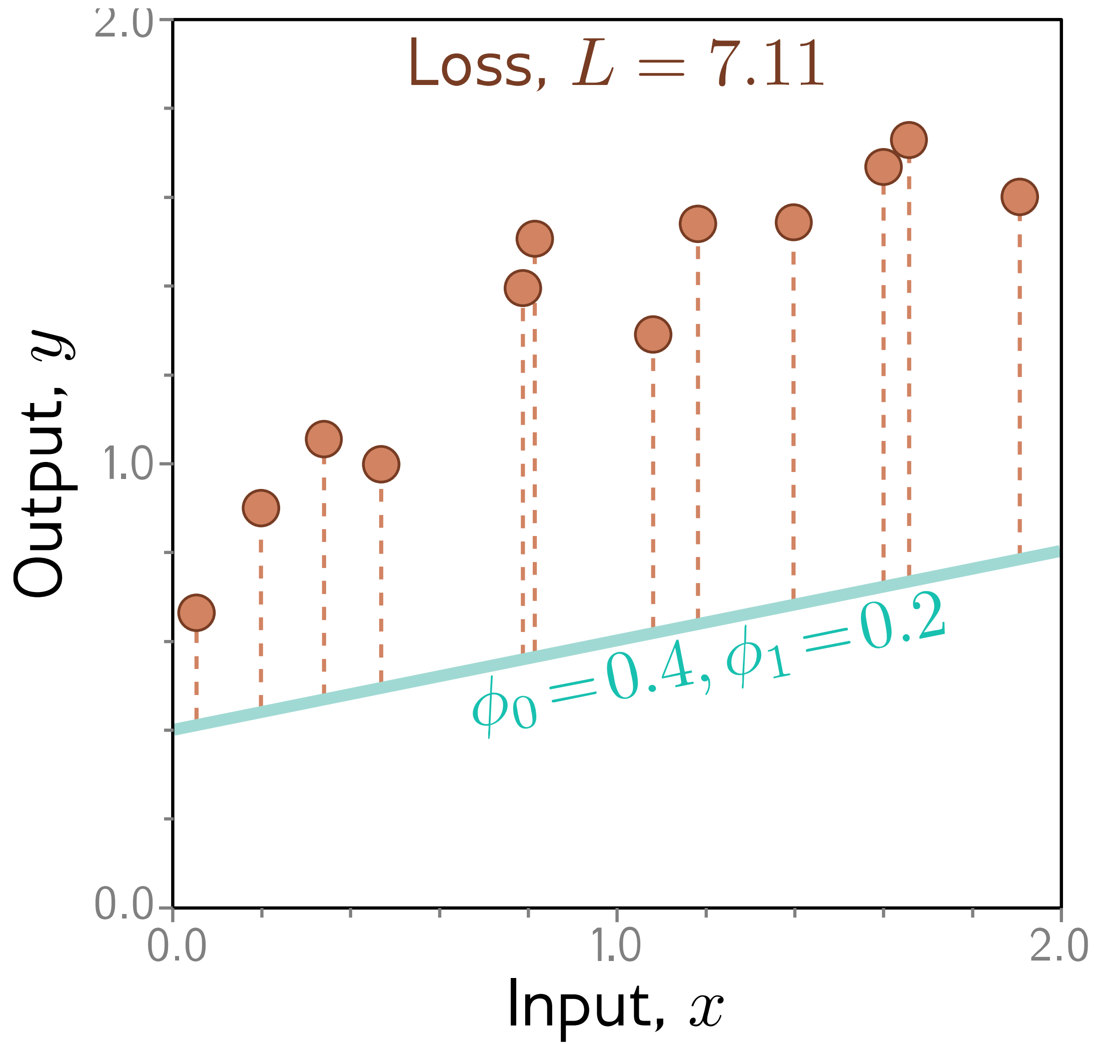
Loss function:

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

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

"Least squares loss function"

Example: 1D Linear regression loss function



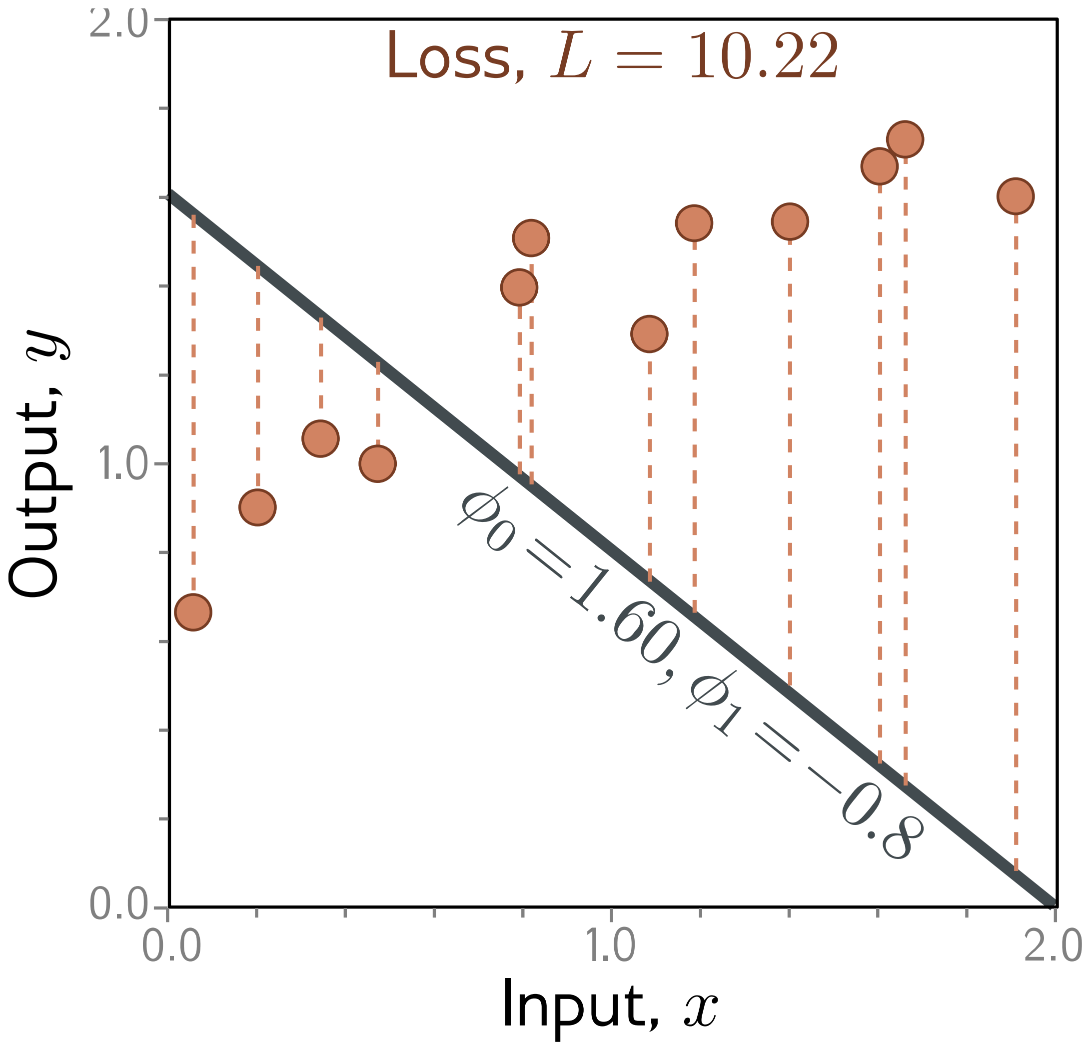
Loss function:

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

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

"Least squares loss function"

Example: 1D Linear regression loss function



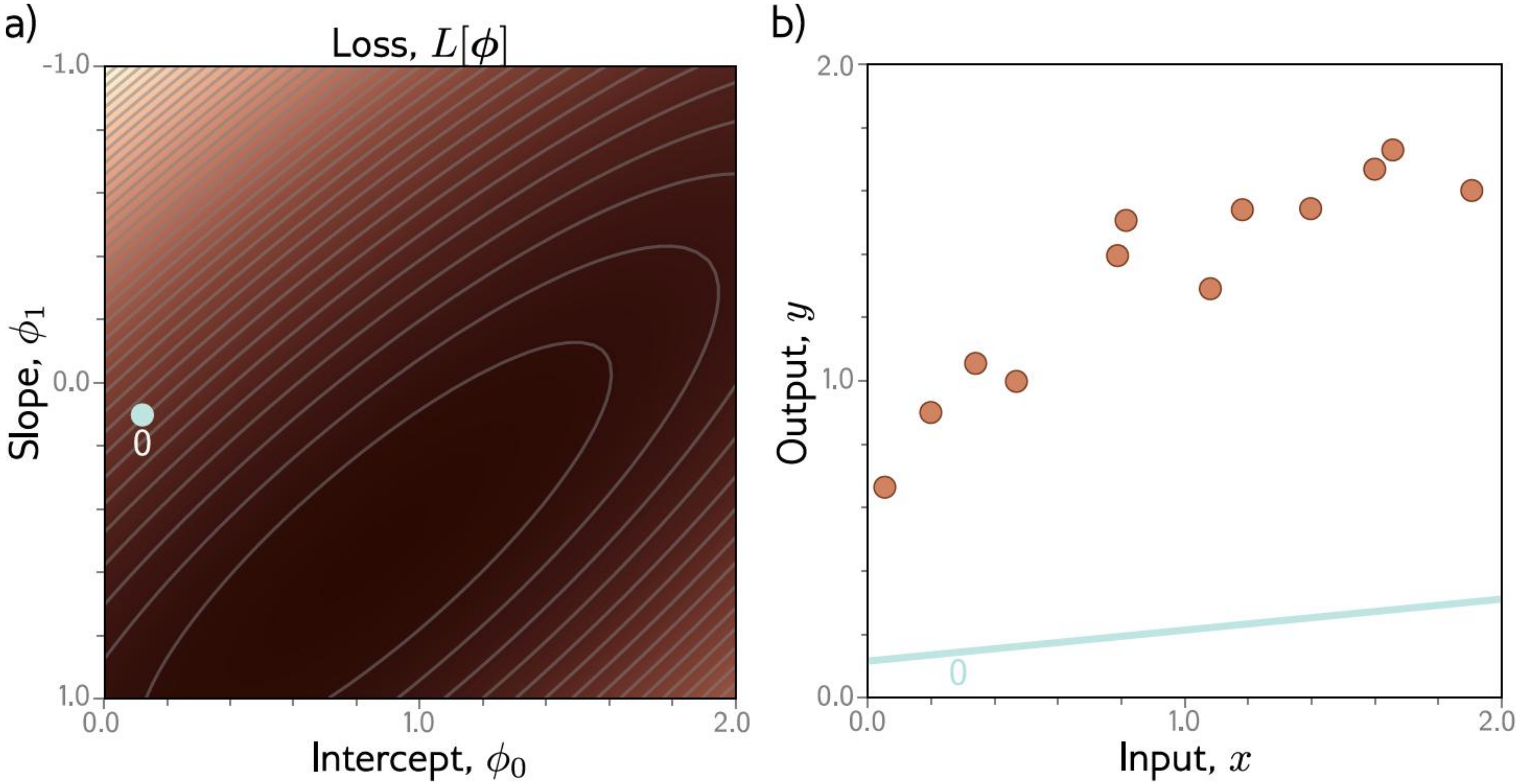
Loss function:

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

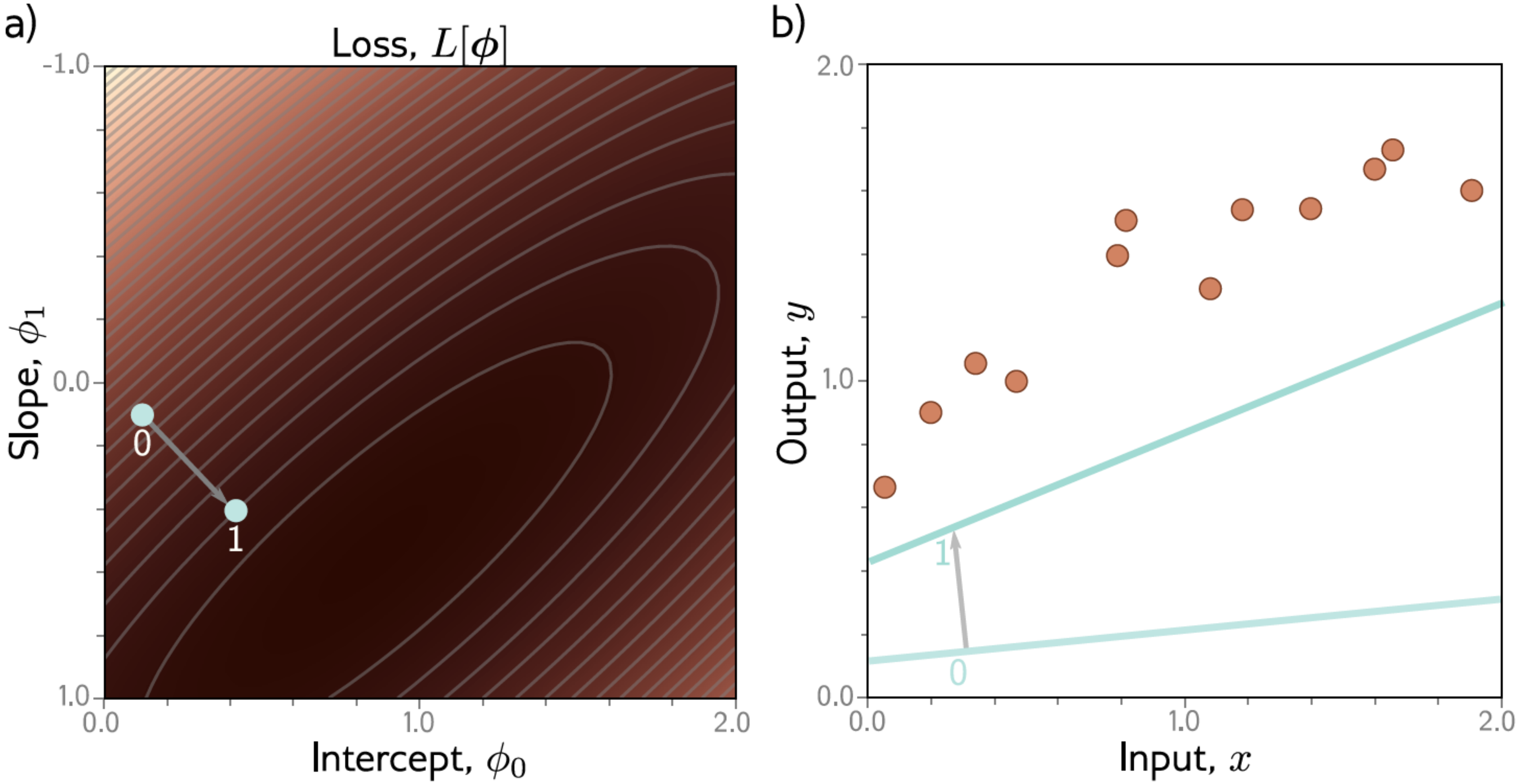
$$= \sum_{i=1}^{I} (\phi_0 + \phi_1 x_i - y_i)^2$$

