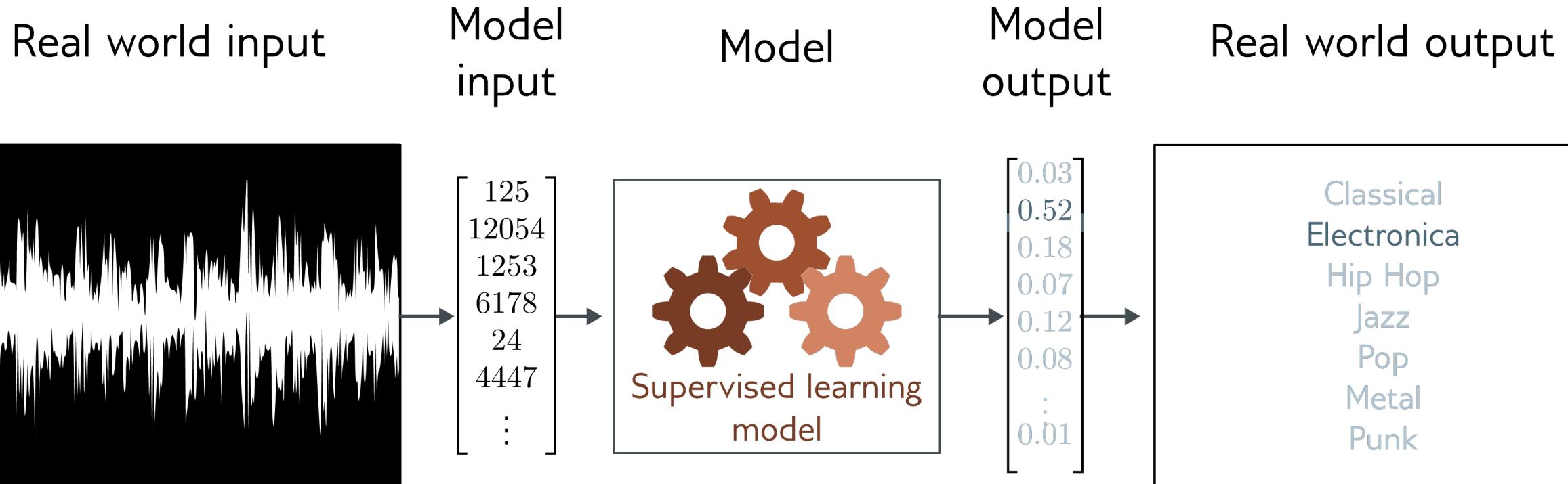


CM20315 - Machine Learning

Prof. Simon Prince
7a. Gradients



Music genre classification



- Multiclass classification problem (discrete classes, >2 possible values)
- Convolutional network

Loss function

- Training dataset of I pairs of input/output examples:

$$\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^I$$

- Loss function or cost function measures how bad model is:

