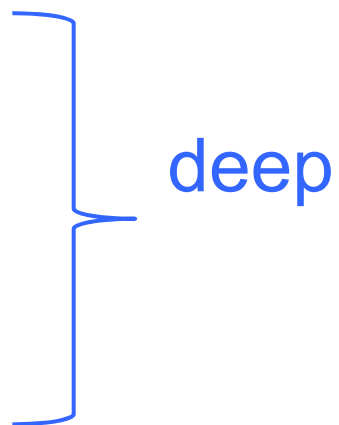


Today

- Neural networks and deep learning
 - Single layer neural networks
 - using keras package
 - architectures for different questions
 - regression, classification, multifunction
 - Multi-layer neural networks
 - Convolutional neural networks
 - U-net
 - Transformers
- 
- deep

Neural network libraries

- PyTorch

- torch (LUA, C): original Swiss nonprofit
- Facebook/Meta, Linux foundation
- Python (C++ backend)
- most popular framework last 5 years
- R torch library (PyTorch port)
 - developed over last few years
 - capability increasing
 - new book: Keydana 2023

Neural network libraries

- Tensorflow, keras
 - tensor: multidimensional array
 - Google
 - original: tensorflow C++ library
 - Python tensorflow library (2.0, 2019)
 - Python keras library (easier interface)
 - R tensorflow library > tensorflow R API
 - R keras library > Python keras

We will use keras since both R and Python; similar to PyTorch API

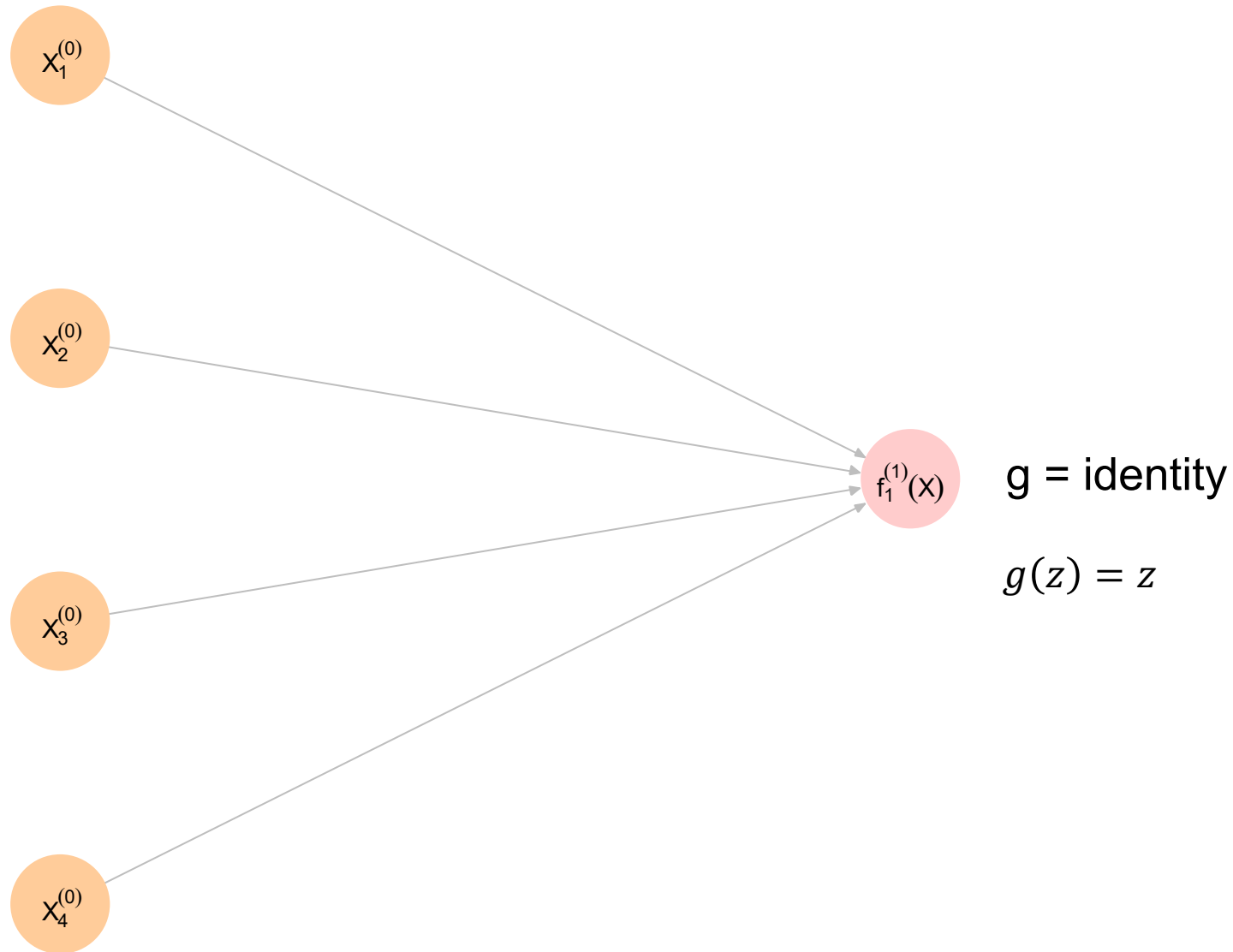
Reminder

- Data science framework
 - model algorithm
 - training algorithm
 - inference algorithm