"Least squares loss function"

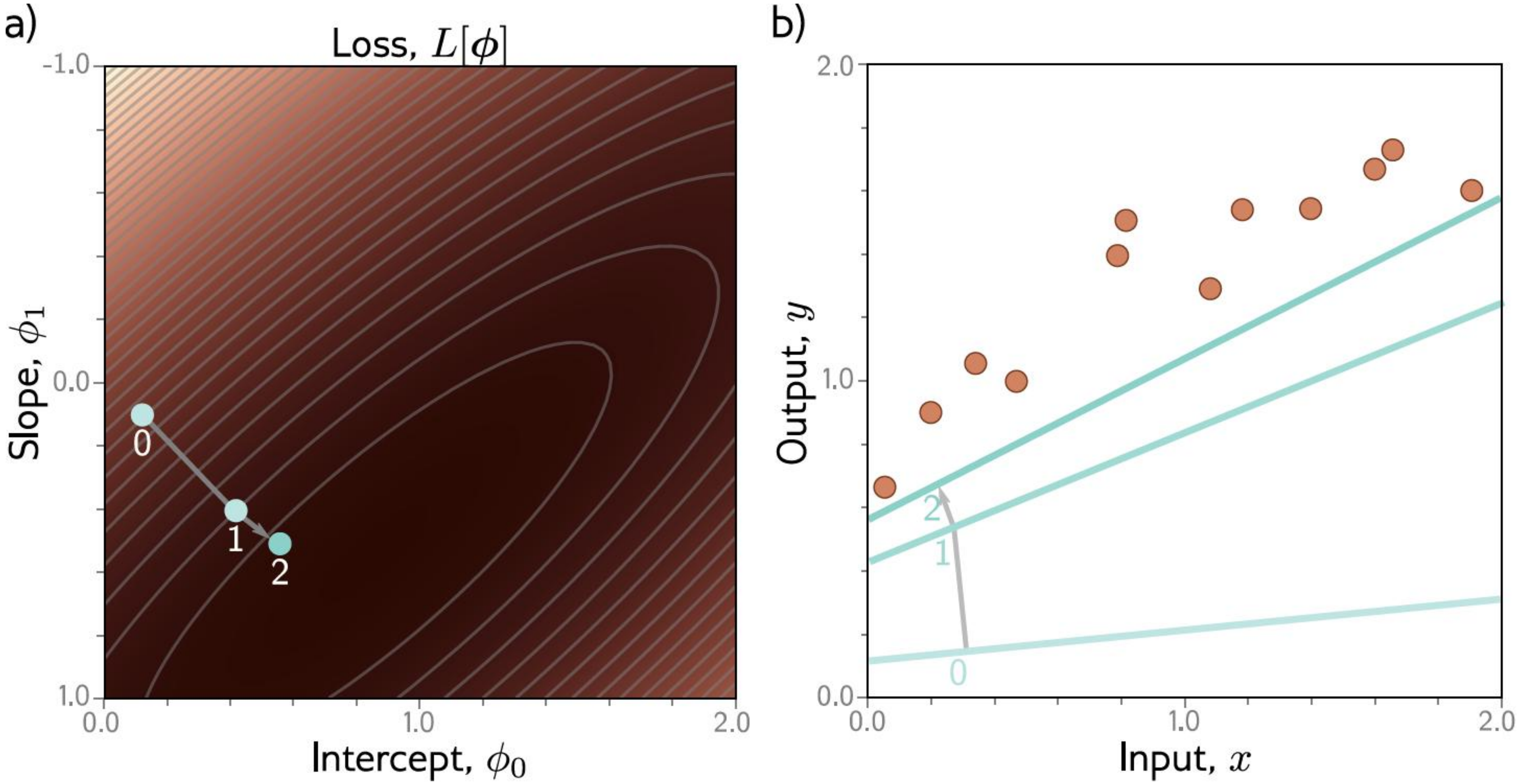
Example: 1D Linear regression training



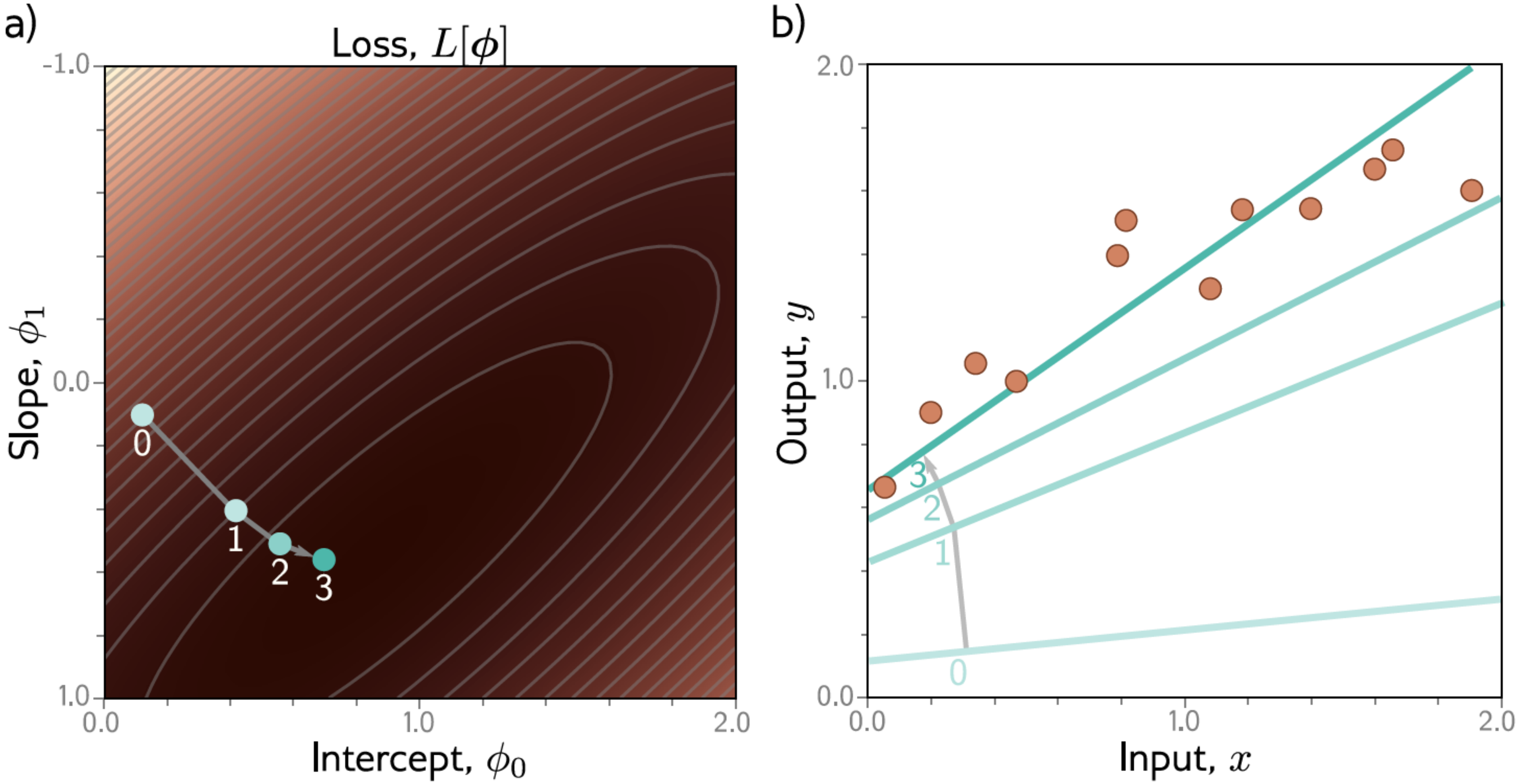
Example: 1D Linear regression training



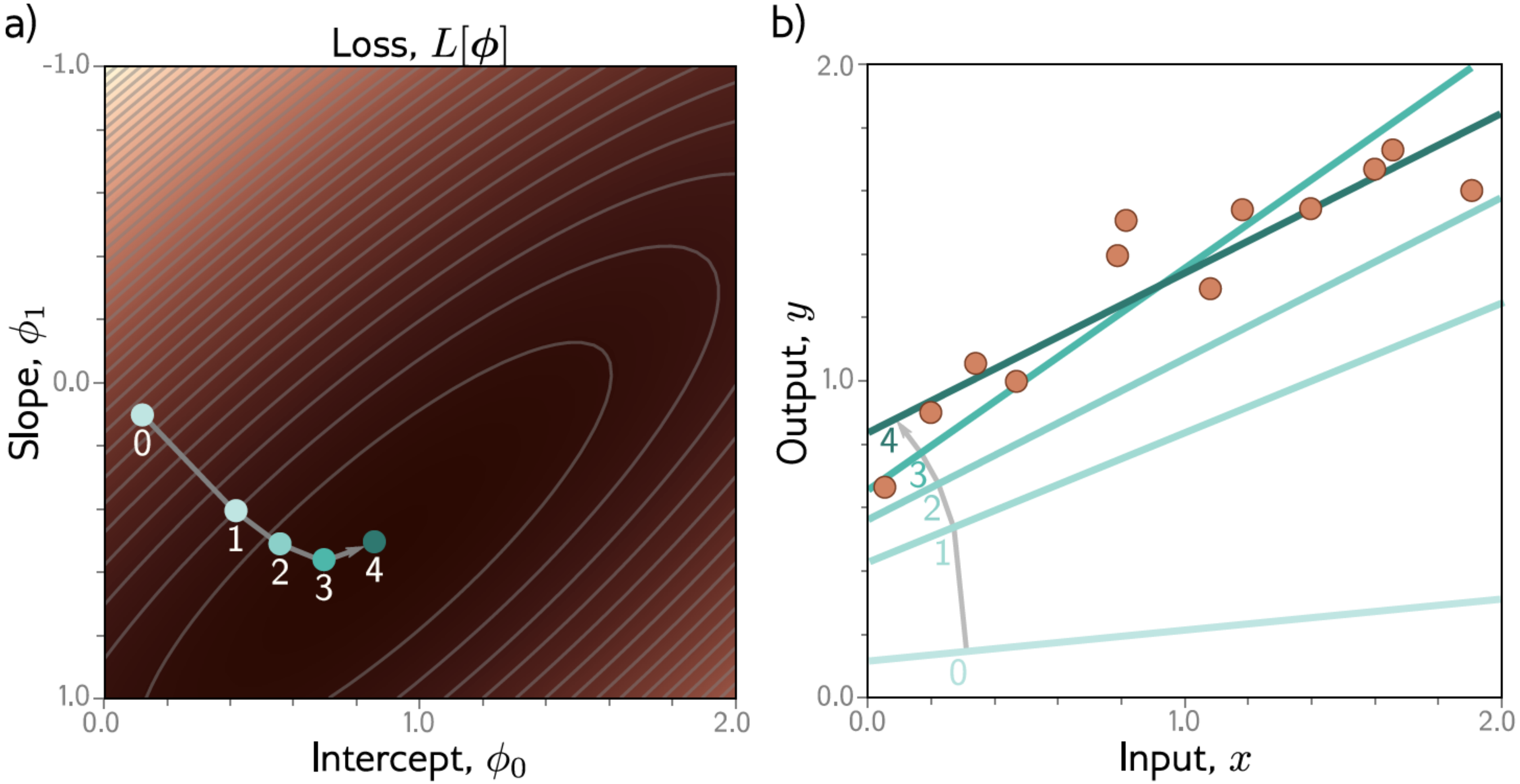
Example: 1D Linear regression training



Example: 1D Linear regression training



Example: 1D Linear regression training



This technique is known as gradient descent

Possible objections

- But you can fit the line model in closed form!
  - Yes – but we won't be able to do this for more complex models
- But we could exhaustively try every slope and intercept combo!
  - Yes – but we won't be able to do this when there are a million parameters

Example in python notebook

- Notebook 2.1 Supervised Learning

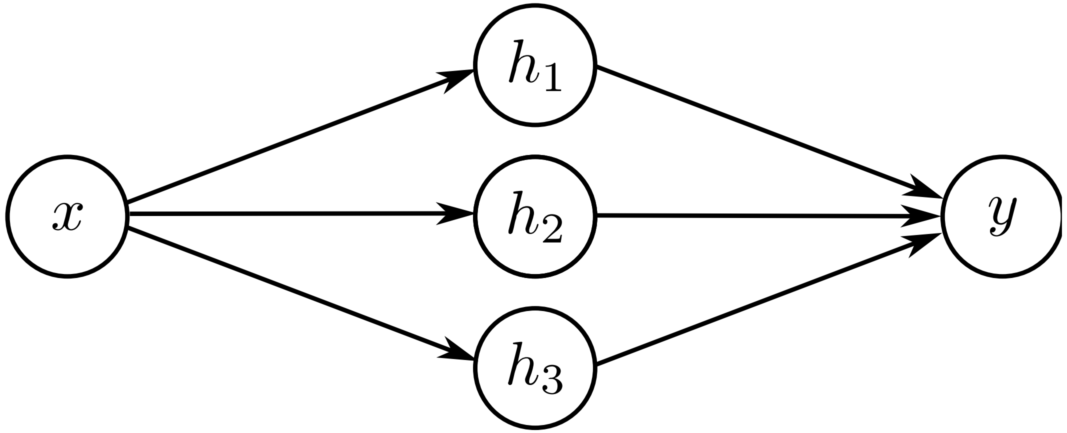
2. Approximation with shallow neural network

- Data points $(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots (x_{n-1}, y_{n-1}), (x_n, y_n)$
- Loss function
- Estimation of parameters (from training data)

Depicting neural networks

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

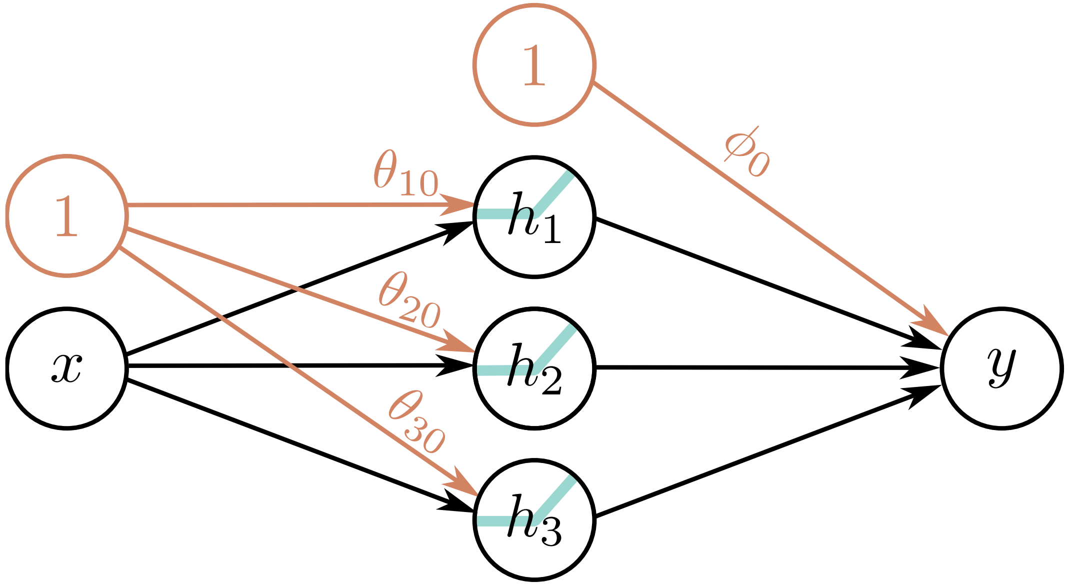
$$y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$



