Understanding Deep Learning

Chapter 4: Deep Neural Networks

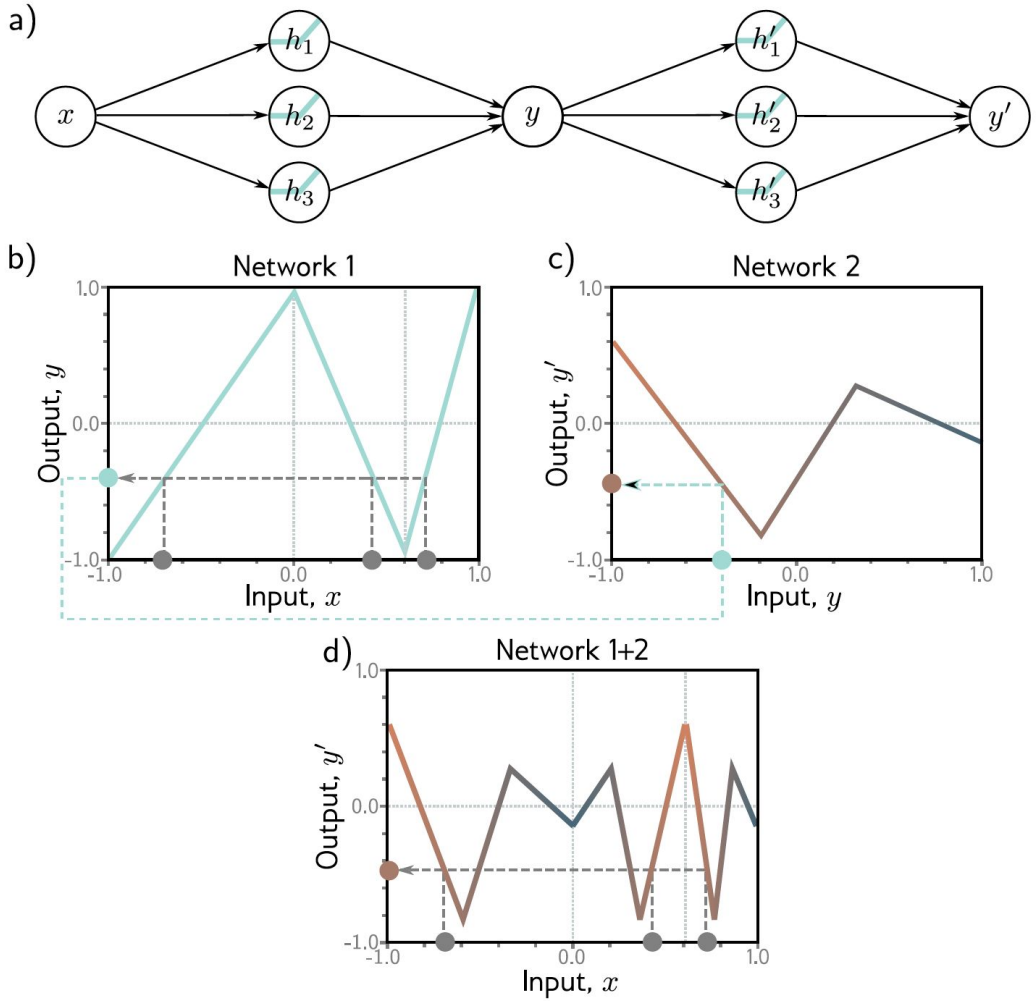

Figure 4.1 Composing two single-layer networks with three hidden units each. a) The output y of the first network constitutes the input to the second network. b) The first network maps inputs $x \in [-1, 1]$ to outputs $y \in [-1, 1]$ using a function comprised of three linear regions that are chosen so that they alternate the sign of their slope. Multiple inputs x (gray circles) now map to the same output y (cyan circle). c) The second network defines a function comprising three linear regions that takes y and returns y' (i.e., the cyan circle is mapped to the brown circle). d) The combined effect of these two functions when composed is that (i) three different inputs x are mapped to any given value of y by the first network and (ii) are processed in the same way by the second network; the result is that the function defined by the second network in panel (c) is duplicated three times, variously flipped and rescaled according to the slope of the regions of panel (b).