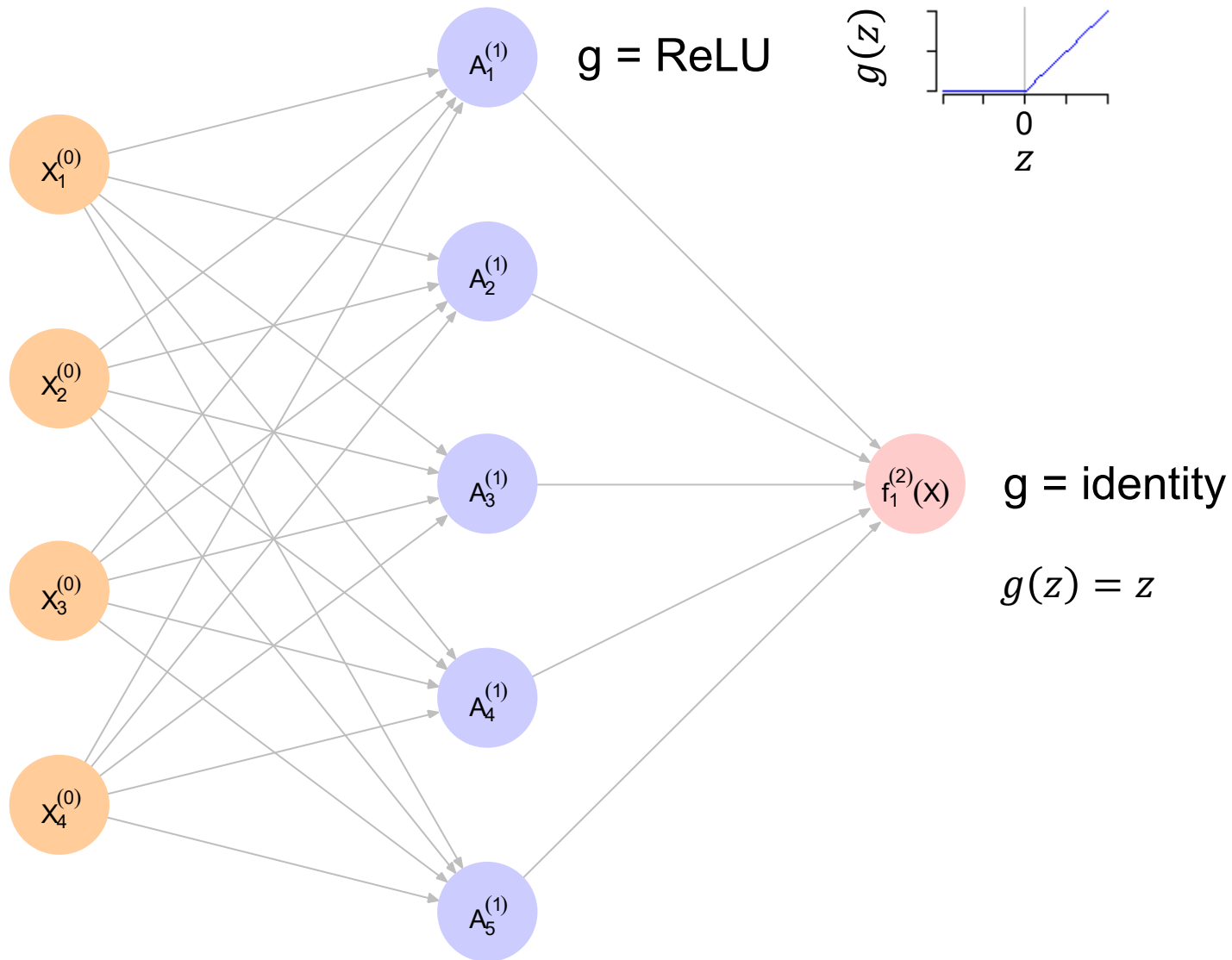
Model algorithm: architecture

- How many units and how they are connected
- What activation functions are applied to each unit

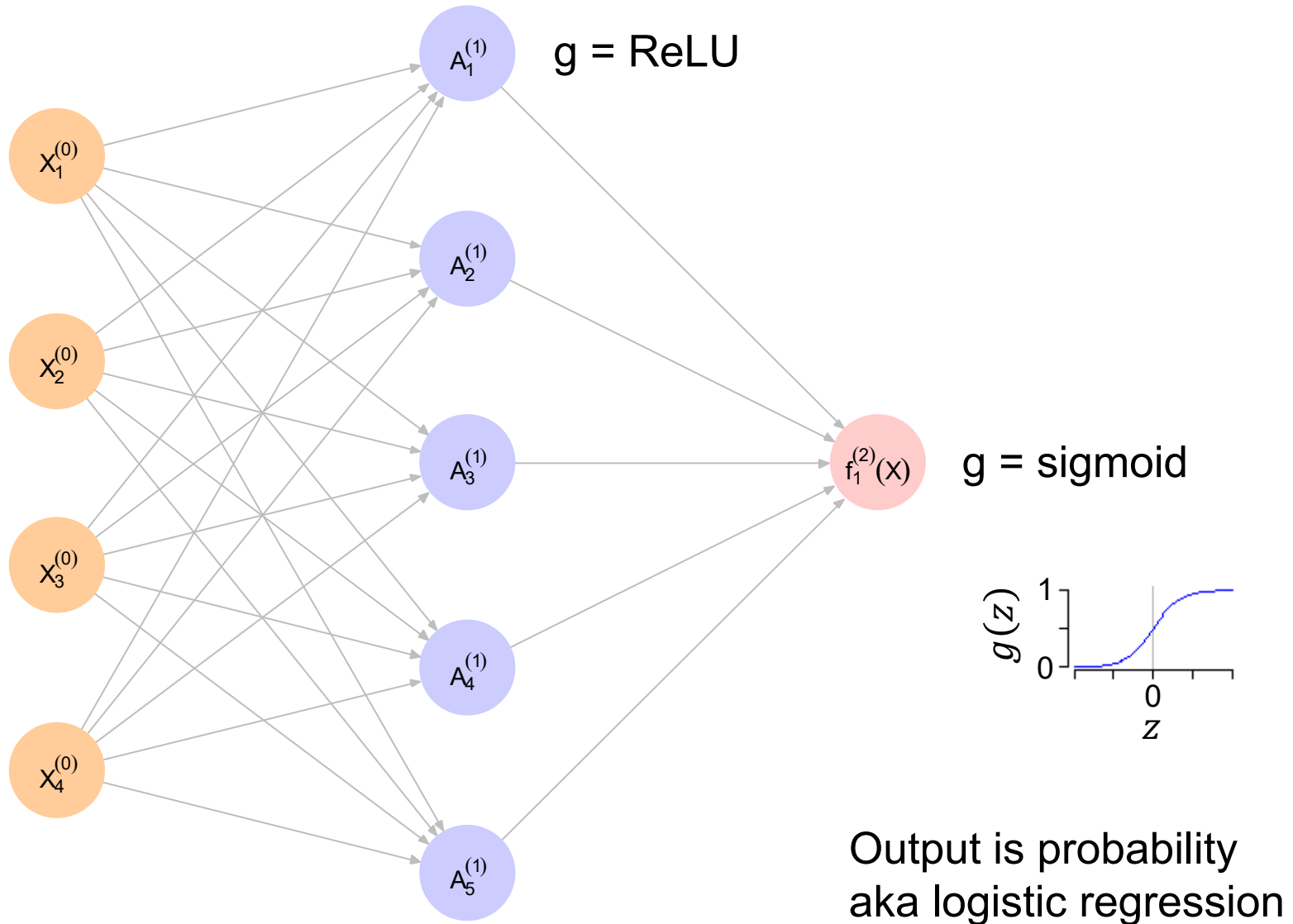
Architecture: ordinary linear regression