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

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

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

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

Rectified Linear Unit

(particular kind of activation function)

Example shallow network

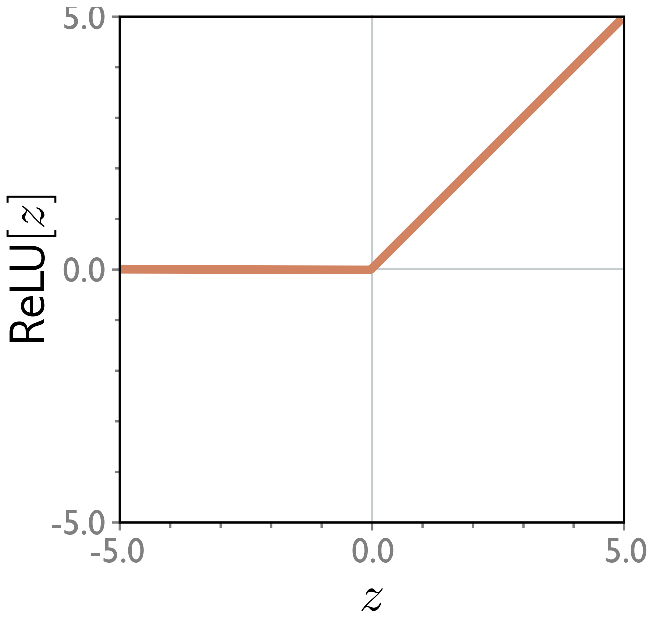
Activation function

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

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

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

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

Hidden units

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

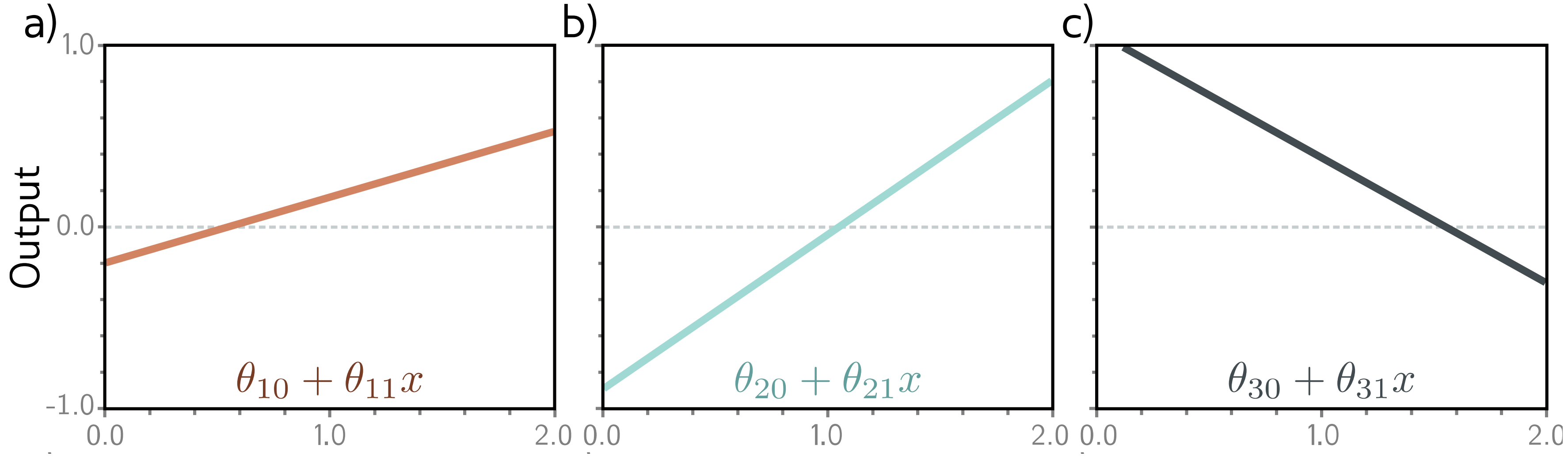
Break down into two parts:

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

where:

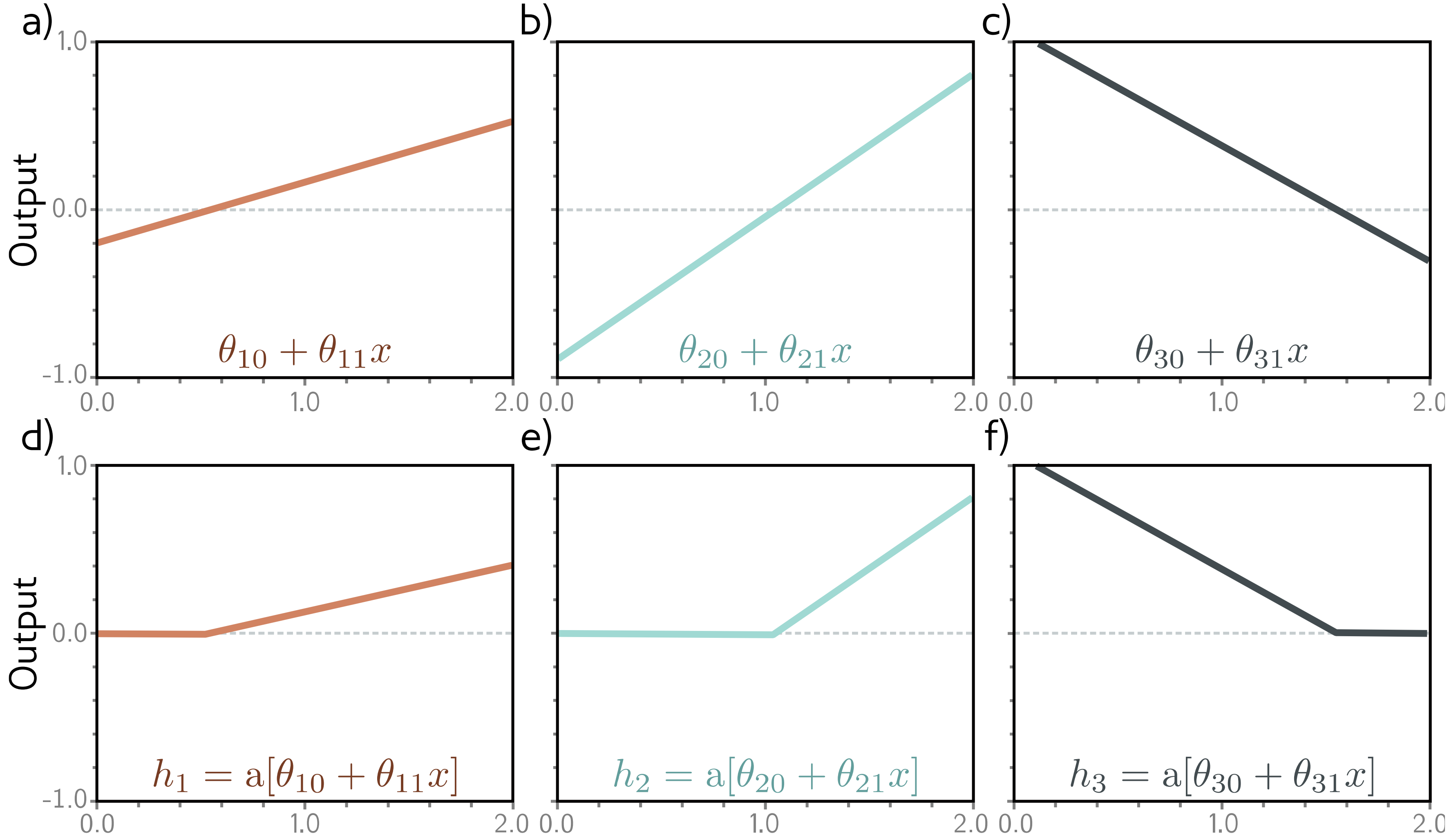
$$\text{Hidden units} \left\{ \begin{aligned} h_1 &= \mathrm{a}[\theta_{10} + \theta_{11}x] \\ h_2 &= \mathrm{a}[\theta_{20} + \theta_{21}x] \\ h_3 &= \mathrm{a}[\theta_{30} + \theta_{31}x] \end{aligned} \right.$$