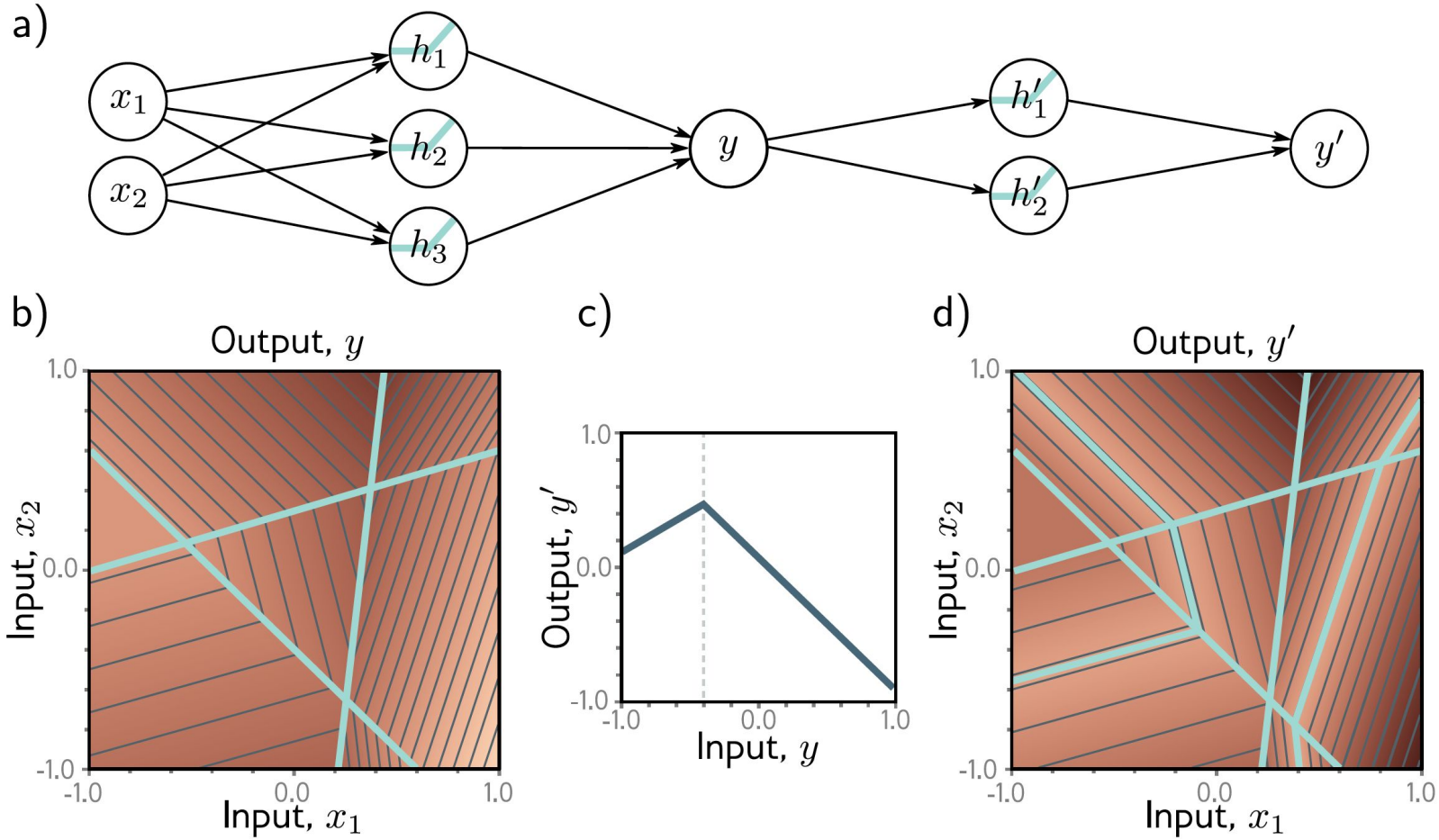

Figure 4.2 Composing neural networks with a 2D input. a) The first network (from figure 3.8) has three hidden units and takes two inputs x_1 and x_2 and returns a scalar output y. This is passed into a second network with two hidden units to produce y'. b) The first network produces a function consisting of seven linear regions, one of which is flat. c) The second network defines a function comprising two linear regions in $y \in [-1, 1]$. d) When these networks are composed, each of the six non-flat regions from the first network is divided into two new regions by the second network to create a total of 13 linear regions.