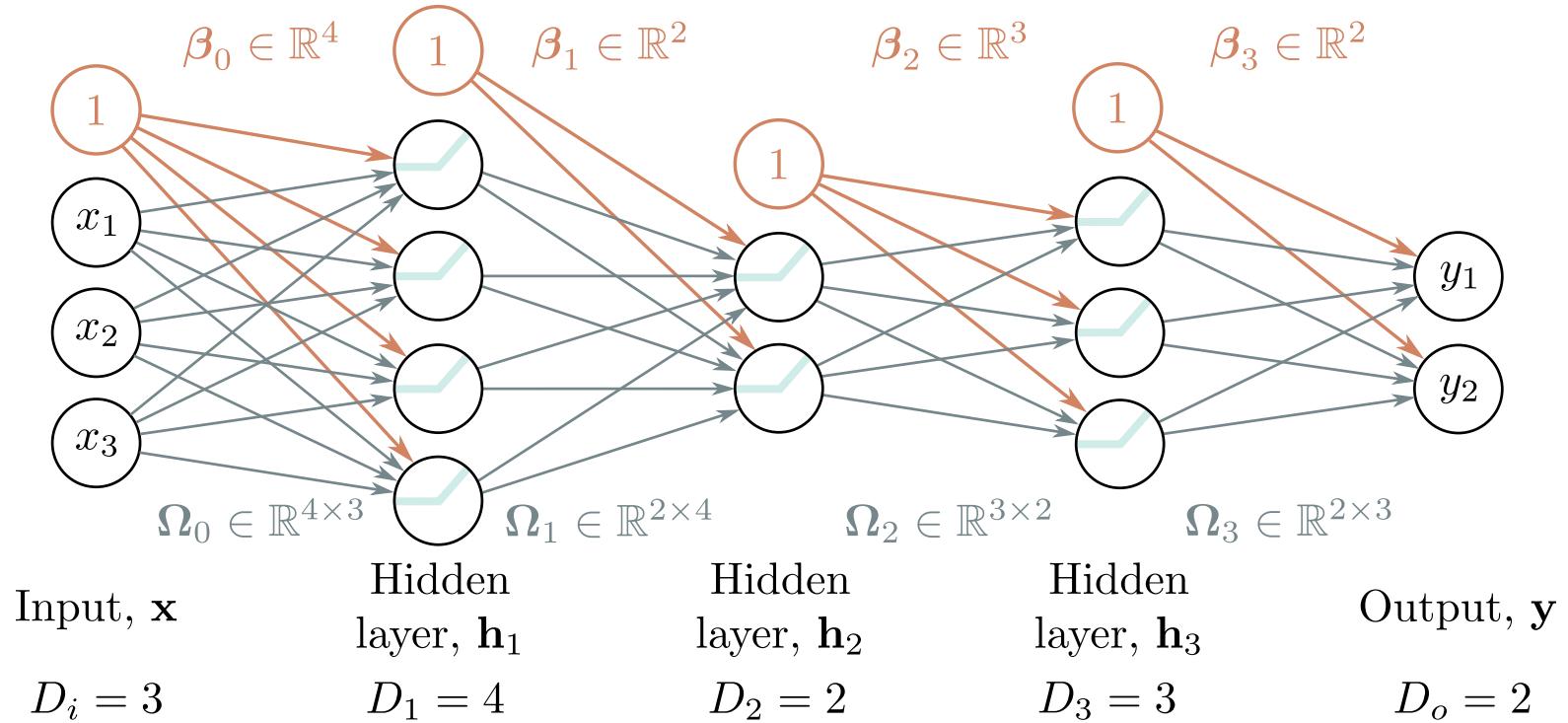
$$L[\phi, f[\mathbf{x}_i, \phi], \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^I]$$

or for short:

$$L[\phi]$$

>Returns a scalar that is smaller when model maps inputs to outputs better

Example



$$\mathbf{h}_1 = \mathbf{a}[\beta_0 + \Omega_0 \mathbf{x}]$$

$$\mathbf{h}_2 = \mathbf{a}[\beta_1 + \Omega_1 \mathbf{h}_1]$$

$$\mathbf{h}_3 = \mathbf{a}[\beta_2 + \Omega_2 \mathbf{h}_2]$$

$$\mathbf{f}[\mathbf{x}, \phi] = \beta_3 + \Omega_3 \mathbf{h}_3$$

Problem 1: Computing gradients

Loss: sum of individual terms:

$$L[\phi] = \sum_{i=1}^I \ell_i = \sum_{i=1}^I l[f(\mathbf{x}_i, \phi), y_i]$$

SGD Algorithm:

$$\phi_{t+1} \leftarrow \phi_t - \alpha \sum_{i \in \mathcal{B}_t} \frac{\partial \ell_i[\phi_t]}{\partial \phi}$$

Parameters:

$$\phi = \{\beta_0, \Omega_0, \beta_1, \Omega_1, \beta_2, \Omega_2, \beta_3, \Omega_3\}$$

Need to compute gradients

$$\frac{\partial \ell_i}{\partial \beta_k} \quad \text{and} \quad \frac{\partial \ell_i}{\partial \Omega_k}$$