1. compute three linear functions

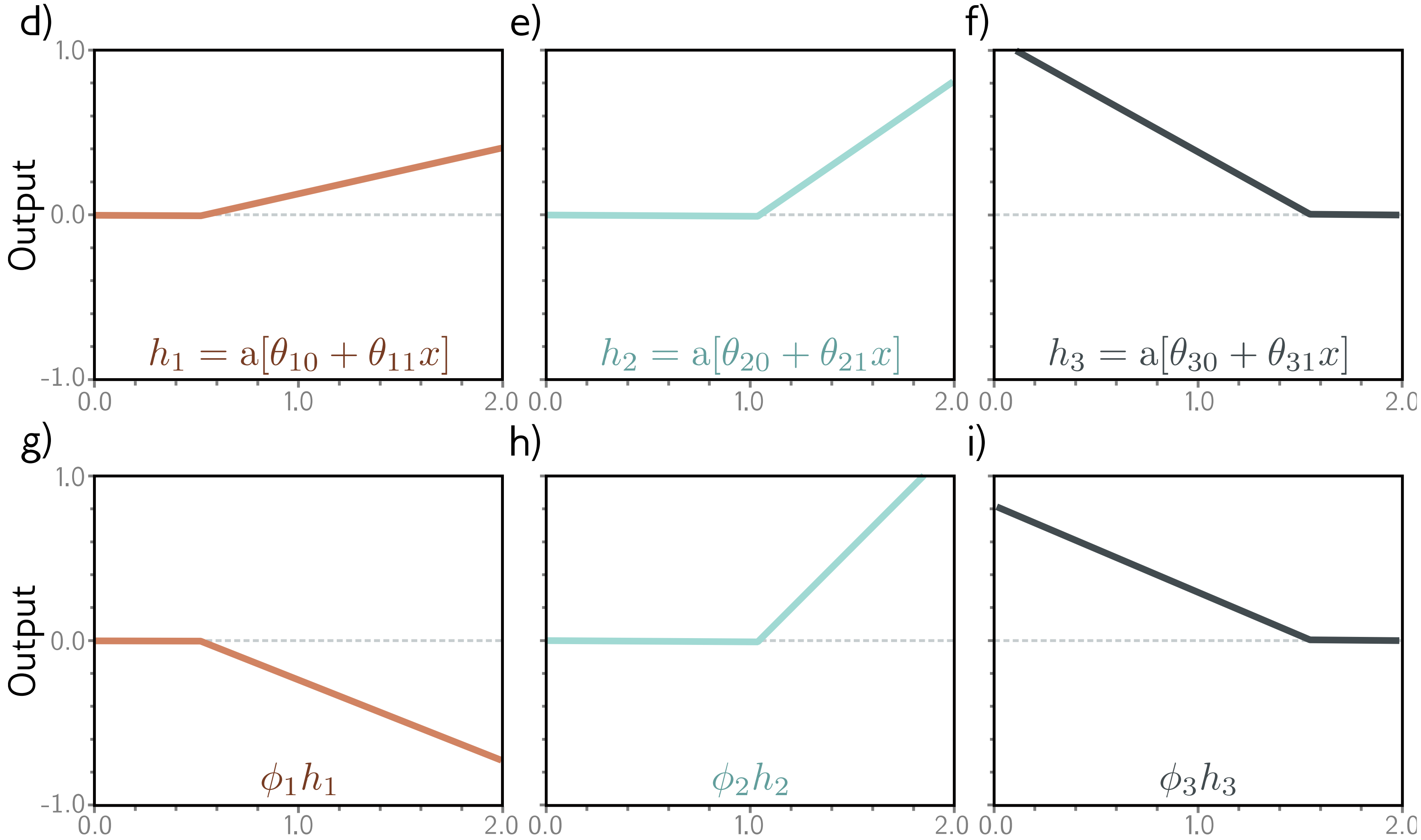


2. Pass through ReLU functions (creates hidden units)

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

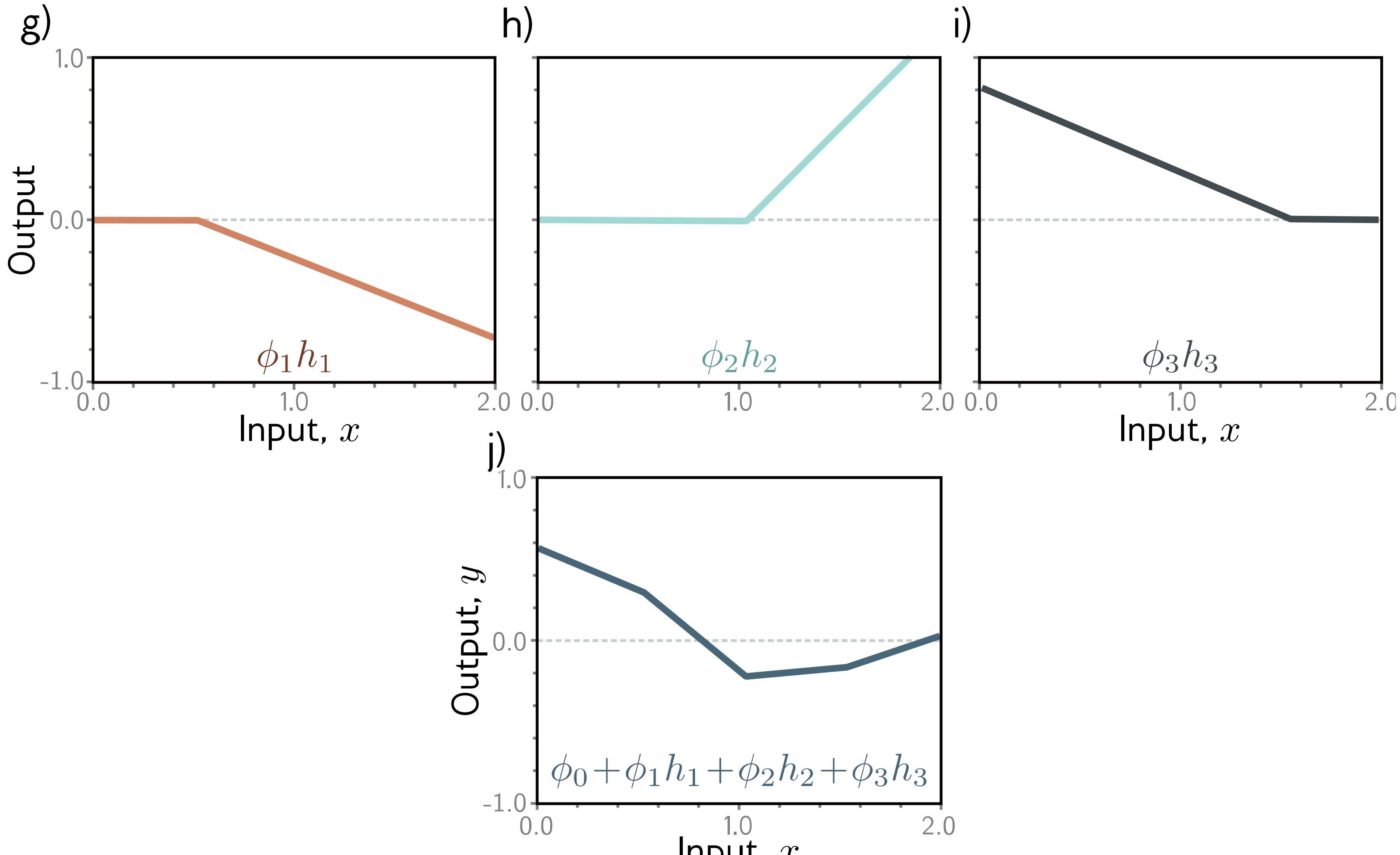


2. Weight the hidden units

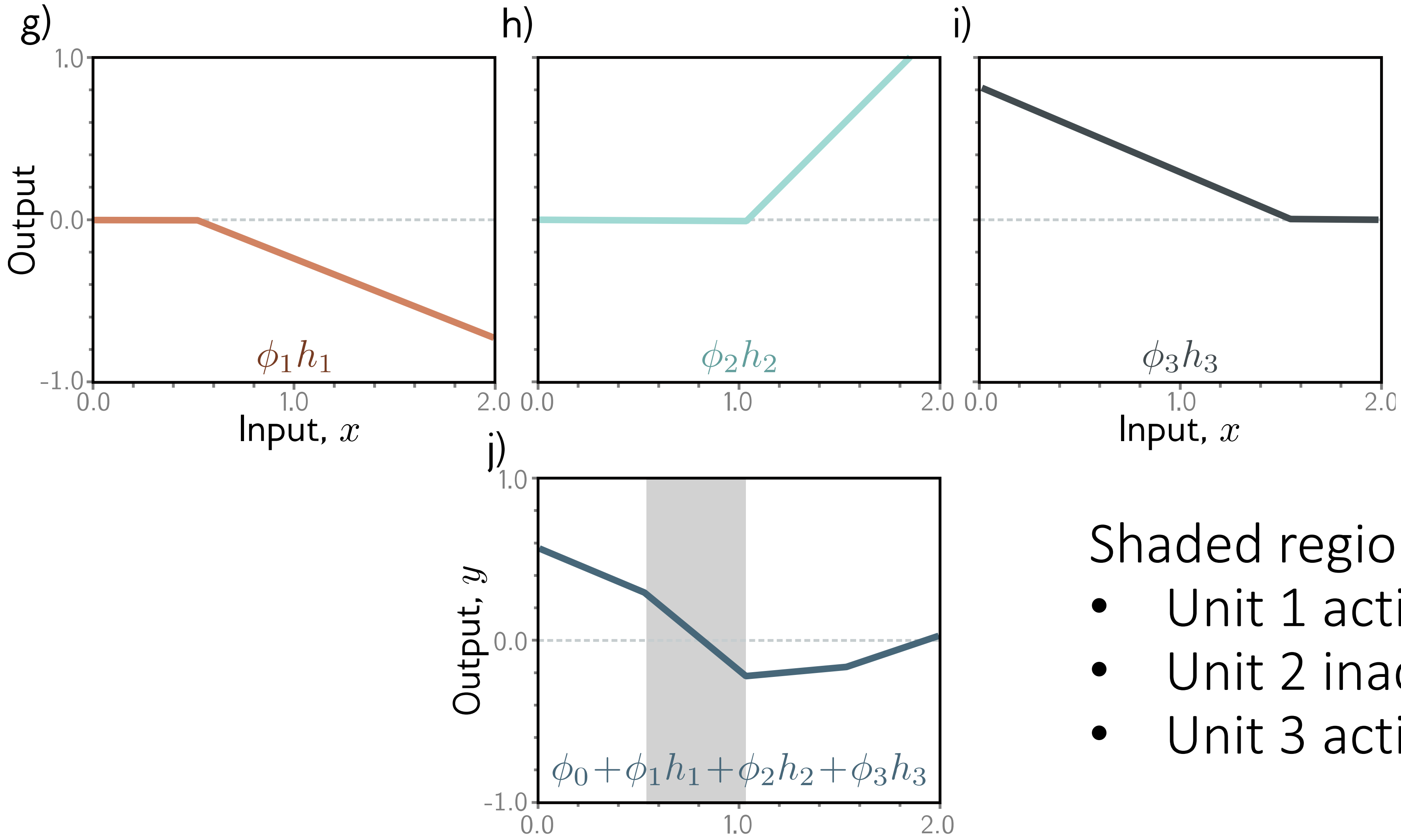


4. Sum the weighted hidden units to create output

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



Activation pattern = which hidden units are activated



Shaded region:

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

Example in python notebook

- Notebook 3.1 Shallow neural networks I

3. Approximation with deep neural network

- Data points $(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots (x_{n-1}, y_{n-1}), (x_n, y_n)$
- Loss function
- Estimation of parameters

Deep neural networks

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

Composing two networks.

Network 1:

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

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

Network 2:

$$h'_1 = \mathrm{a}[\theta'_{10} + \theta'_{11}y]$$
$$h'_2 = \mathrm{a}[\theta'_{20} + \theta'_{21}y]$$
$$h'_3 = \mathrm{a}[\theta'_{30} + \theta'_{31}y]$$

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

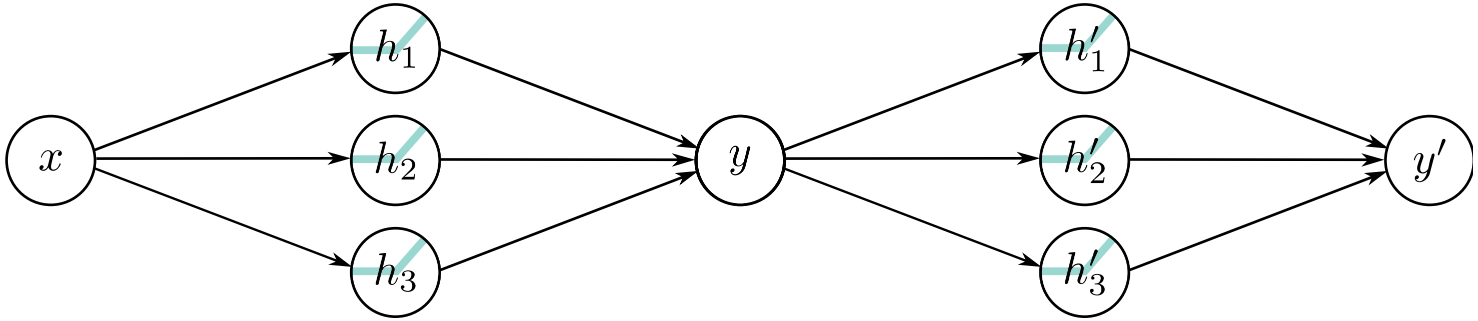
Composing two networks.

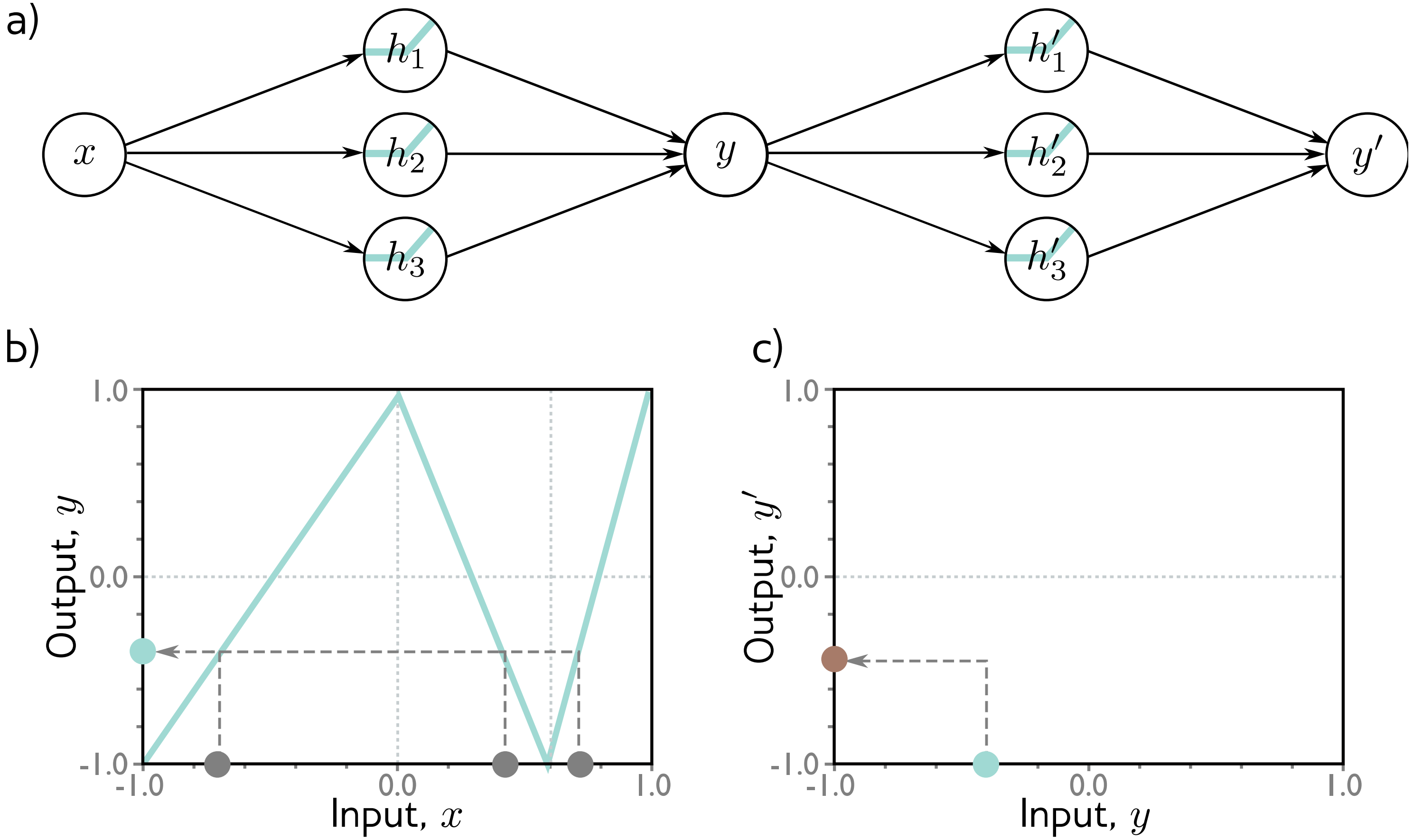
Network 1:

$$h_1 = \mathrm{a}[\theta_{10} + \theta_{11}x]$$
$$h_2 = \mathrm{a}[\theta_{20} + \theta_{21}x] \qquad y = \phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$$
$$h_3 = \mathrm{a}[\theta_{30} + \theta_{31}x]$$

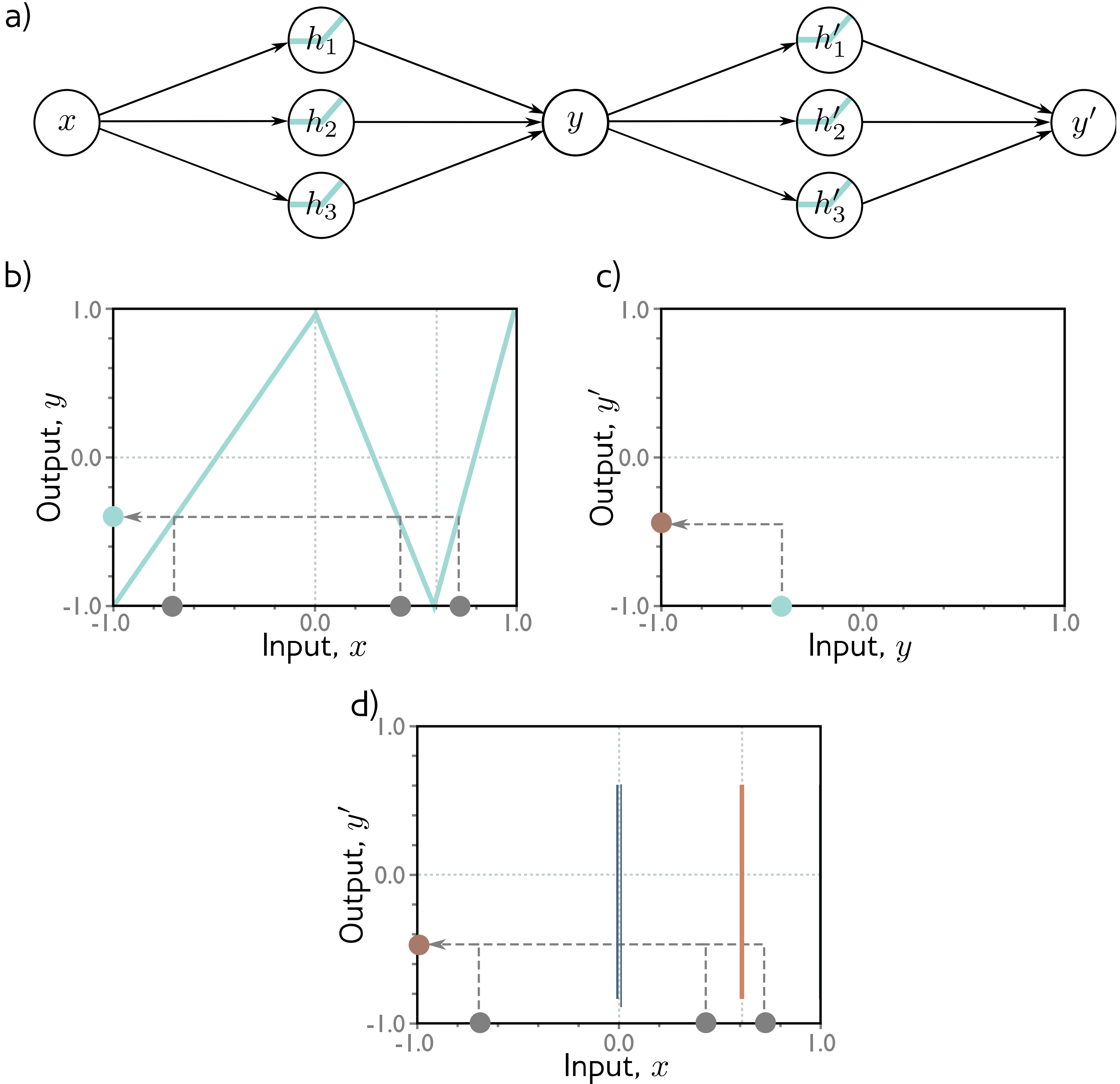
Network 2:

$$h'_1 = \mathrm{a}[\theta'_{10} + \theta'_{11}y]$$
$$h'_2 = \mathrm{a}[\theta'_{20} + \theta'_{21}y] \qquad y' = \phi'_0 + \phi'_1 h'_1 + \phi'_2 h'_2 + \phi'_3 h'_3$$
$$h'_3 = \mathrm{a}[\theta'_{30} + \theta'_{31}y]$$