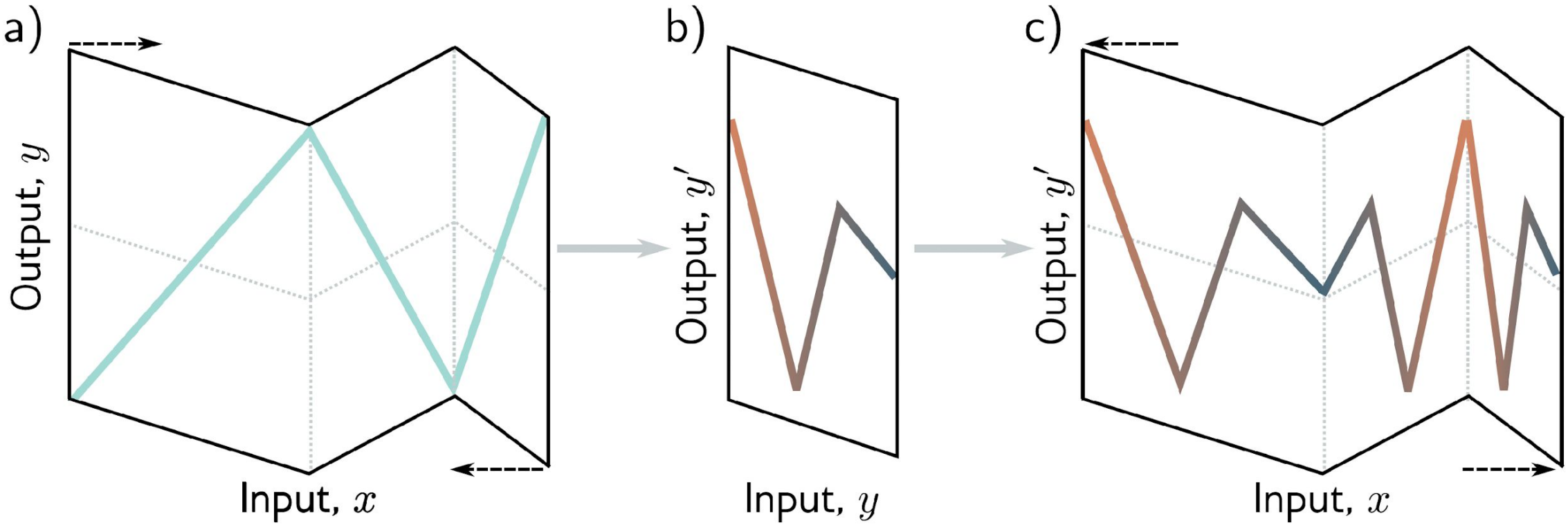

Figure 4.3 Deep networks as folding input space. a) One way to think about the first network from figure 4.1 is that it "folds" the input space back on top of itself. b) The second network applies its function to the folded space. c) The final output is revealed by "unfolding" again.

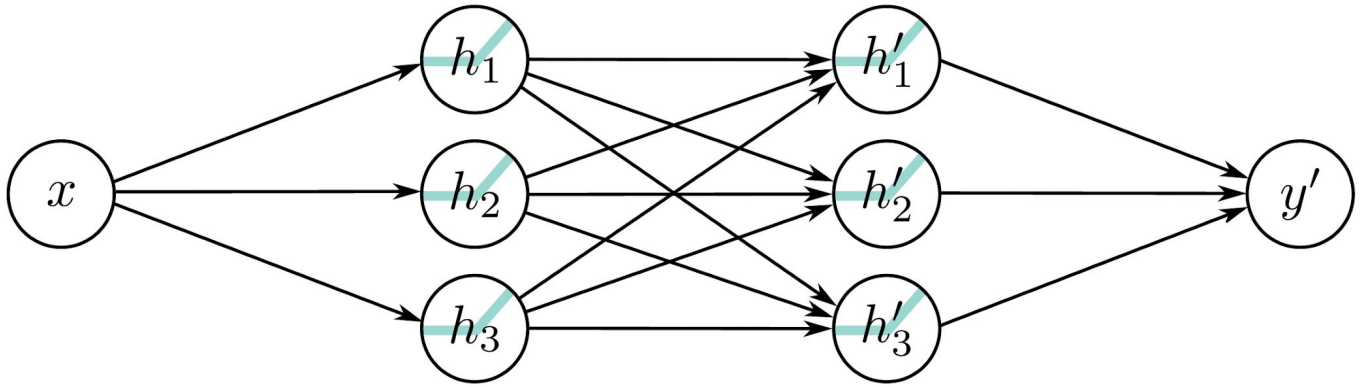

Figure 4.4 Neural network with one input, one output, and two hidden layers, each containing three hidden units.