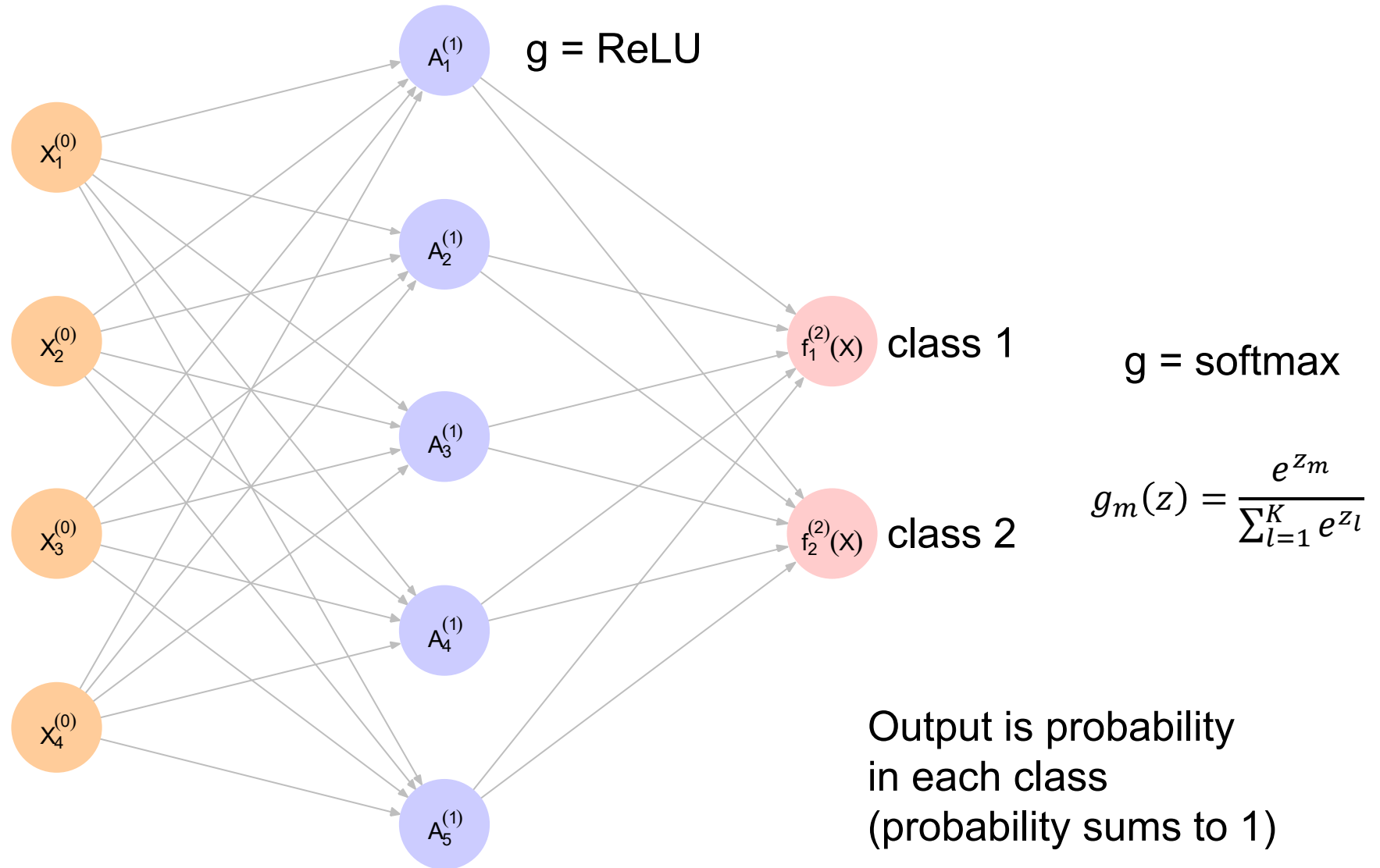
Architecture: regression



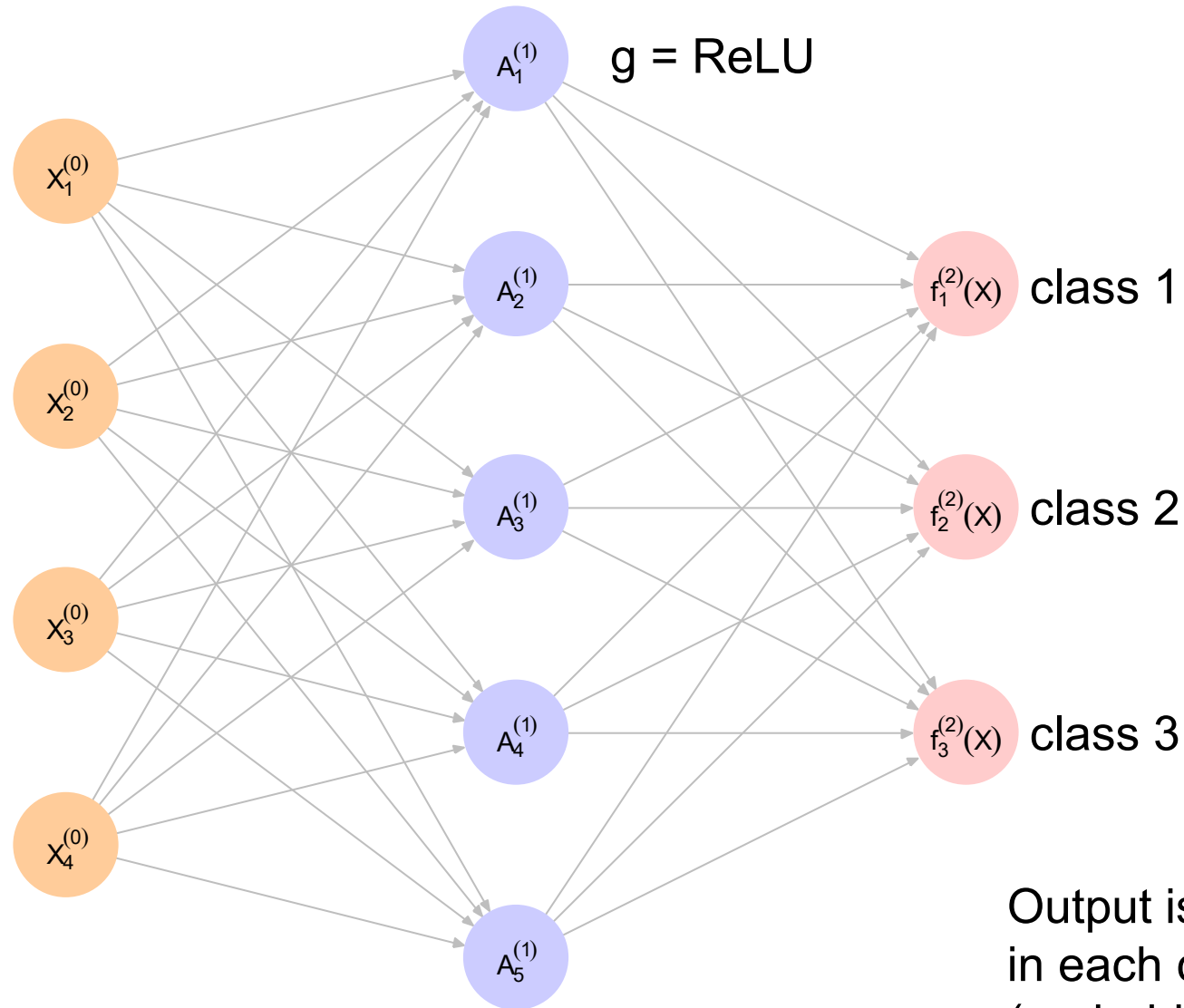
Architecture: binary classification (opt 1)



Architecture: binary classification (opt 2)