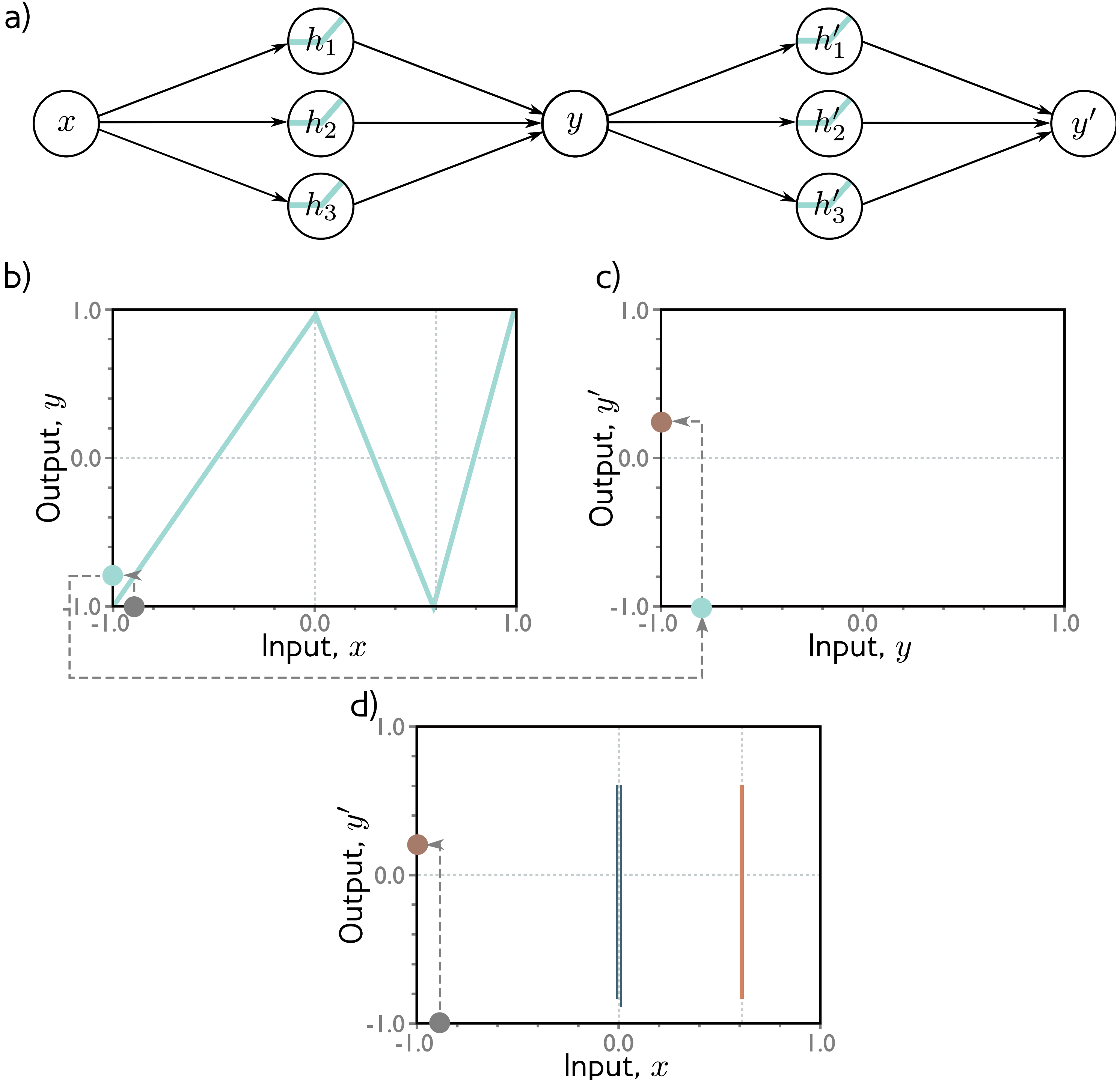




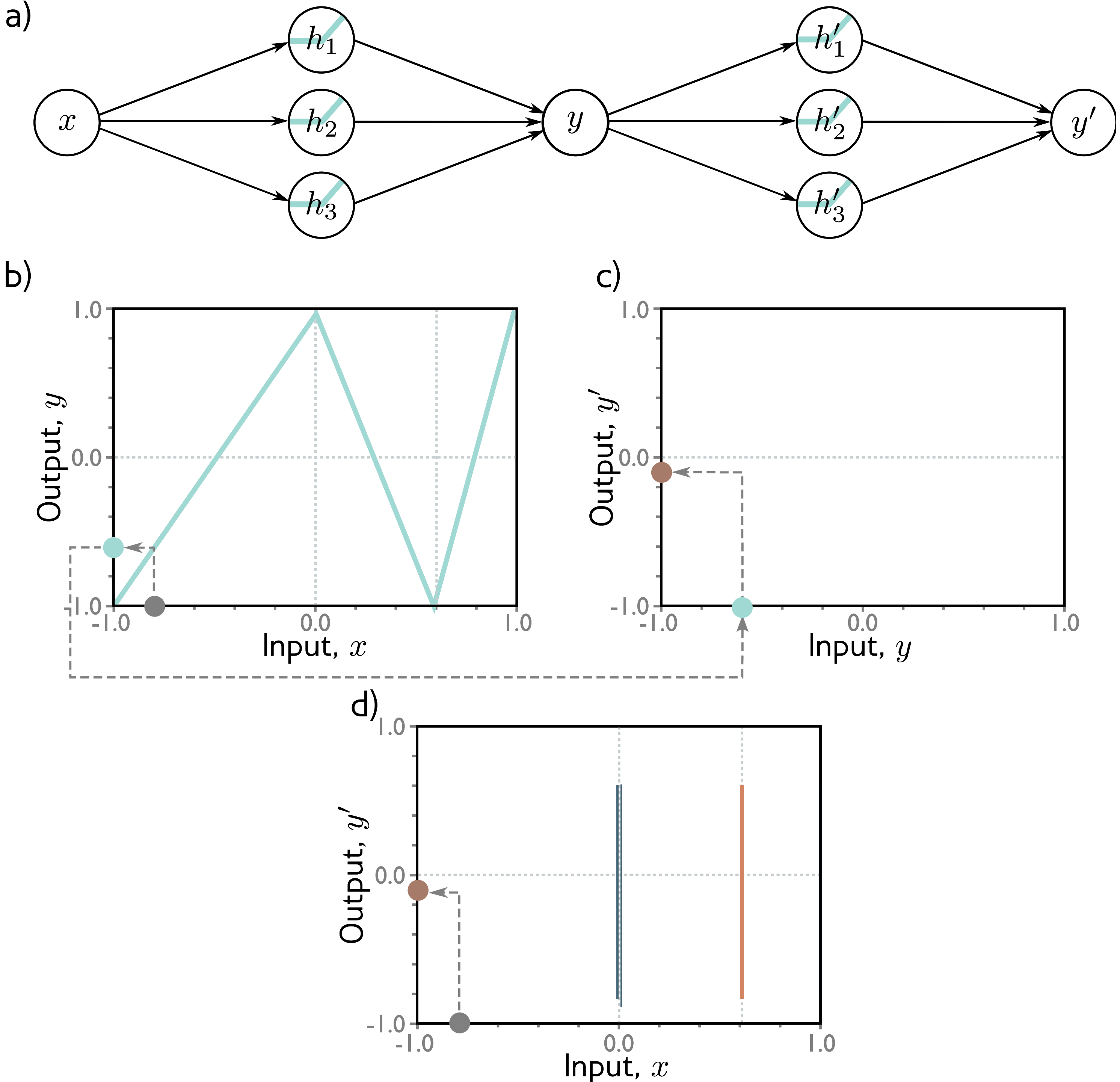
a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
1.0
0.0
-1.0
-1.0
0.0
1.0
Output, y
Input, x
c)
1.0
0.0
-1.0
-1.0
0.0
1.0
Output, y'
Input, y



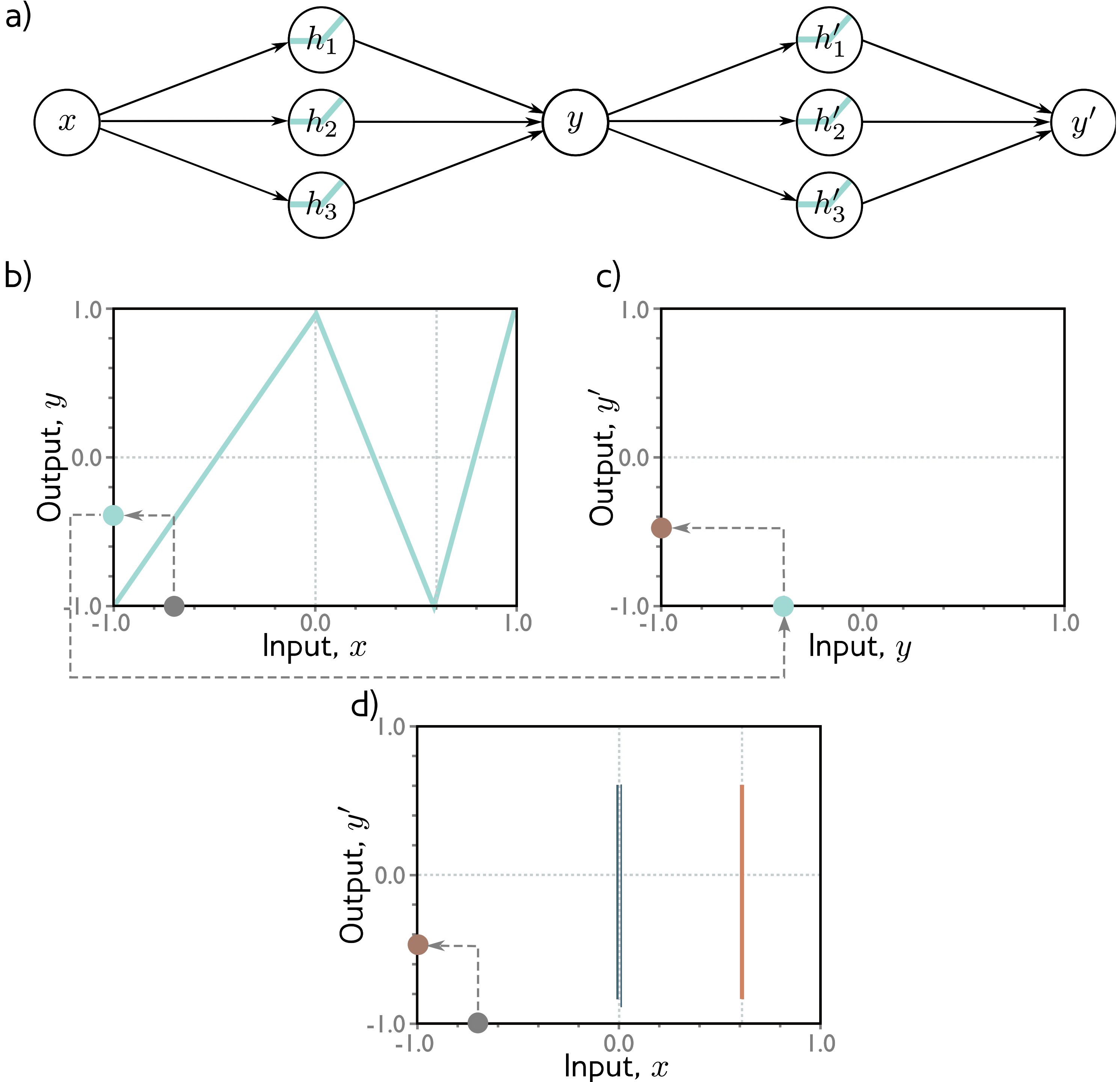
a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
Output, y
Input, x
c)
Output, y'
Input, y
d)
Output, y'
Input, x
1.0
0.0
-1.0



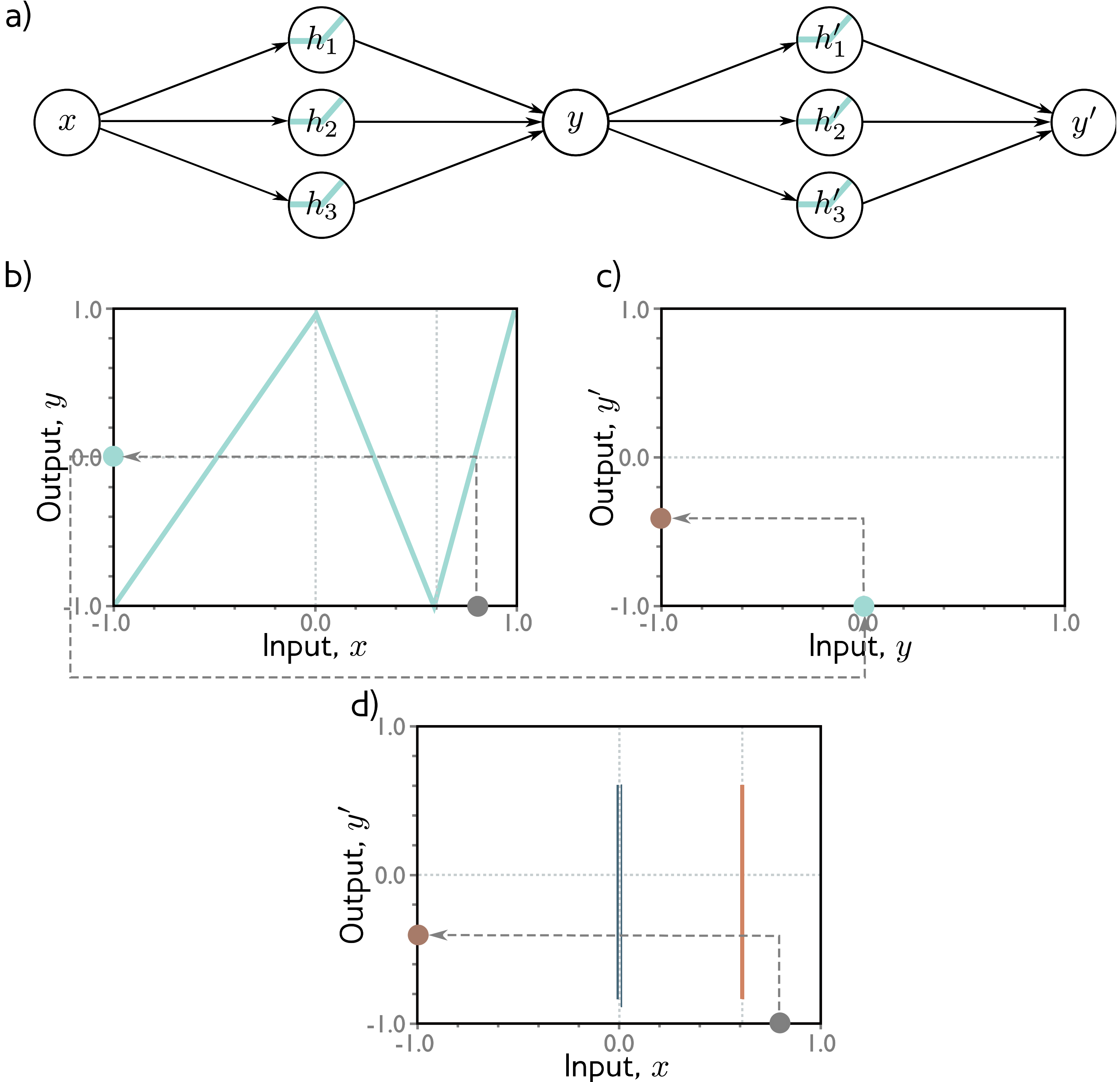
a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
Output, y
Input, x
1.0
0.0
-1.0
c)
Output, y'
Input, y
d)
Output, y'
Input, x