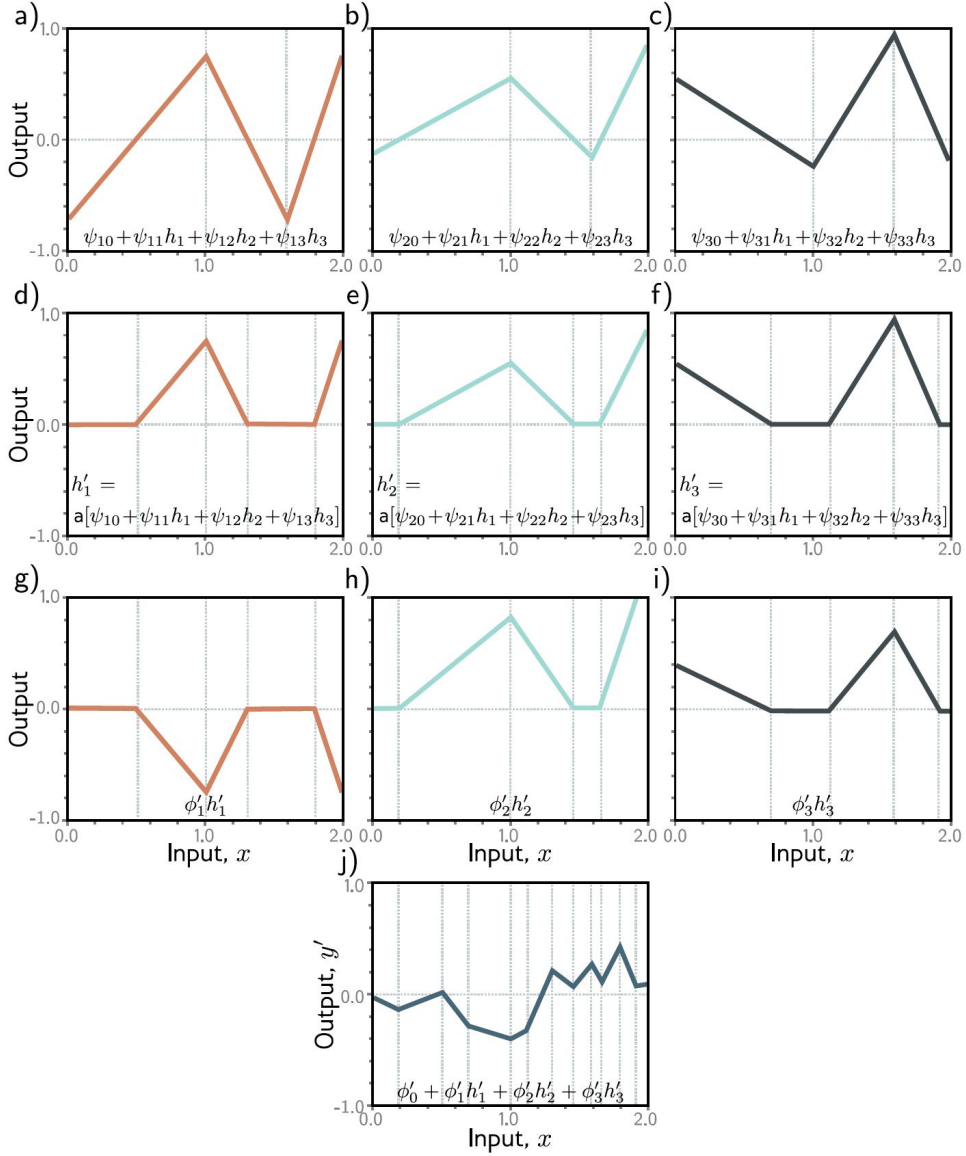

Figure 4.5 Computation for the deep network in figure 4.4. a–c) The inputs to the second hidden layer (i.e., the pre-activations) are three piecewise linear functions where the "joints" between the linear regions are at the same places (see figure 3.6). d–f) Each piecewise linear function is clipped to zero by the ReLU activation function. g–i) These clipped functions are then weighted with parameters ϕ_1', ϕ_2', and ϕ_3', respectively. j) Finally, the clipped and weighted functions are summed and an offset ϕ_0' that controls the overall height is added.