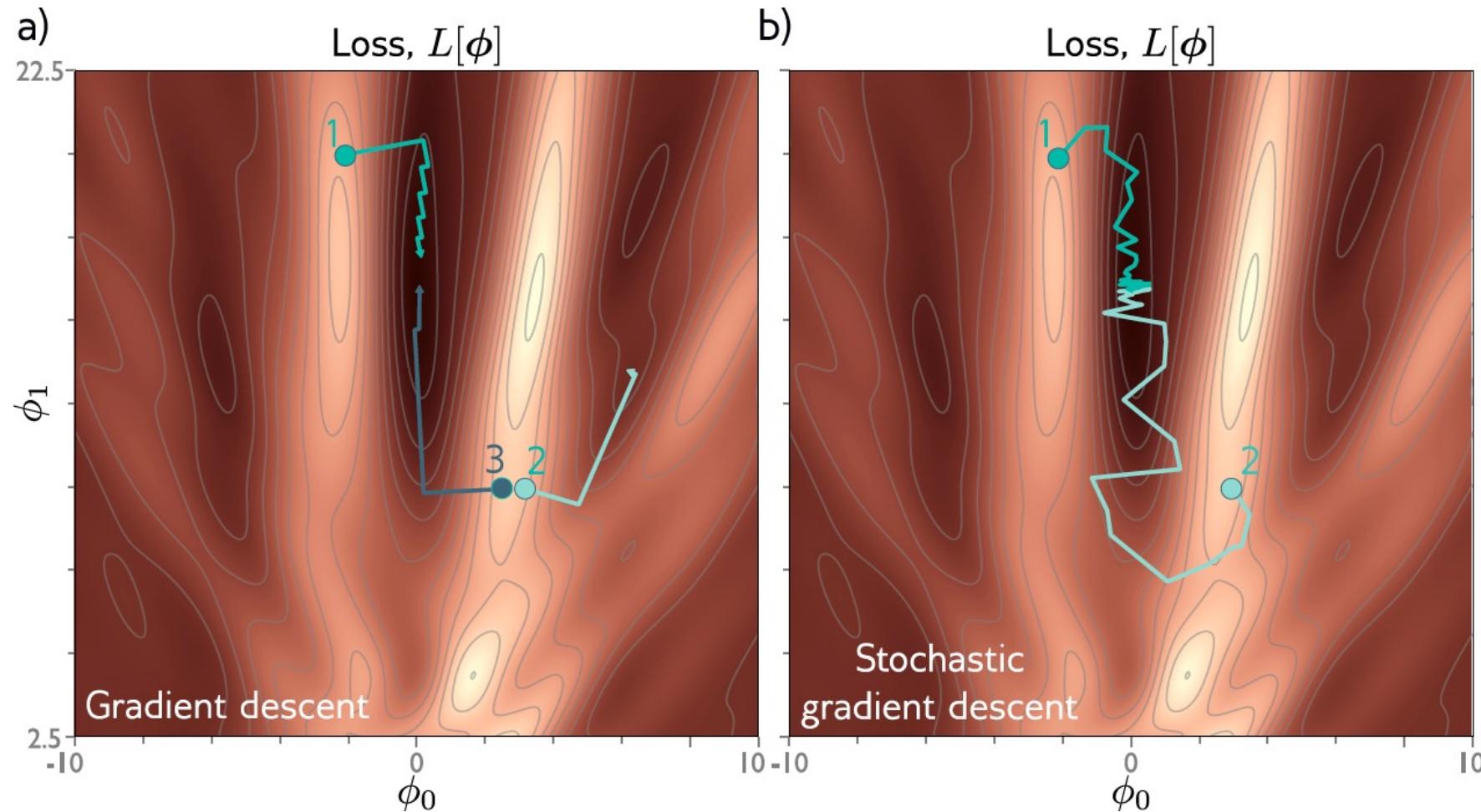
Why is this such a big deal?

- A neural network is just an equation:

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

- But it's a huge equation, and we need to compute derivative
 - for every parameter
 - for every point in the batch
 - for every iteration of SGD

Problem 2: initialization



Where should we start the parameters before we commence SGD?

Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- Code

Problem 1: Computing gradients

Loss: sum of individual terms:

$$L[\phi] = \sum_{i=1}^I \ell_i = \sum_{i=1}^I l[f(\mathbf{x}_i, \phi), y_i]$$

SGD Algorithm:

$$\phi_{t+1} \leftarrow \phi_t - \alpha \sum_{i \in \mathcal{B}_t} \frac{\partial \ell_i[\phi_t]}{\partial \phi}$$

Parameters:

$$\phi = \{\beta_0, \Omega_0, \beta_1, \Omega_1, \beta_2, \Omega_2, \beta_3, \Omega_3\}$$