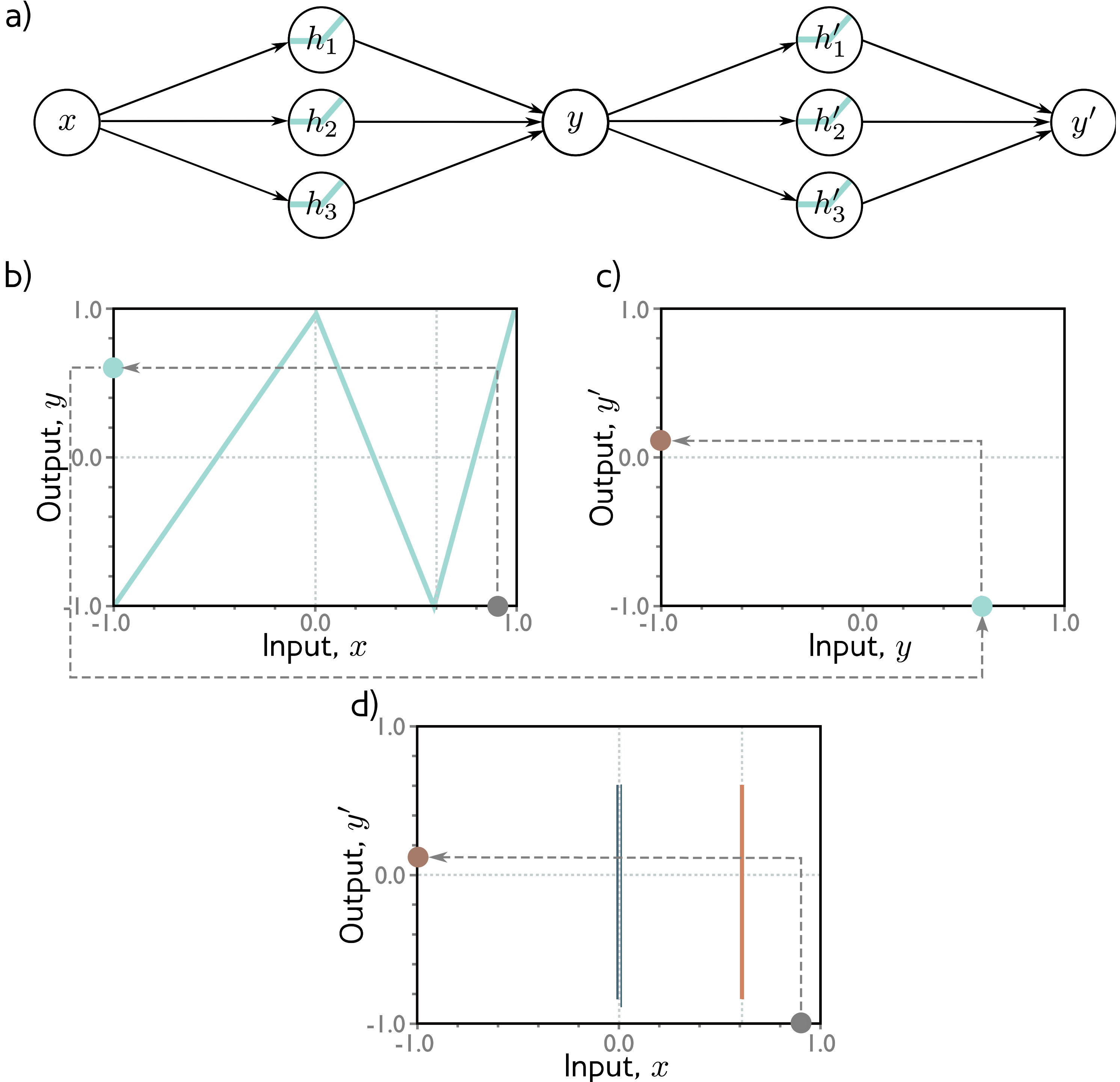
a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
1.0
0.0
-1.0
-1.0
0.0
1.0
Output, y
Input, x
c)
1.0
0.0
-1.0
-1.0
0.0
1.0
Output, y'
Input, y
d)
1.0
0.0
-1.0
-1.0
0.0
1.0
Output, y'
Input, x



a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
Output, y
Input, x
c)
Output, y'
Input, y
d)
Output, y'
Input, x
1.0
0.0
-1.0

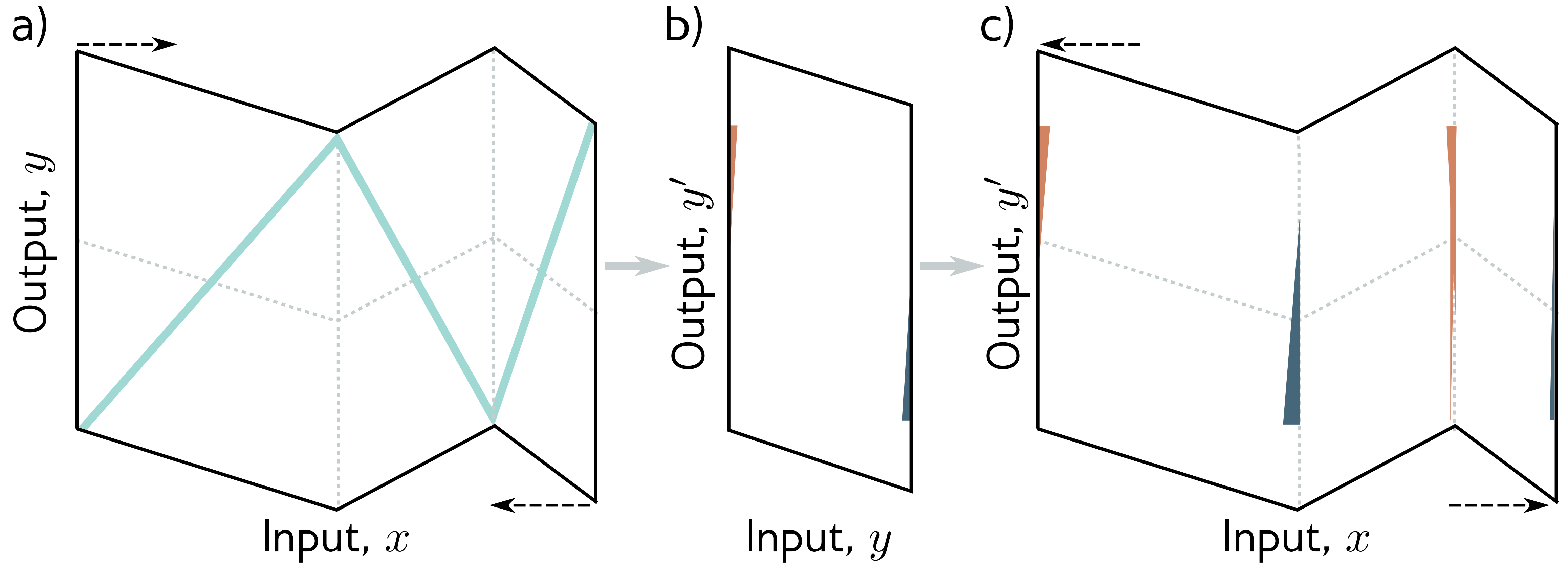


a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
Output, y
Input, x
1.0
0.0
-1.0
-1.0
0.0
1.0
c)
Output, y'
Input, y
1.0
0.0
-1.0
-1.0
0.0
1.0
d)
Output, y'
Input, x
1.0
0.0
-1.0
-1.0
0.0
1.0

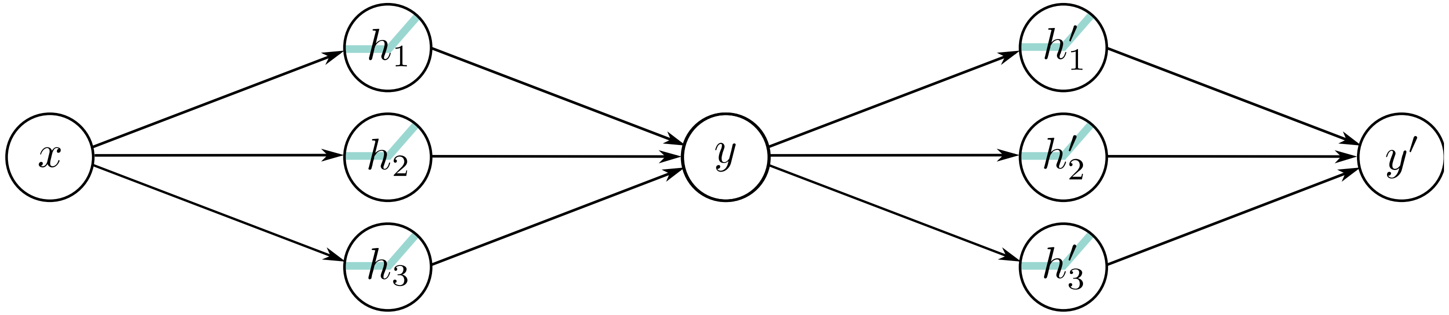


a)
x
h_1
h_2
h_3
y
h'_1
h'_2
h'_3
y'
b)
Output, y
Input, x
c)
Output, y'
Input, y
d)
Output, y'
Input, x
1.0
0.0
-1.0

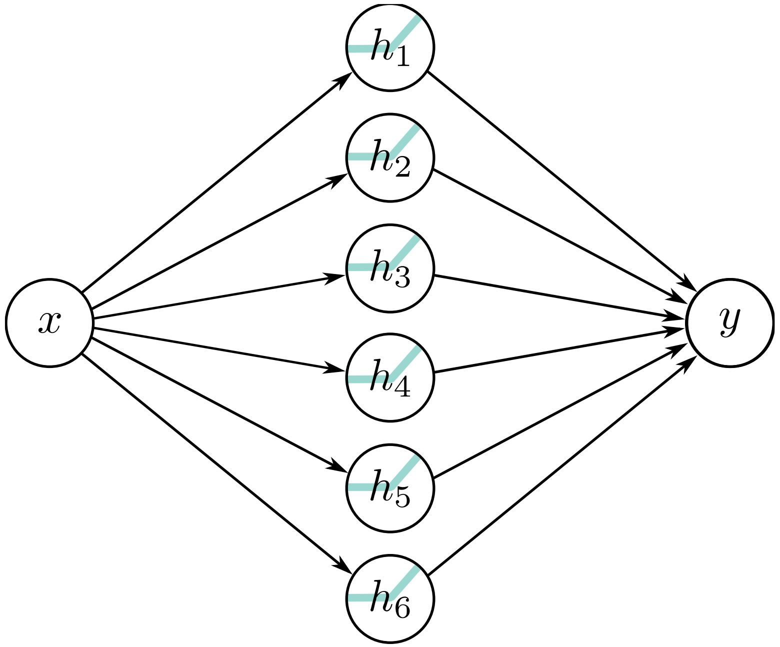
“Folding analogy”



Comparing to shallow with six hidden units



- 20 parameters
- (at least) 9 regions



- 19 parameters
- Max 7 regions

Combine two networks into one

Network 1:

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

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

Network 2:

$$\begin{aligned} h'_1 &= \mathrm{a}[\theta'_{10} + \theta'_{11}y] \\ h'_2 &= \mathrm{a}[\theta'_{20} + \theta'_{21}y] \\ h'_3 &= \mathrm{a}[\theta'_{30} + \theta'_{31}y] \end{aligned}$$

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

Hidden units of second network in terms of first:

$$\begin{aligned} h'_1 = \quad & \mathrm{a}[\theta'_{10} + \theta'_{11}y] \quad &=& \quad \mathrm{a}[\theta'_{10} + \theta'_{11}\phi_0 + \theta'_{11}\phi_1 h_1 + \theta'_{11}\phi_2 h_2 + \theta'_{11}\phi_3 h_3] \\ h'_2 = \quad & \mathrm{a}[\theta'_{20} + \theta'_{21}y] \quad &=& \quad \mathrm{a}[\theta'_{20} + \theta'_{21}\phi_0 + \theta'_{21}\phi_1 h_1 + \theta'_{21}\phi_2 h_2 + \theta'_{21}\phi_3 h_3] \\ h'_3 = \quad & \mathrm{a}[\theta'_{30} + \theta'_{31}y] \quad &=& \quad \mathrm{a}[\theta'_{30} + \theta'_{31}\phi_0 + \theta'_{31}\phi_1 h_1 + \theta'_{31}\phi_2 h_2 + \theta'_{31}\phi_3 h_3] \end{aligned}$$

Create new variables

$$
\begin{aligned}
h'_1 = &\quad \mathrm{a}[\theta'_{10} + \theta'_{11}y] &=& \quad \mathrm{a}[\theta'_{10} + \theta'_{11}\phi_0 + \theta'_{11}\phi_1 h_1 + \theta'_{11}\phi_2 h_2 + \theta'_{11}\phi_3 h_3] \\
h'_2 = &\quad \mathrm{a}[\theta'_{20} + \theta'_{21}y] &=& \quad \mathrm{a}[\theta'_{20} + \theta'_{21}\phi_0 + \theta'_{21}\phi_1 h_1 + \theta'_{21}\phi_2 h_2 + \theta'_{21}\phi_3 h_3] \\
h'_3 = &\quad \mathrm{a}[\theta'_{30} + \theta'_{31}y] &=& \quad \mathrm{a}[\theta'_{30} + \theta'_{31}\phi_0 + \theta'_{31}\phi_1 h_1 + \theta'_{31}\phi_2 h_2 + \theta'_{31}\phi_3 h_3]
\end{aligned}
$$