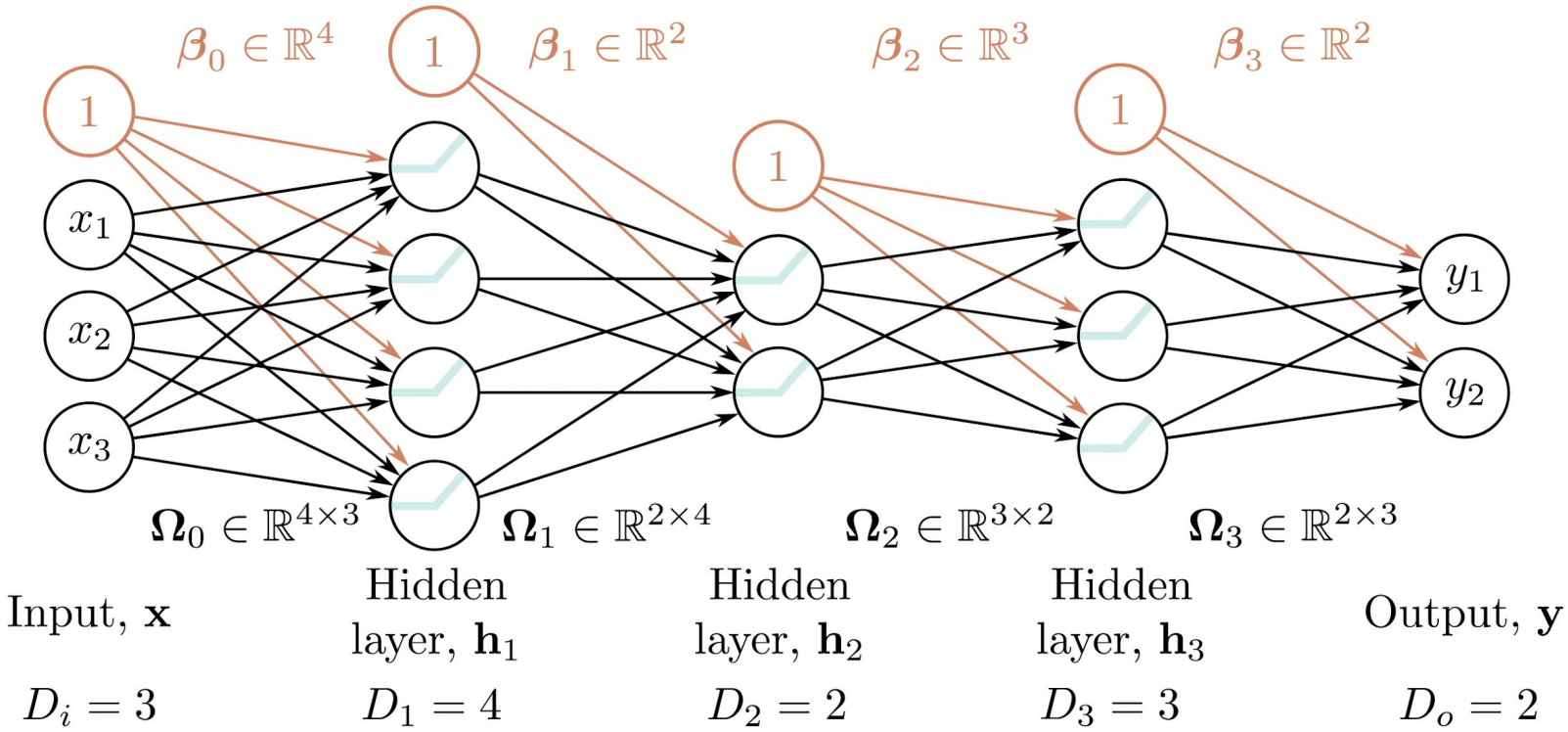

Figure 4.6 Matrix notation for network with $D_i = 3$-dimensional input $\mathbf{x}$, $D_o = 2$-dimensional output $\mathbf{y}$, and $K = 3$ hidden layers $\mathbf{h}_1, \mathbf{h}_2$, and $\mathbf{h}_3$ of dimensions $D_1 = 4$, $D_2 = 2$, and $D_3 = 3$ respectively. The weights are stored in matrices $\boldsymbol{\Omega}_k$ that pre-multiply the activations from the preceding layer to create the pre-activations at the subsequent layer. For example, the weight matrix $\boldsymbol{\Omega}_1$ that computes the pre-activations at $\mathbf{h}_2$ from the activations at $\mathbf{h}_1$ has dimension 2×4. It is applied to the four hidden units in layer one and creates the inputs to the two hidden units at layer two. The biases are stored in vectors $\boldsymbol{\beta}_k$ and have the dimension of the layer into which they feed. For example, the bias vector $\boldsymbol{\beta}_2$ is length three because layer $\mathbf{h}_3$ contains three hidden units.