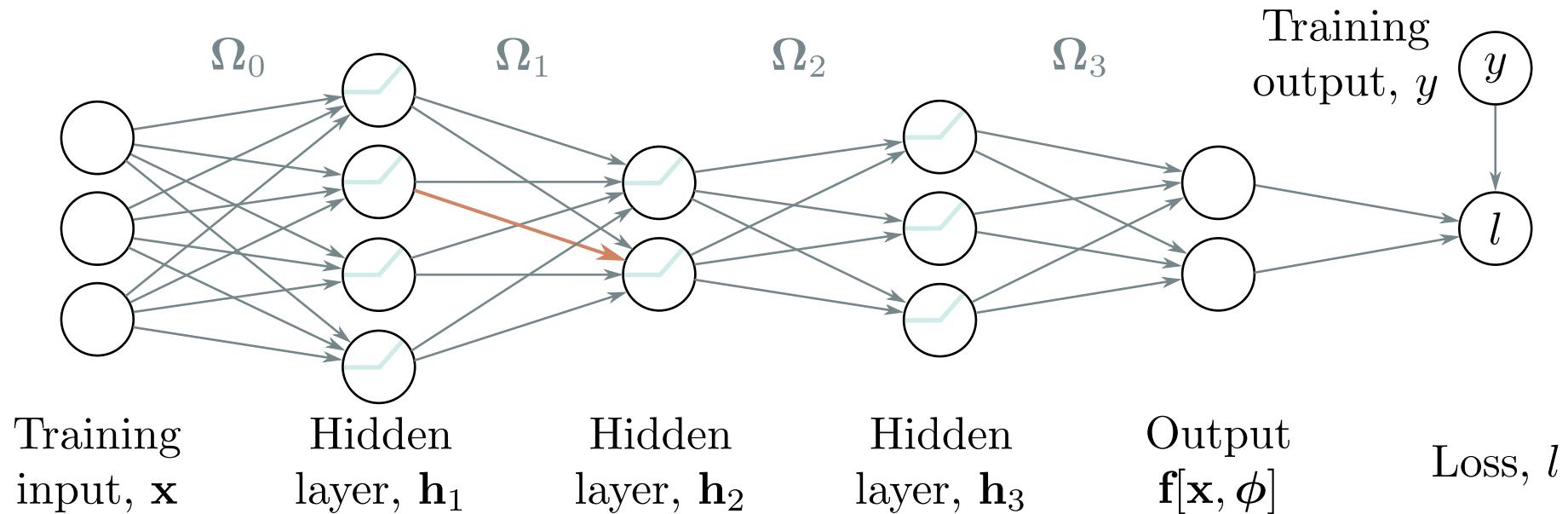
Need to compute gradients

$$\frac{\partial \ell_i}{\partial \beta_k} \quad \text{and} \quad \frac{\partial \ell_i}{\partial \Omega_k}$$

Algorithm to compute gradient efficiently

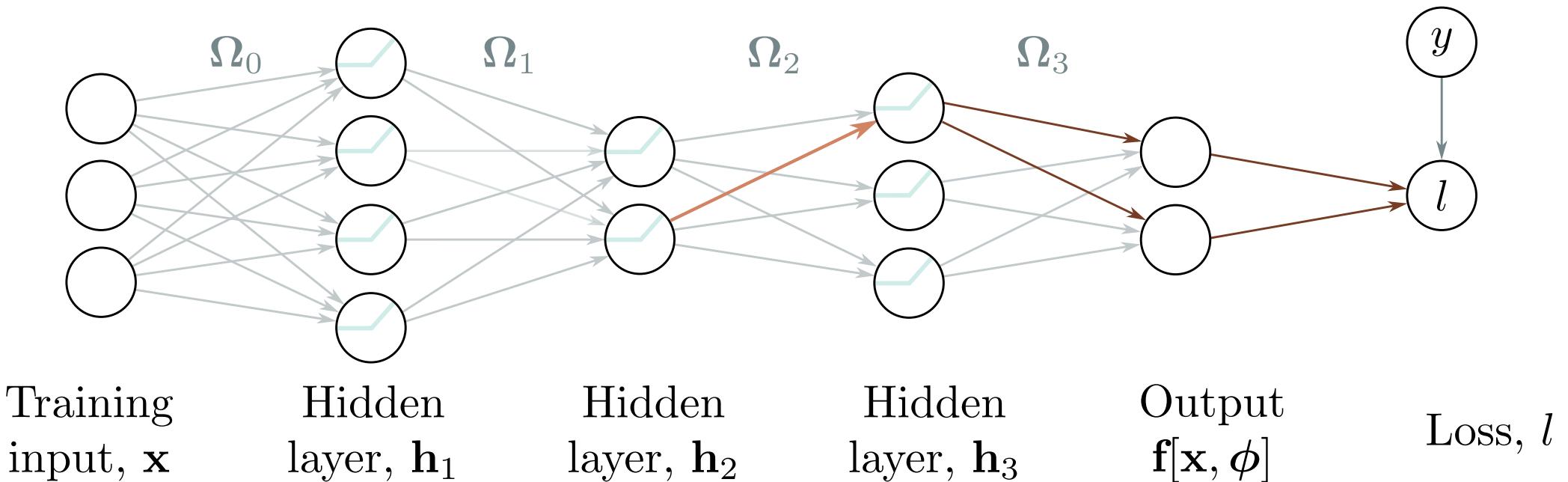
- “Backpropagation algorithm”
- Rumelhart, Hinton, and Williams (1986)