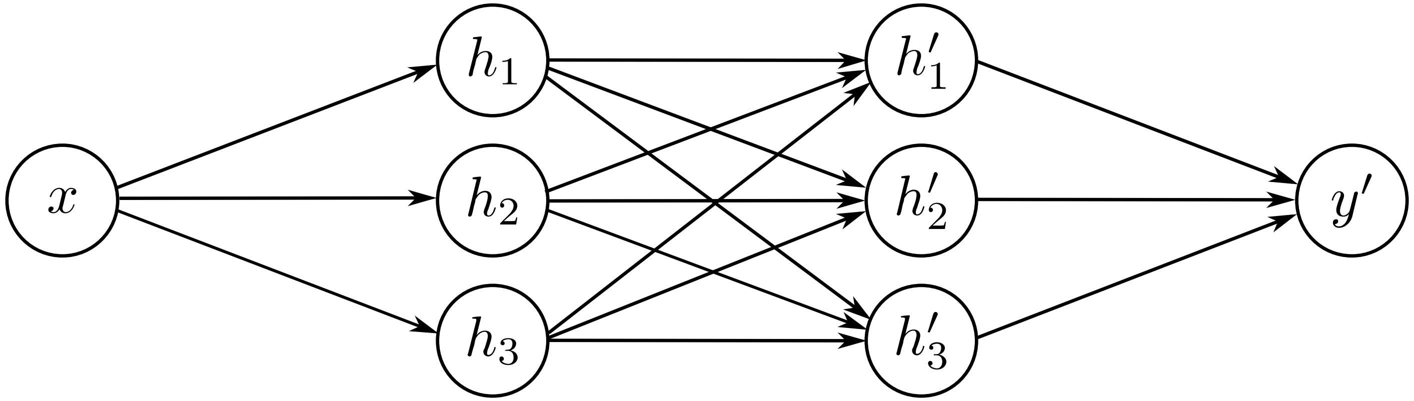
$$
\begin{aligned}
h'_1 &= \mathrm{a}[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3] \\
h'_2 &= \mathrm{a}[\psi_{20} + \psi_{21}h_1 + \psi_{22}h_2 + \psi_{23}h_3] \\
h'_3 &= \mathrm{a}[\psi_{30} + \psi_{31}h_1 + \psi_{32}h_2 + \psi_{33}h_3]
\end{aligned}
$$

Two-layer network

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

$$h'_1 = \mathrm{a}[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$
$$h'_2 = \mathrm{a}[\psi_{20} + \psi_{21}h_2 + \psi_{22}h_2 + \psi_{23}h_3]$$
$$h'_3 = \mathrm{a}[\psi_{30} + \psi_{31}h_2 + \psi_{32}h_2 + \psi_{33}h_3]$$

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



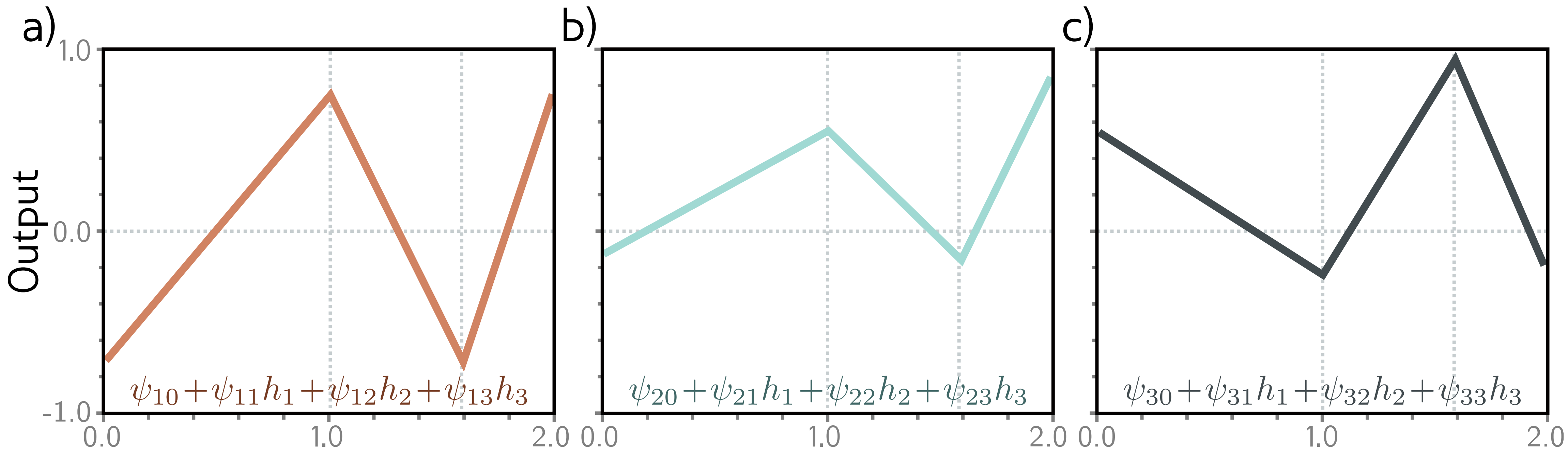
Two-layer network as one equation

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

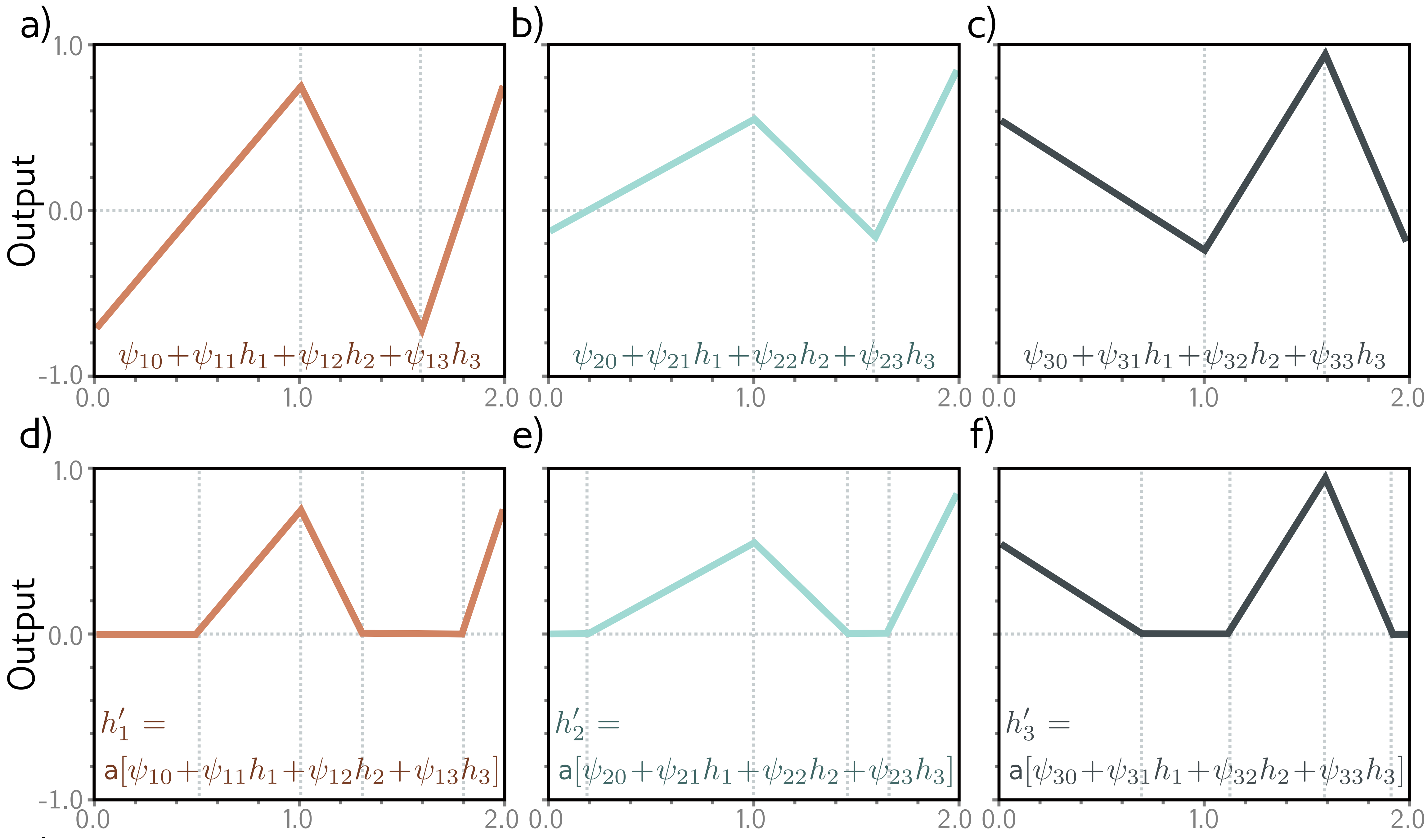
$$h'_1 = \mathrm{a}[\psi_{10} + \psi_{11}h_1 + \psi_{12}h_2 + \psi_{13}h_3]$$
$$h'_2 = \mathrm{a}[\psi_{20} + \psi_{21}h_1 + \psi_{22}h_2 + \psi_{23}h_3]$$
$$h'_3 = \mathrm{a}[\psi_{30} + \psi_{31}h_1 + \psi_{32}h_2 + \psi_{33}h_3]$$

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

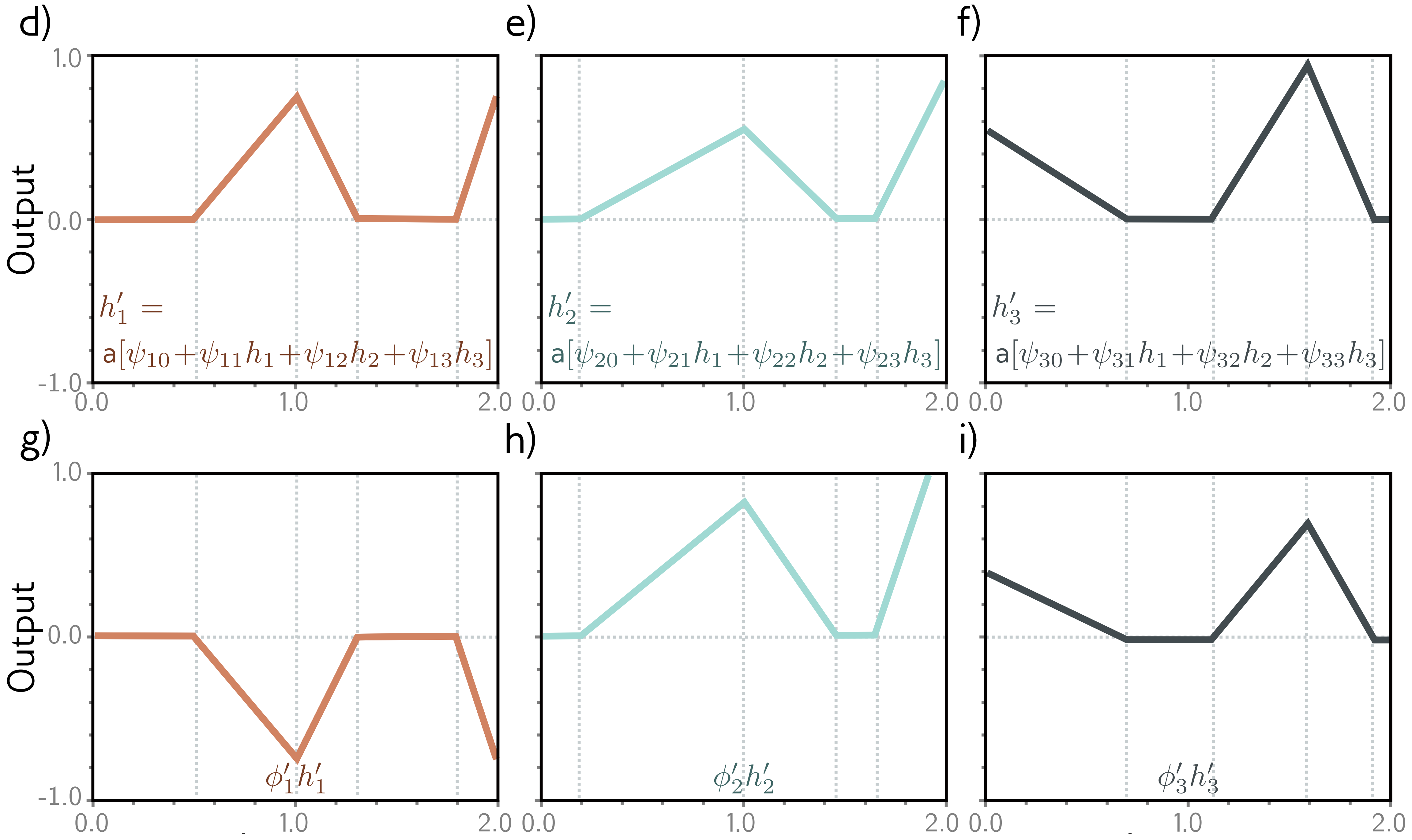
$$\begin{aligned} y' = \phi'_0 &+ \phi'_1 \mathrm{a}\left[\psi_{10} + \psi_{11}\mathrm{a}[\theta_{10} + \theta_{11}x] + \psi_{12}\mathrm{a}[\theta_{20} + \theta_{21}x] + \psi_{13}\mathrm{a}[\theta_{30} + \theta_{31}x]\right] \\ &+ \phi'_2 \mathrm{a}[\psi_{20} + \psi_{21}\mathrm{a}[\theta_{10} + \theta_{11}x]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