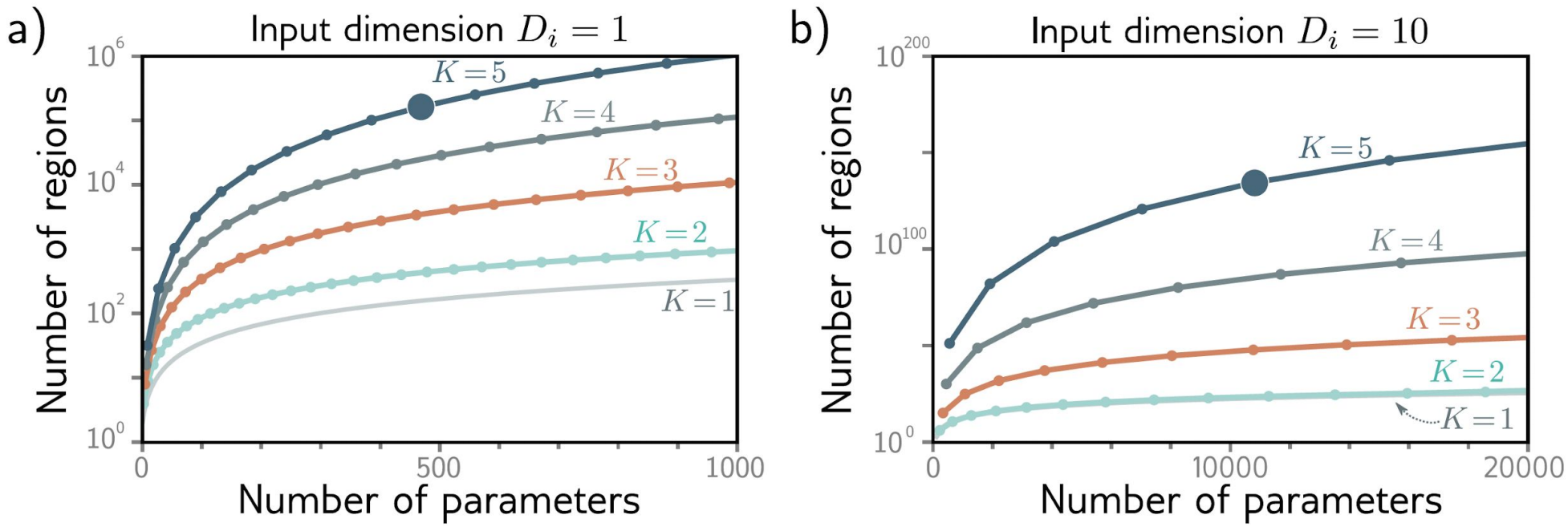

Figure 4.7 The maximum number of linear regions for neural networks increases rapidly with the network depth. a) Network with $D_i = 1$ input. Each curve represents a fixed number of hidden layers K, as we vary the number of hidden units D per layer. For a fixed parameter budget (horizontal position), deeper networks produce more linear regions than shallower ones. A network with $K = 5$ layers and $D = 10$ hidden units per layer has 471 parameters (highlighted point) and can produce 161,051 regions. b) Network with $D_i = 10$ inputs. Each subsequent point along a curve represents ten hidden units. Here, a model with $K = 5$ layers and $D = 50$ hidden units per layer has 10,801 parameters (highlighted point) and can create more than 10^{134} linear regions.