BackProp intuition #1: the forward pass



- Orange weight multiplies activation (ReLU output) in previous layer
- We want to know how change in orange weight affects loss
- If we double activation in previous layer, weight will have twice the effect
- Conclusion: **we need to know the activations at each layer.**

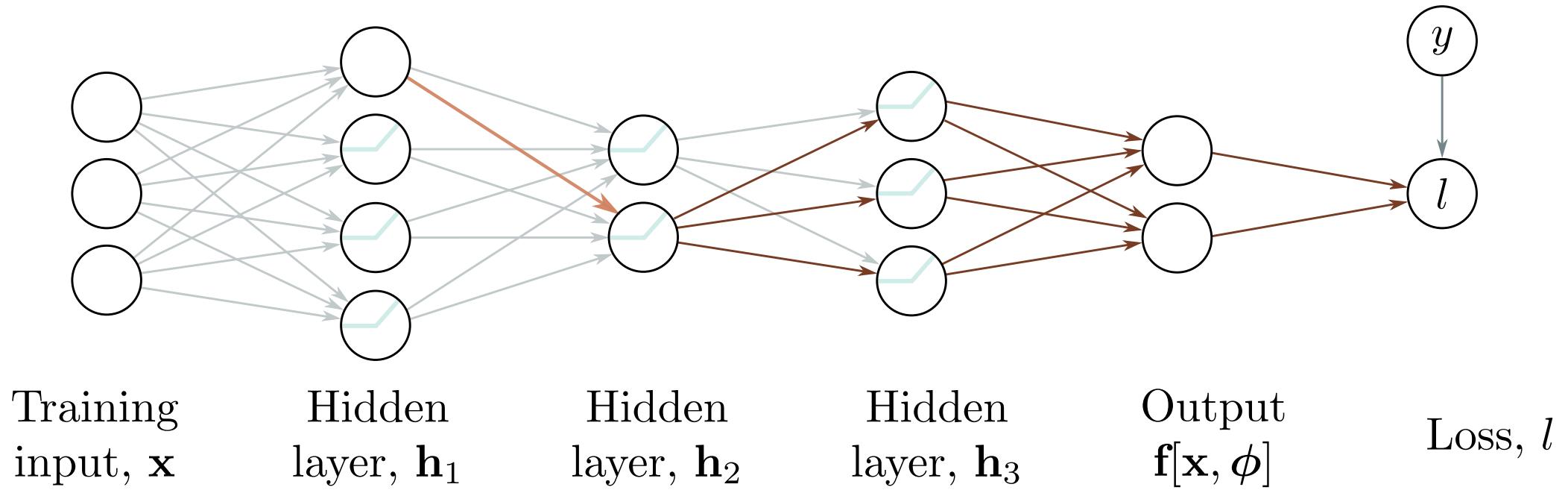
BackProp intuition #2: the backward pass



To calculate how a small change in a weight or bias feeding into hidden layer \mathbf{h}_3 modifies the loss, we need to know:

- how a change in layer \mathbf{h}_3 changes the model output \mathbf{f}
- how a change in model output changes the loss l