Architecture: **multicategory classification**

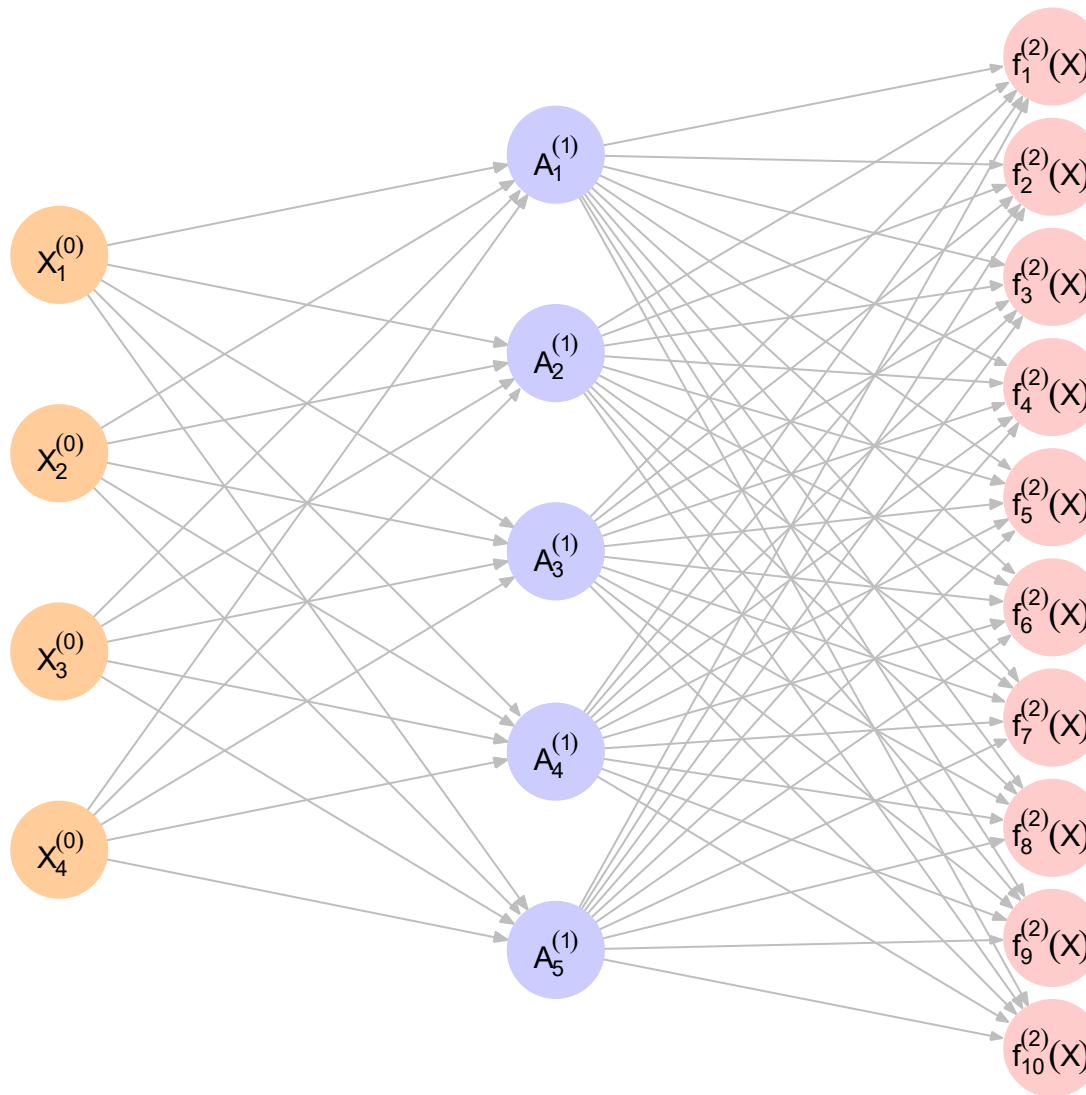


$g = \text{softmax}$

$$g_m(z) = \frac{e^{z_m}}{\sum_{l=1}^K e^{z_l}}$$

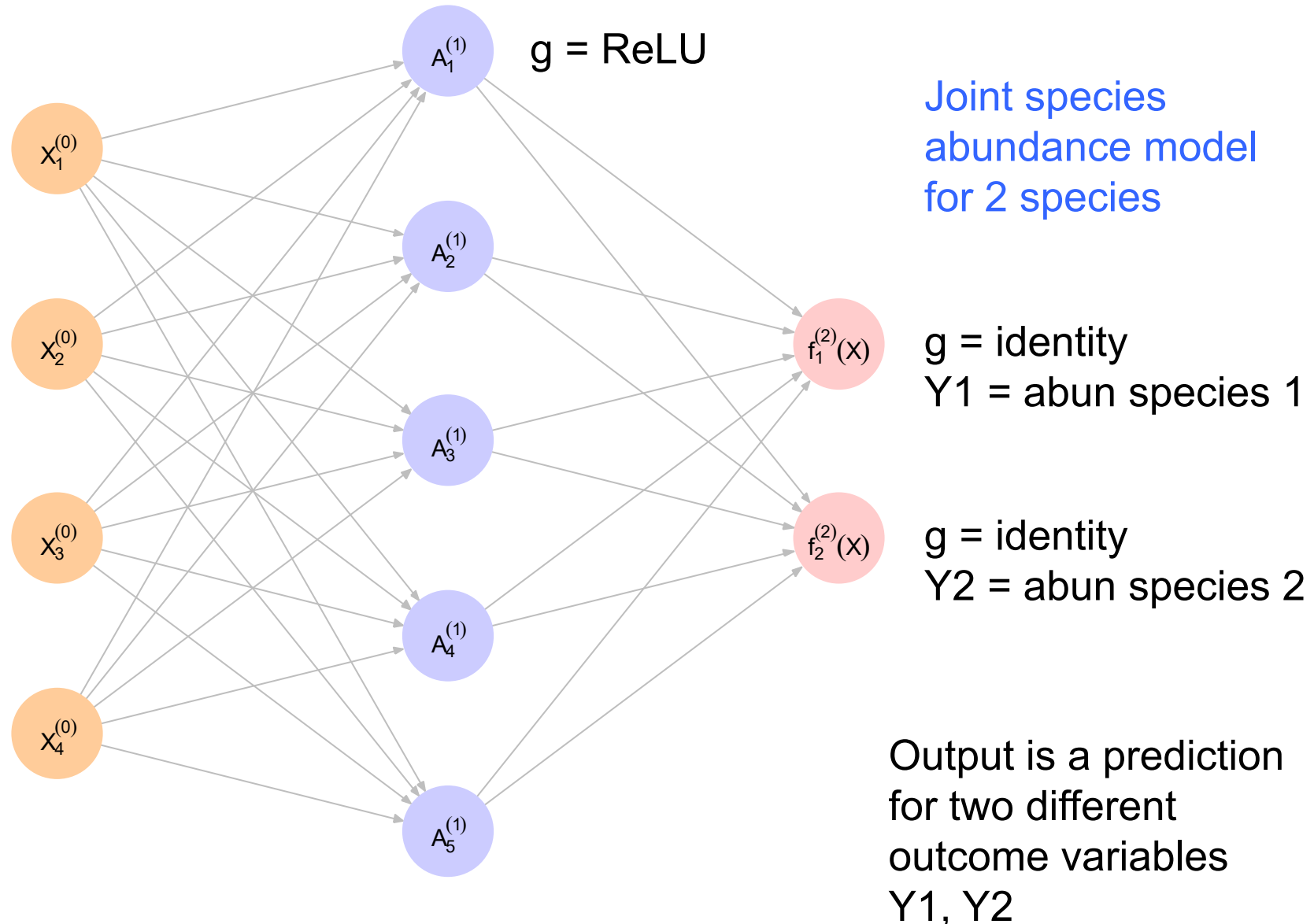
Output is probability
in each class
(probability sums to 1)