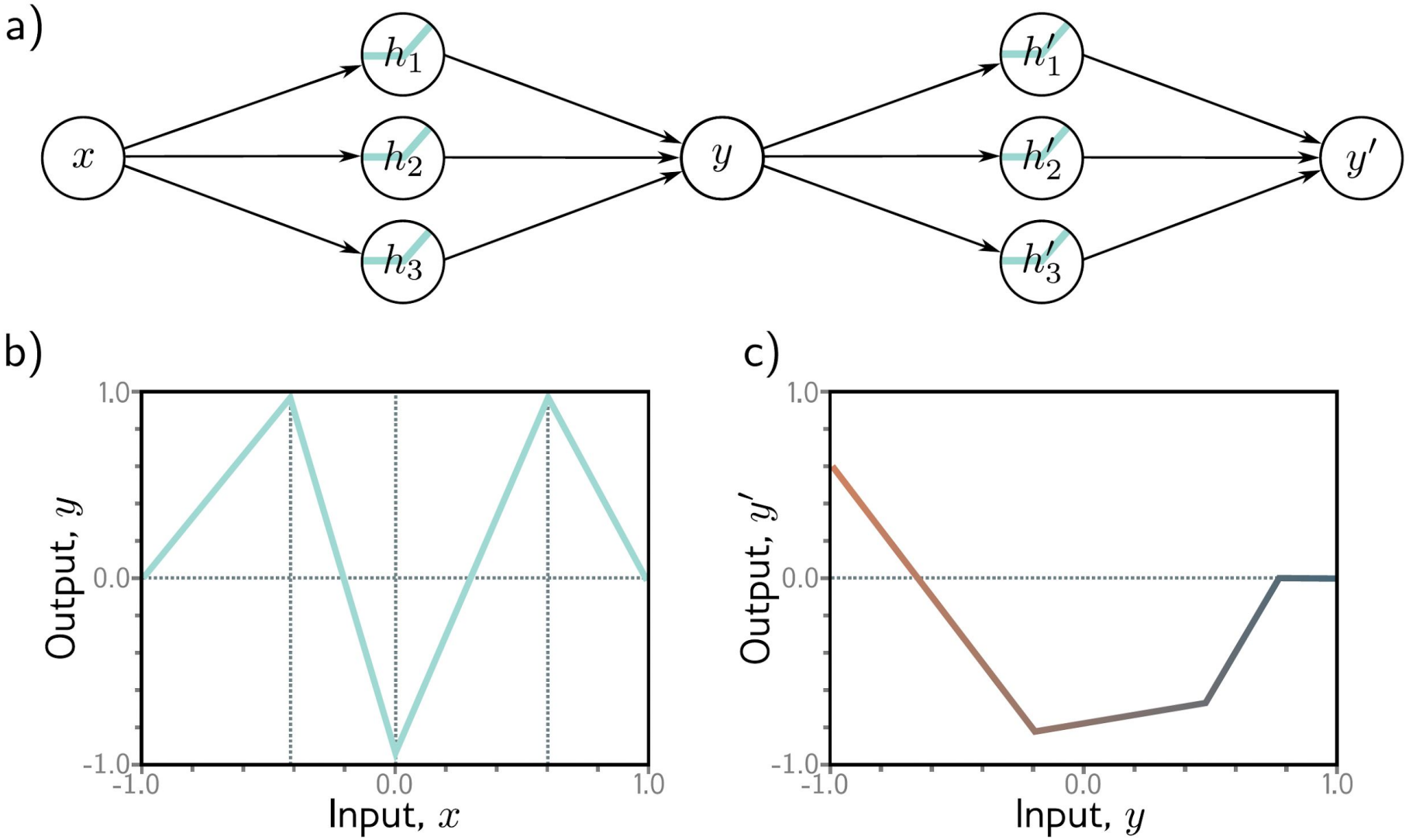

Figure 4.8 Composition of two networks for problem 4.1. a) The output y of the first network becomes the input to the second. b) The first network computes this function with output values $y \in [-1, 1]$. c) The second network computes this function on the input range $y \in [-1, 1]$.