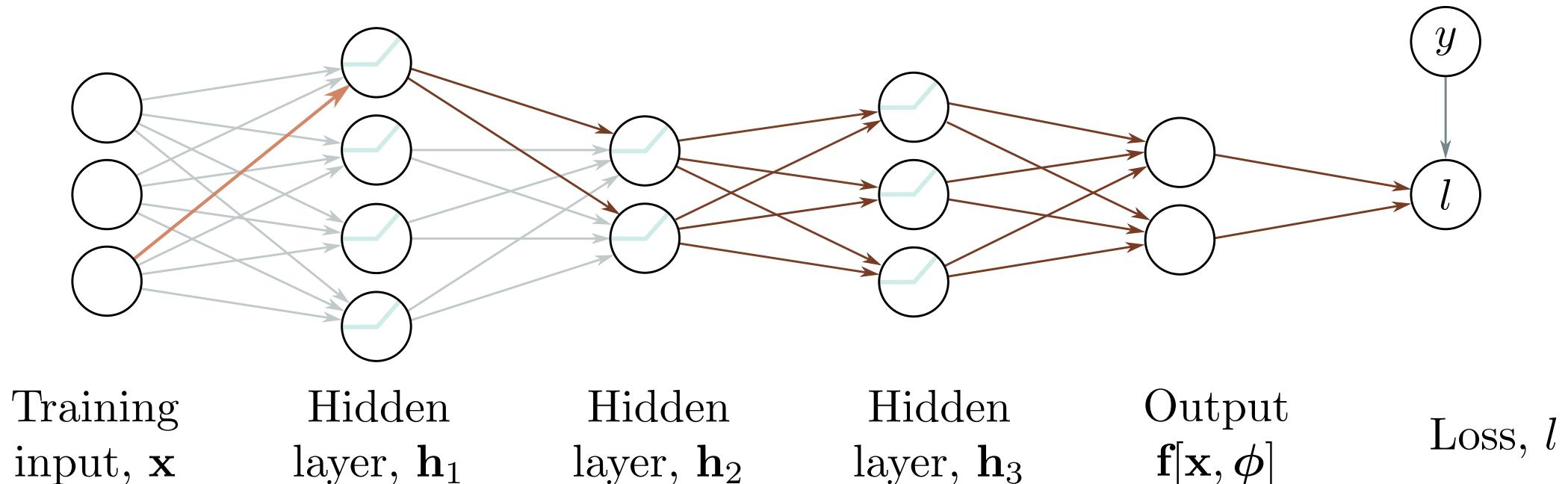
BackProp intuition #2: the backward pass



To calculate how a small change in a weight or bias feeding into hidden layer \mathbf{h}_2 modifies the loss, we need to know:

- how a change in layer \mathbf{h}_2 affects \mathbf{h}_3
- how \mathbf{h}_3 changes the model output
- how this output changes the loss

BackProp intuition #2: the backward pass



To calculate how a small change in a weight or bias feeding into hidden layer \mathbf{h}_1 modifies the loss, we need to know:

- how a change in layer \mathbf{h}_1 affects layer \mathbf{h}_2
- how a change in layer \mathbf{h}_2 affects layer \mathbf{h}_3
- how layer \mathbf{h}_3 changes the model output
- how the model output changes the loss

Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- Code

Toy function

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos\left[\beta_2 + \omega_2 \cdot \exp\left[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]\right]\right]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

- Consists of a series of functions that are composed with each other.
- Unlike in neural networks just uses scalars (not vectors)
- “Activation functions” sin, exp, cos

Toy function

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos\left[\beta_2 + \omega_2 \cdot \exp\left[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]\right]\right]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

Derivatives

$$\frac{\partial \cos[z]}{\partial z} = -\sin[z] \quad \frac{\partial \exp[z]}{\partial z} = \exp[z] \quad \frac{\partial \sin[z]}{\partial z} = \cos[z]$$

Gradients of toy function

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]]]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

We want to calculate:

$$\frac{\partial \ell_i}{\partial \beta_0}, \quad \frac{\partial \ell_i}{\partial \omega_0}, \quad \frac{\partial \ell_i}{\partial \beta_1}, \quad \frac{\partial \ell_i}{\partial \omega_1}, \quad \frac{\partial \ell_i}{\partial \beta_2}, \quad \frac{\partial \ell_i}{\partial \omega_2}, \quad \frac{\partial \ell_i}{\partial \beta_3}, \quad \text{and} \quad \frac{\partial \ell_i}{\partial \omega_3}$$

How does a small change in β_3 change the loss ℓ_i for the i 'th example?