Architecture: **multicategory classification**

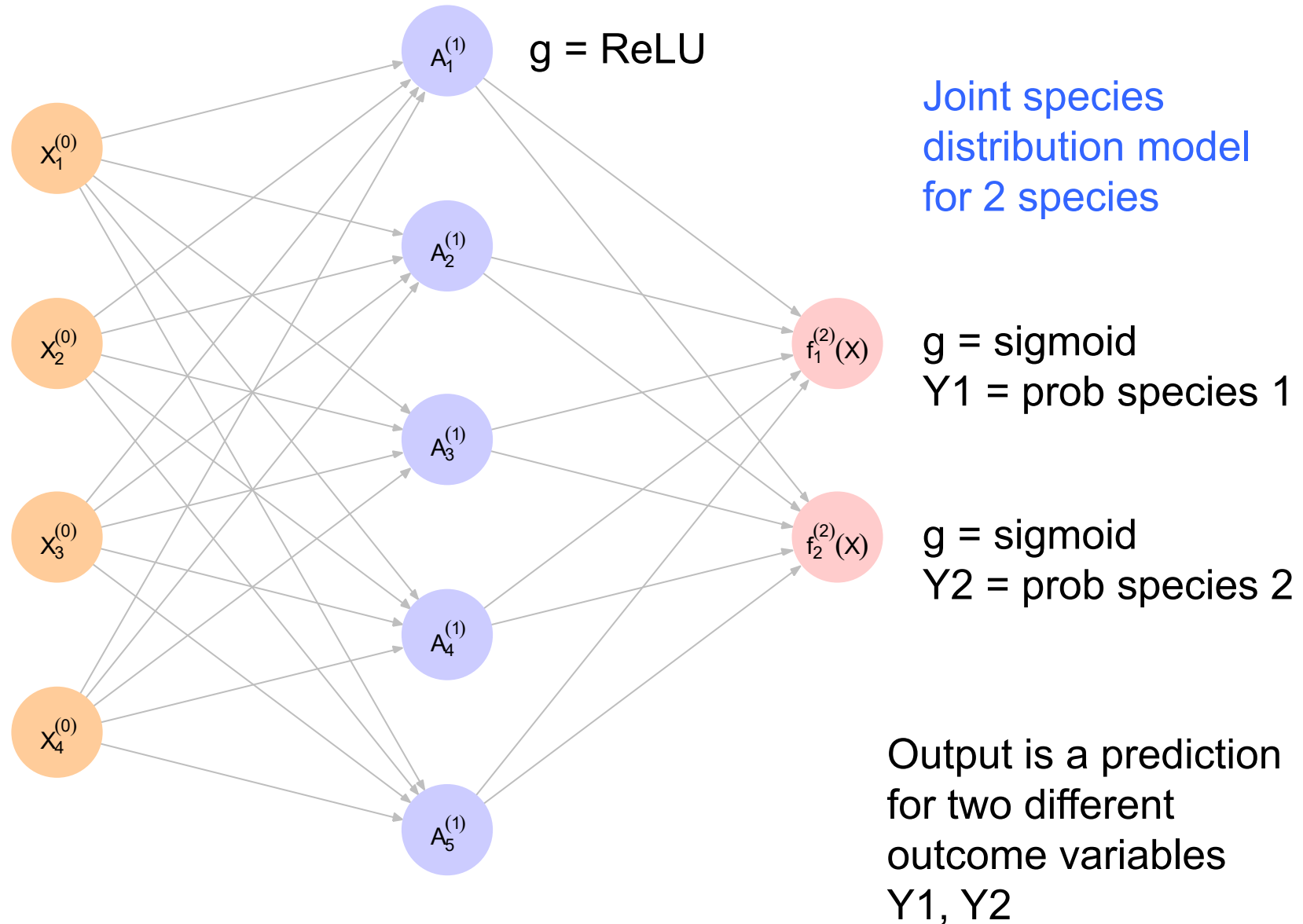


Could have any
number of categories

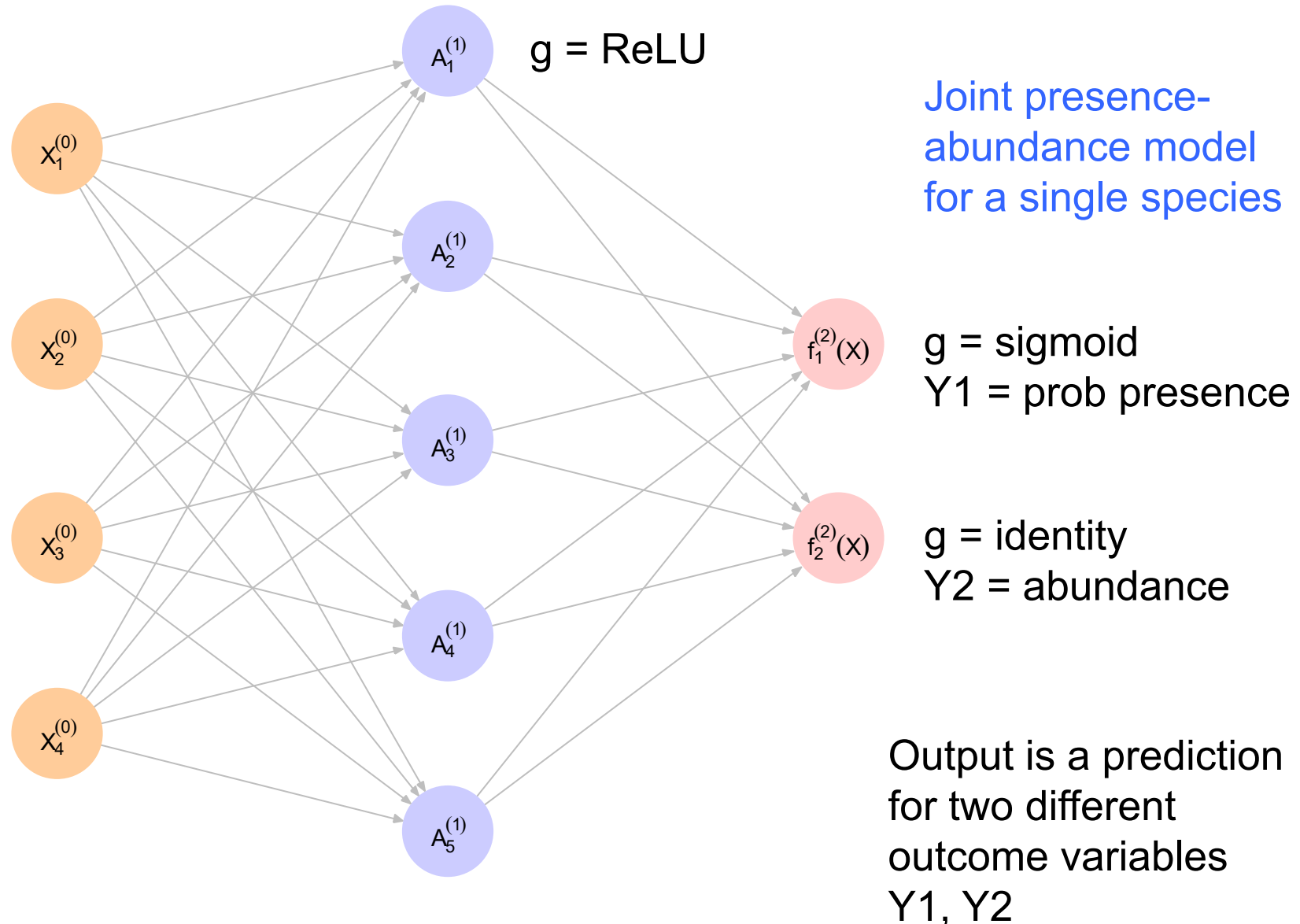
Architecture: multifunction (examples)