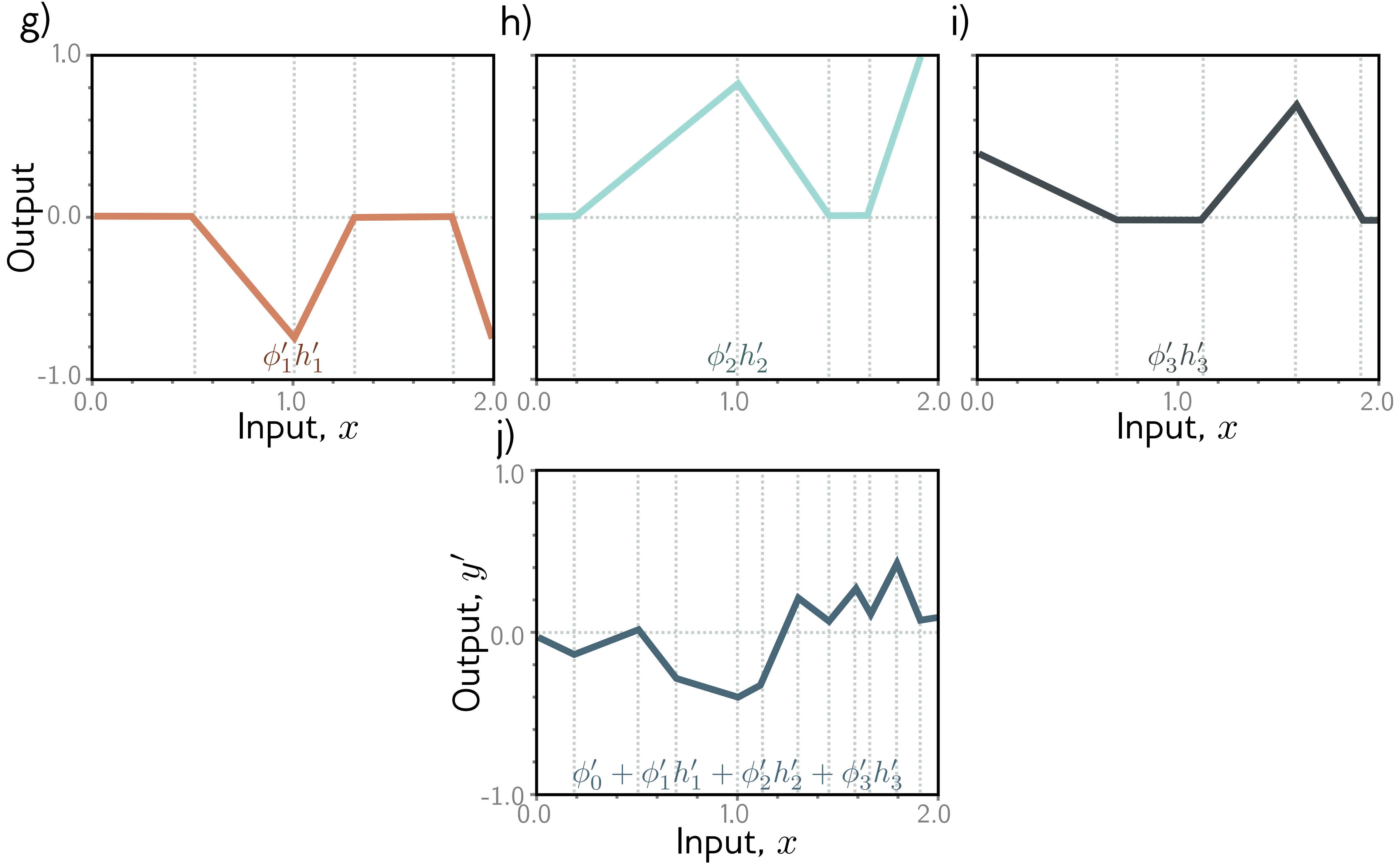
a)
b)
c)
Output
1.0
0.0
-1.0
0.0
1.0
2.0
$\psi_{10}+\psi_{11}h_1+\psi_{12}h_2+\psi_{13}h_3$
$\psi_{20}+\psi_{21}h_1+\psi_{22}h_2+\psi_{23}h_3$
$\psi_{30}+\psi_{31}h_1+\psi_{32}h_2+\psi_{33}h_3$



a)
b)
c)
d)
e)
f)
Output
Output
1.0
0.0
-1.0
0.0
1.0
2.0
$\psi_{10}+\psi_{11}h_1+\psi_{12}h_2+\psi_{13}h_3$
$\psi_{20}+\psi_{21}h_1+\psi_{22}h_2+\psi_{23}h_3$
$\psi_{30}+\psi_{31}h_1+\psi_{32}h_2+\psi_{33}h_3$
$h'_1 =$
$\mathrm{a}[\psi_{10}+\psi_{11}h_1+\psi_{12}h_2+\psi_{13}h_3]$
$h'_2 =$
$\mathrm{a}[\psi_{20}+\psi_{21}h_1+\psi_{22}h_2+\psi_{23}h_3]$
$h'_3 =$
$\mathrm{a}[\psi_{30}+\psi_{31}h_1+\psi_{32}h_2+\psi_{33}h_3]$



d)
e)
f)
g)
h)
i)
Output
Output
1.0
0.0
-1.0
0.0
1.0
2.0
$h'_1 =$
$\mathrm{a}[\psi_{10}+\psi_{11}h_1+\psi_{12}h_2+\psi_{13}h_3]$
$h'_2 =$
$\mathrm{a}[\psi_{20}+\psi_{21}h_1+\psi_{22}h_2+\psi_{23}h_3]$
$h'_3 =$
$\mathrm{a}[\psi_{30}+\psi_{31}h_1+\psi_{32}h_2+\psi_{33}h_3]$
$\phi'_1h'_1$
$\phi'_2h'_2$
$\phi'_3h'_3$



g)
h)
i)
j)
1.0
0.0
-1.0
Output
Output, y'
0.0
1.0
2.0
Input, x
$\phi_1' h_1'$
$\phi_2' h_2'$
$\phi_3' h_3'$
$\phi_0' + \phi_1' h_1' + \phi_2' h_2' + \phi_3' h_3'$

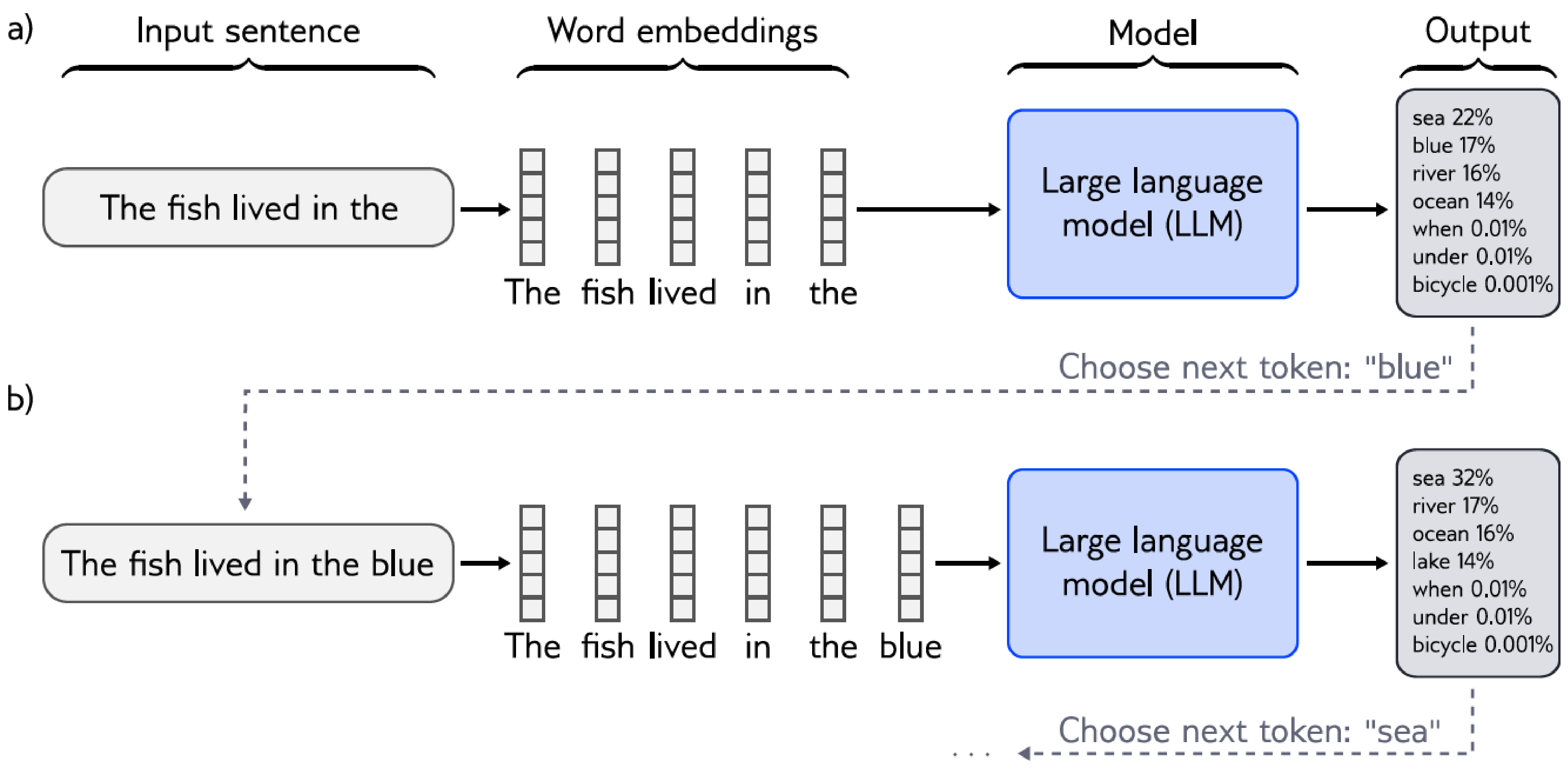
4. Neural Networks: Playground Exercises

- https://developers.google.com/machine-learning/crash-course/introduction-to-neural-networks/playground-exercises

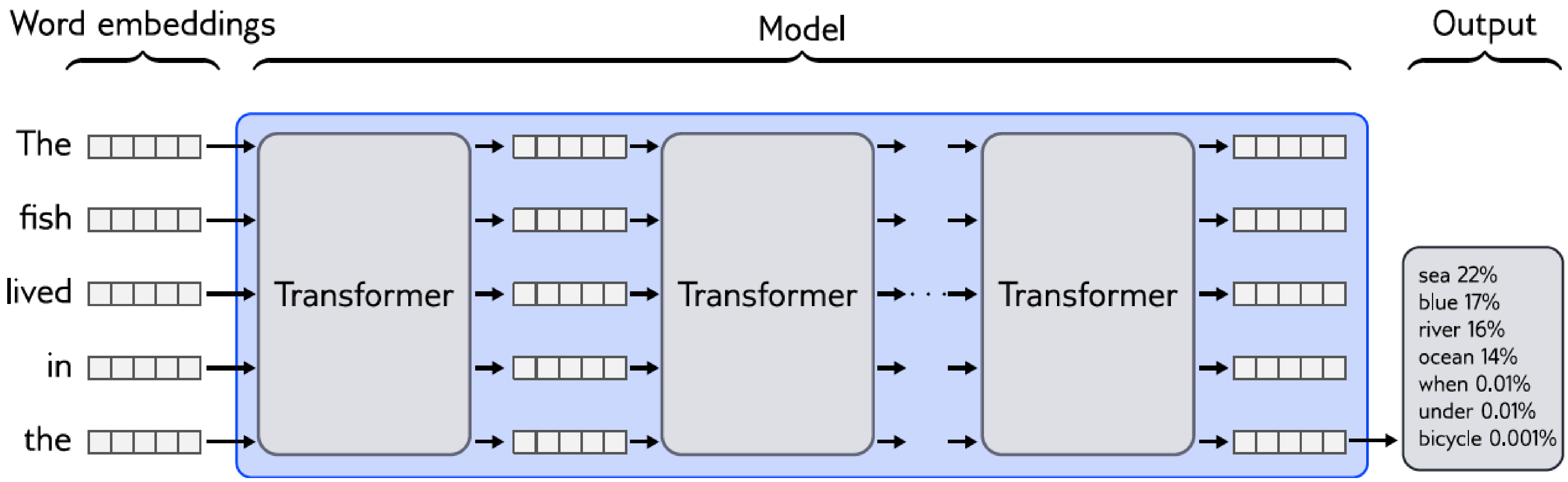
5. nanoGPT demo

- by Andrei Karpathy
- https://www.youtube.com/watch?v=kCc8FmEb1nY
- https://www.youtube.com/watch?v=6jTQ61tBeoQ

Decoder model



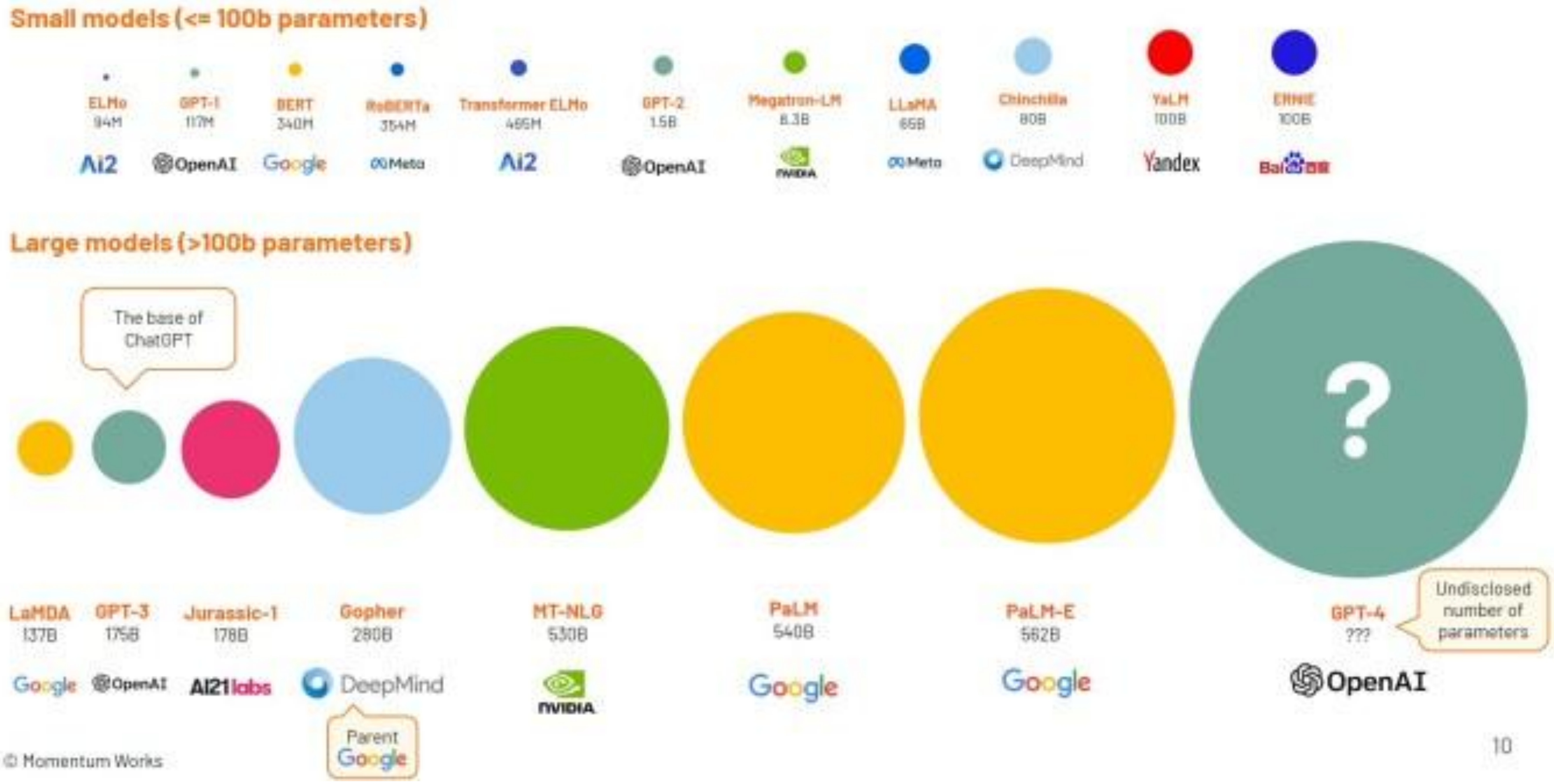
Predicting next



GPT3 (Brown et al. 2020)

- Sequence lengths are 2048 tokens long
- Batch size is 3.2 million tokens.
- 96 transformer layers (some of which implement a sparse version of attention), each of which processes a word embedding of size 12288.
- 96 heads in the self-attention layers and the value, query, and key dimension is 128.
- 300 billion tokens
- 175 billion parameters

Large Language Models are becoming very large indeed



Conclusion

- Demos of linear regression, shallow NN and deep NN
- NN training parameters
- Generative LLM