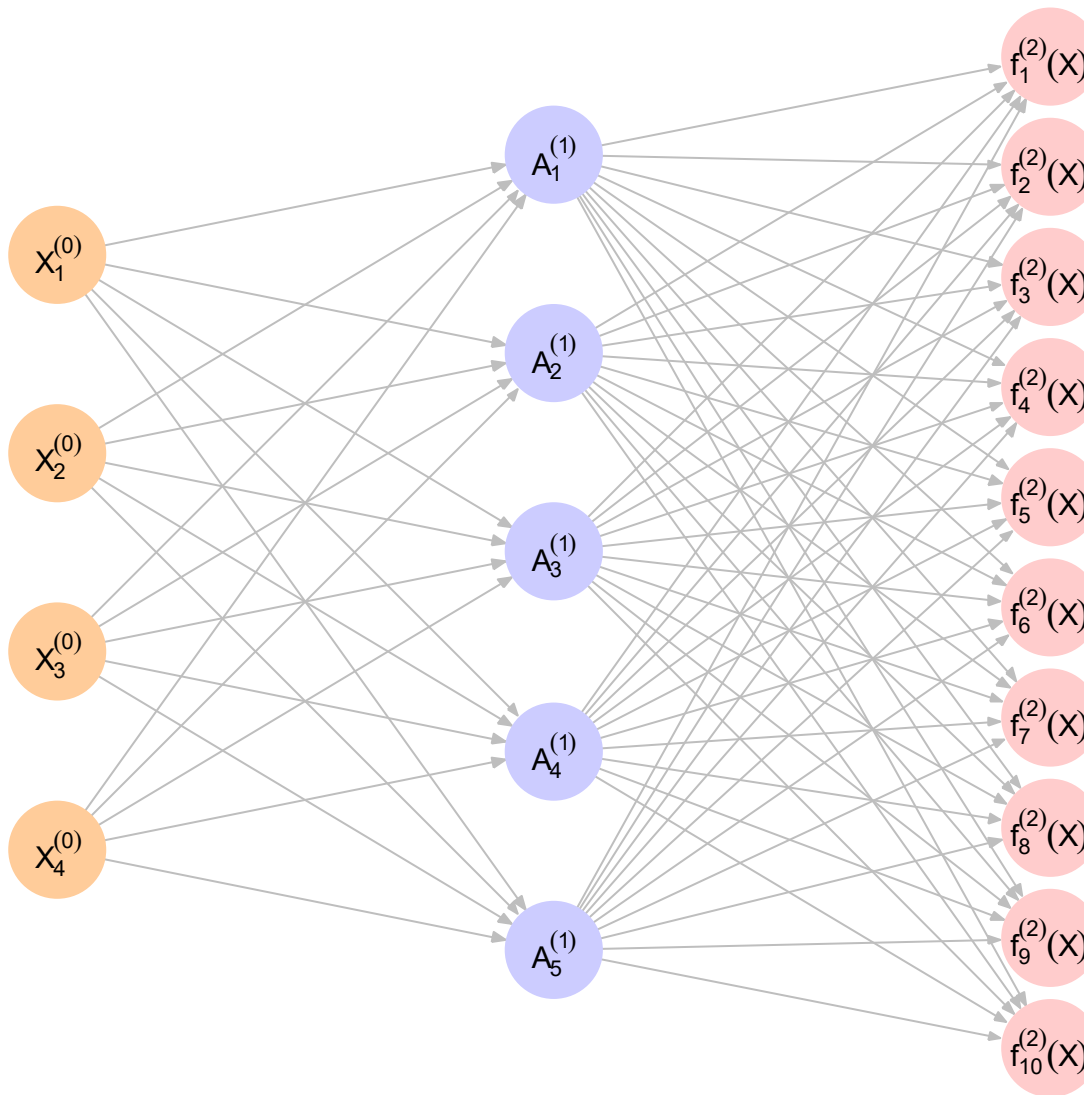
Architecture: multifunction (examples)



Architecture: multifunction (examples)



Architecture: multifunction (examples)



Could be quite complex with different $g(z)$ for many different output variables

Deep learning

- Multilayer neural networks
 - aka deep feedforward networks
 - aka multilayer perceptrons (MLP)
- Model algorithm
 - expressiveness
 - ability to approximate complex nonlinearity
 - architecture: width versus depth
 - depth = number of layers
 - width = number of nodes in a layer

Expressiveness

Universal approximation theorem (1989)

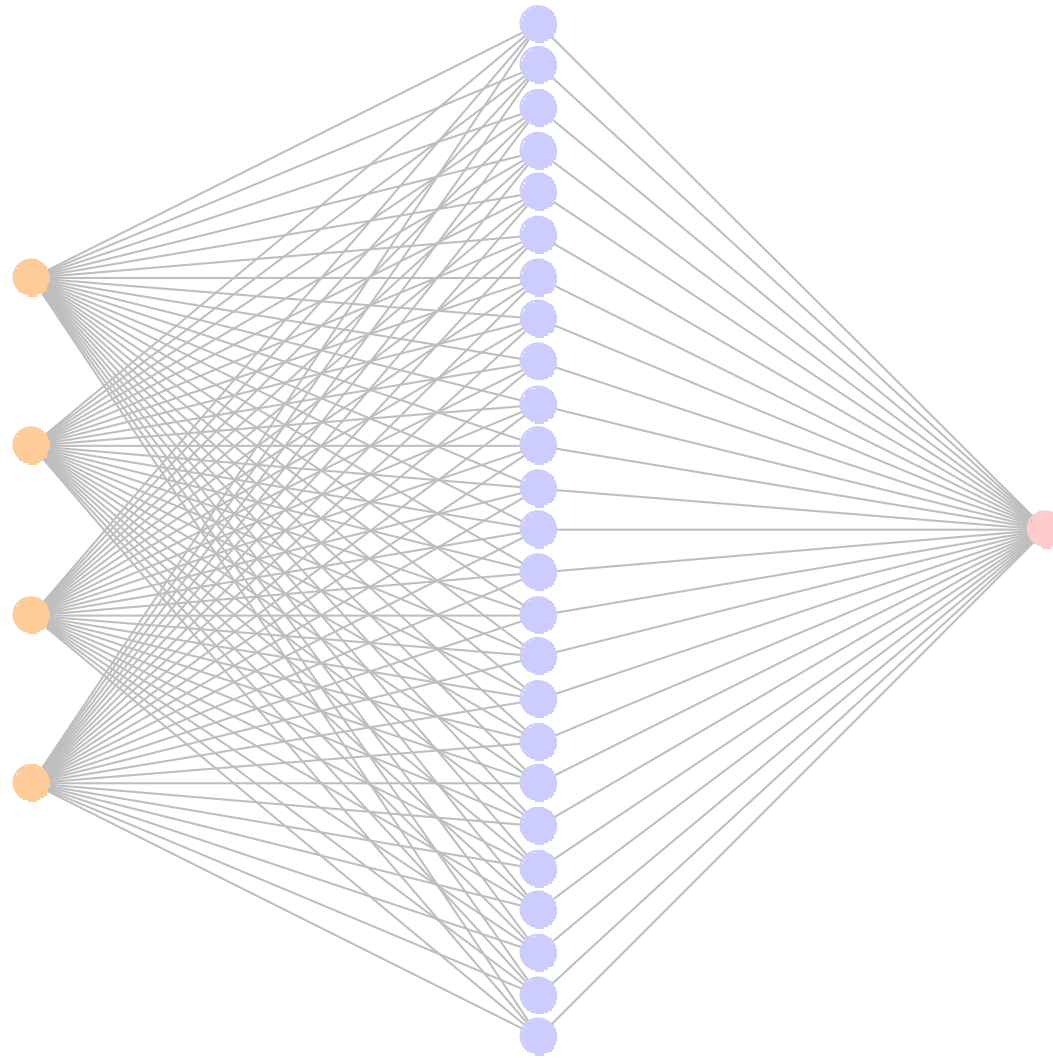
A feedforward neural network with at least one hidden layer can approximate any mapping

$$X \rightarrow f(X)$$

with arbitrarily low error provided it has enough units

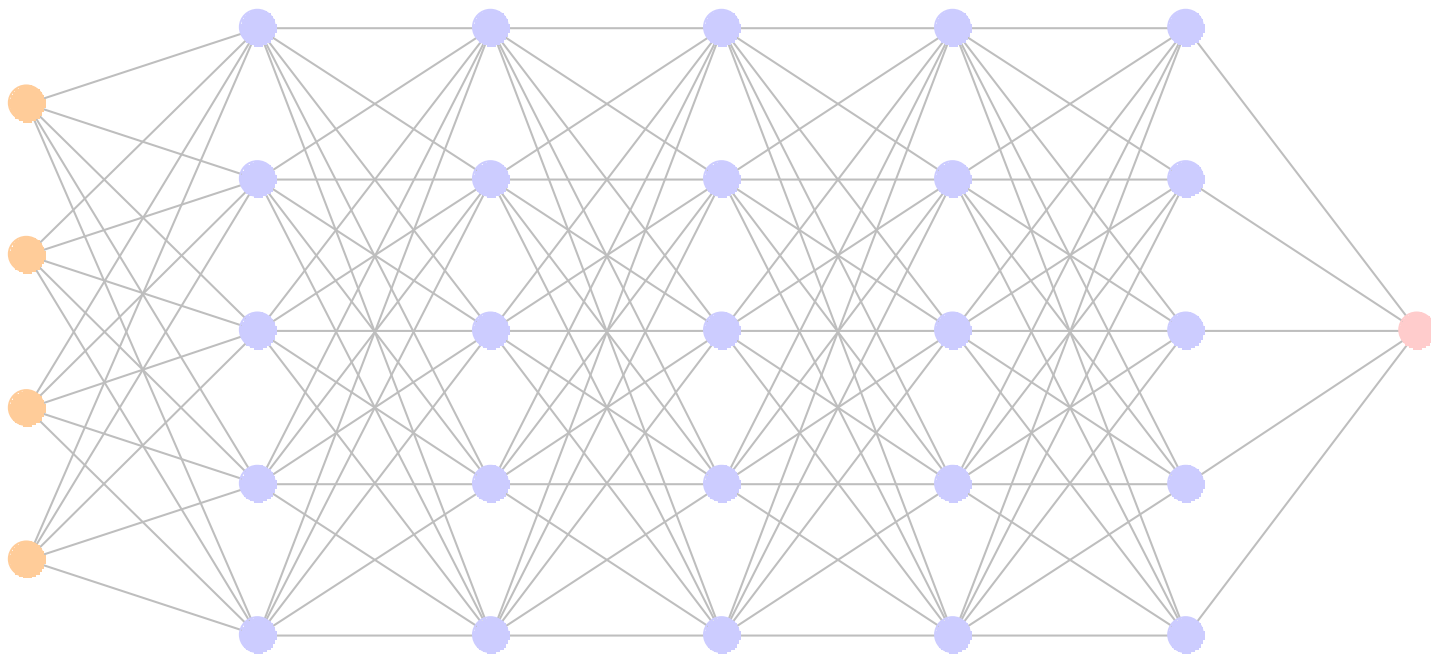
Can be **wide** or **deep**

Wide: 25 hidden units

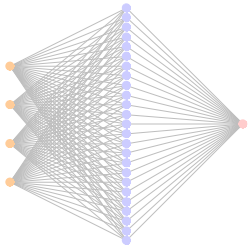


125 connections

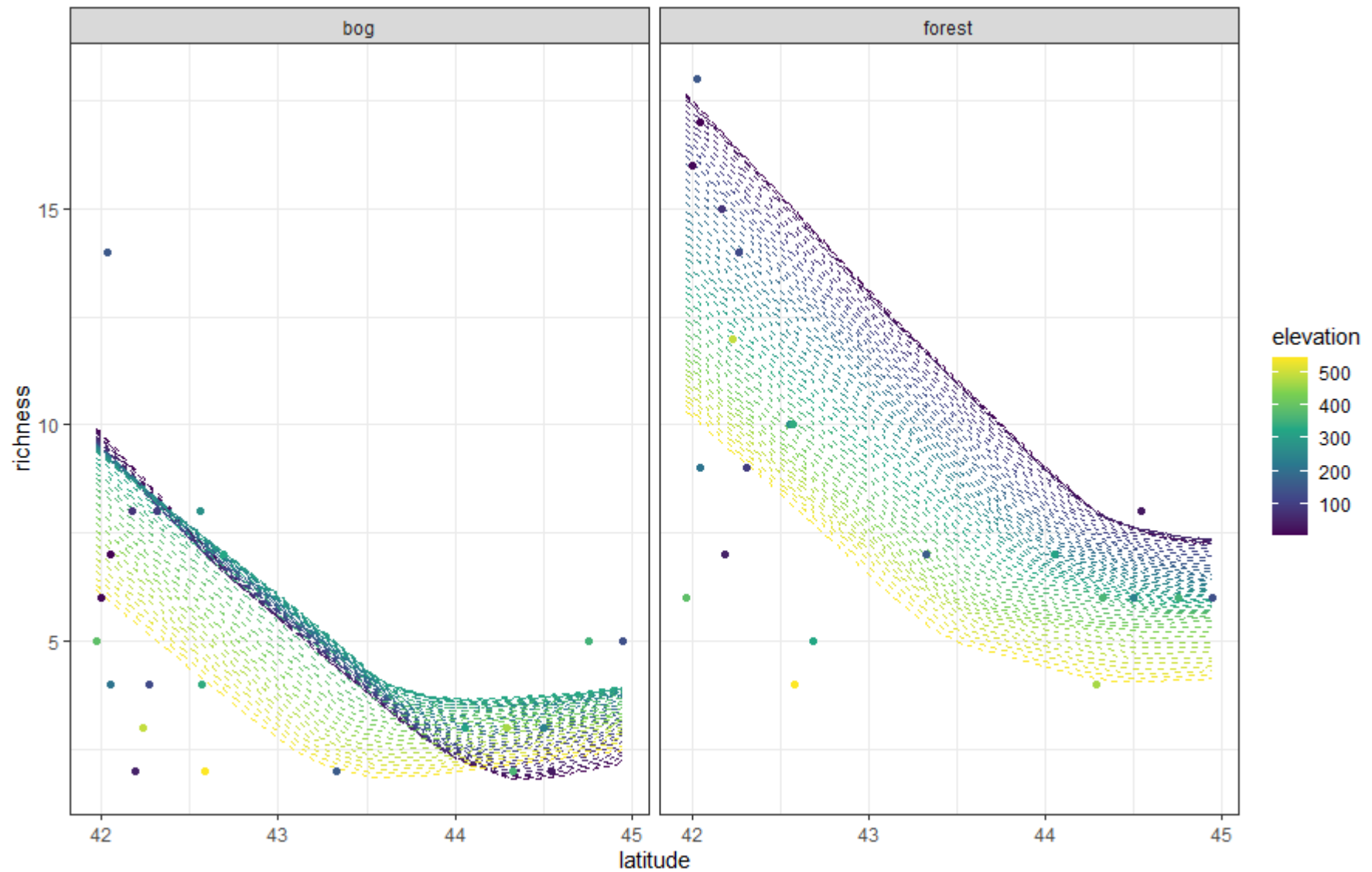
Deep: 25 hidden units

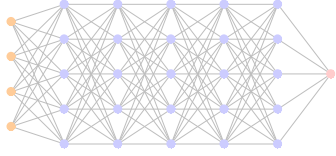


125 connections

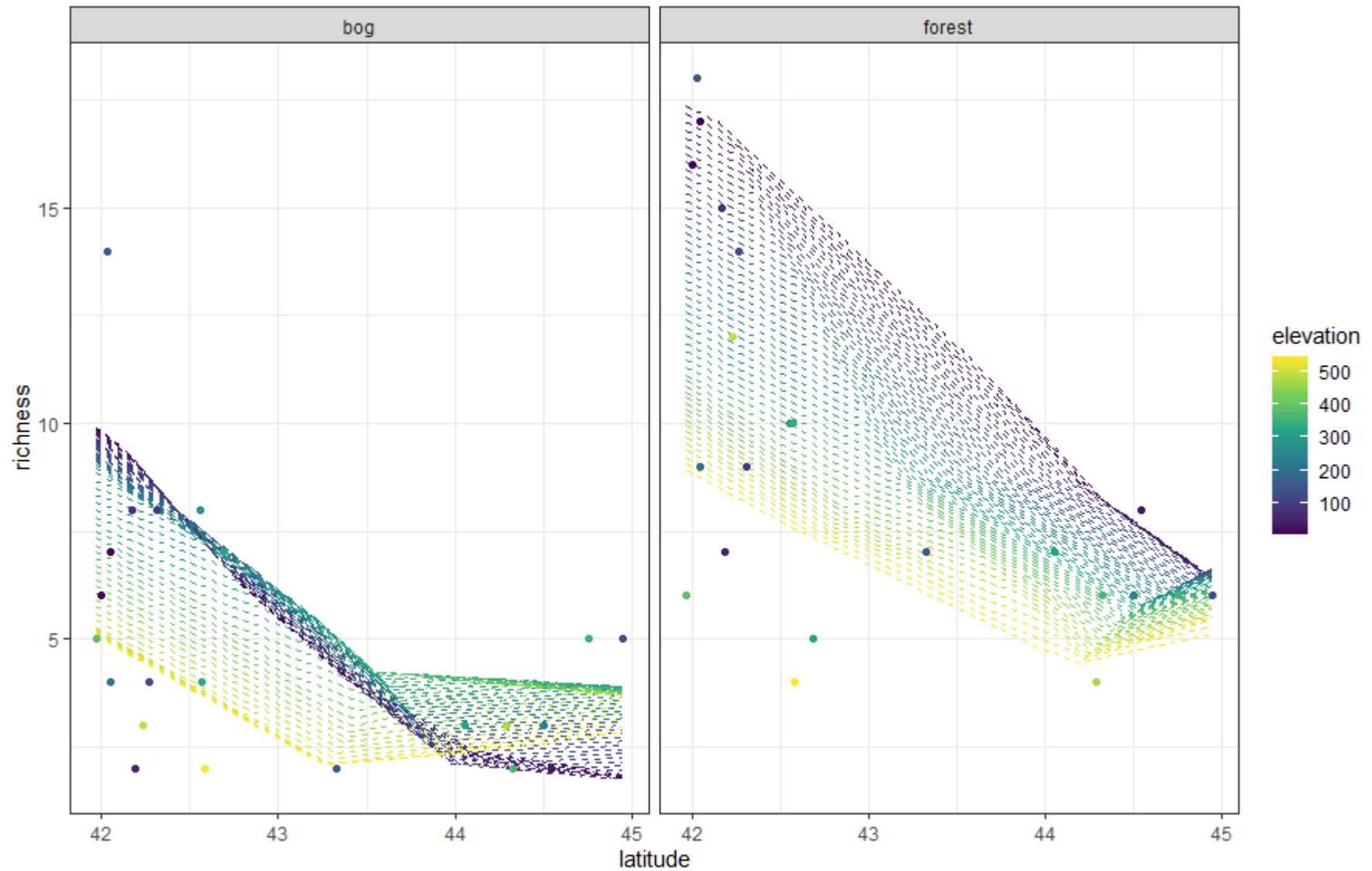


Ants data: wide





Ants data: deep

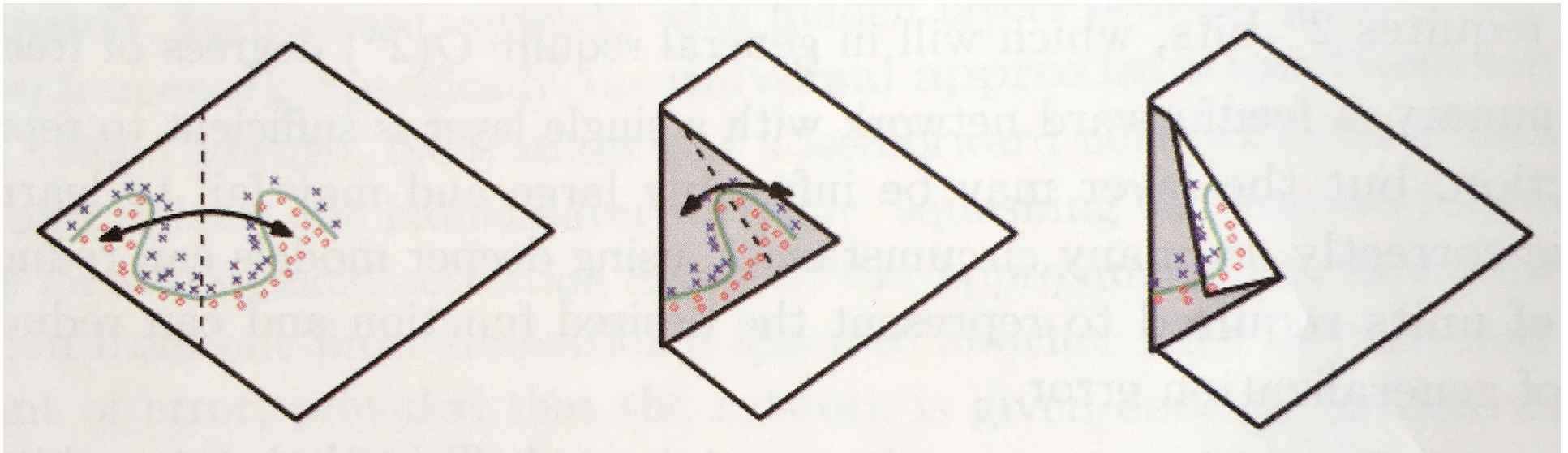


Deep: expressiveness

1 layer

2 layers

3 layers



Deeper networks (more layers) allow more folds, which can represent the same complex function more efficiently by finding symmetries