Gradients of composed functions

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]]]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

Calculating expressions by hand:

- some expressions very complicated.
- obvious redundancy (look at sin terms in bottom equation)

$$\begin{aligned} \frac{\partial \ell_i}{\partial \omega_0} = & -2 \left(\beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x_i]]] - y_i \right) \\ & \cdot \omega_1 \omega_2 \omega_3 \cdot x_i \cdot \cos[\beta_0 + \omega_0 \cdot x_i] \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x_i]] \\ & \cdot \sin[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x_i]]] \end{aligned}$$

Forward pass

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]]]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

1. Write this as a series of intermediate calculations
2. Compute these intermediate quantities

Forward pass

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos \left[\beta_2 + \omega_2 \cdot \exp \left[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x] \right] \right]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

1. Write this as a series of intermediate calculations

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_1 = \sin[f_0]$$

$$h_3 = \cos[f_2]$$

2. Compute these intermediate quantities

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$h_2 = \exp[f_1]$$

$$\ell_i = (f_3 - y_i)^2.$$

Forward pass

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]]]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

1. Write this as a series of intermediate calculations

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_1 = \sin[f_0]$$

$$h_3 = \cos[f_2]$$

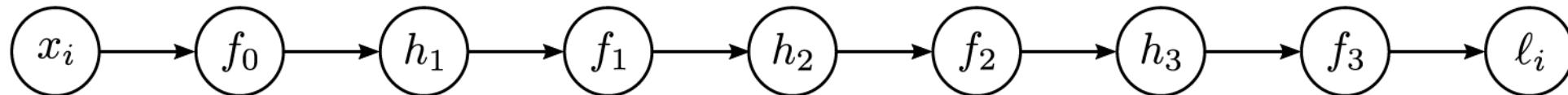
2. Compute these intermediate quantities

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$h_2 = \exp[f_1]$$

$$\ell_i = (f_3 - y_i)^2.$$



Backward pass

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]]]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\frac{\partial \ell_i}{\partial f_3}, \quad \frac{\partial \ell_i}{\partial h_3}, \quad \frac{\partial \ell_i}{\partial f_2}, \quad \frac{\partial \ell_i}{\partial h_2}, \quad \frac{\partial \ell_i}{\partial f_1}, \quad \frac{\partial \ell_i}{\partial h_1}, \quad \text{and} \quad \frac{\partial \ell_i}{\partial f_0}$$

Backward pass

$$f[x, \phi] = \beta_3 + \omega_3 \cdot \cos[\beta_2 + \omega_2 \cdot \exp[\beta_1 + \omega_1 \cdot \sin[\beta_0 + \omega_0 \cdot x]]]$$

$$\ell_i = (f[x_i, \phi] - y_i)^2$$

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\frac{\partial \ell_i}{\partial f_3}, \quad \frac{\partial \ell_i}{\partial h_3}, \quad \frac{\partial \ell_i}{\partial f_2}, \quad \frac{\partial \ell_i}{\partial h_2}, \quad \frac{\partial \ell_i}{\partial f_1}, \quad \frac{\partial \ell_i}{\partial h_1}, \quad \text{and} \quad \frac{\partial \ell_i}{\partial f_0}$$



Backward pass

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{array}{ll} f_0 = \beta_0 + \omega_0 \cdot x_i & f_2 = \beta_2 + \omega_2 \cdot h_2 \\ h_1 = \sin[f_0] & h_3 = \cos[f_2] \\ f_1 = \beta_1 + \omega_1 \cdot h_1 & f_3 = \beta_3 + \omega_3 \cdot h_3 \\ h_2 = \exp[f_1] & \ell_i = (f_3 - y_i)^2. \end{array}$$

- The first of these derivatives is trivial

$$\frac{\partial \ell_i}{\partial f_3} = 2(f_3 - y_i)$$

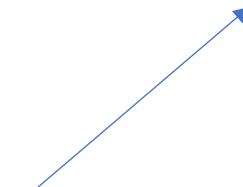
Backward pass

- 1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{aligned}f_0 &= \beta_0 + \omega_0 \cdot x_i & f_2 &= \beta_2 + \omega_2 \cdot h_2 \\h_1 &= \sin[f_0] & h_3 &= \cos[f_2] \\f_1 &= \beta_1 + \omega_1 \cdot h_1 & f_3 &= \beta_3 + \omega_3 \cdot h_3 \\h_2 &= \exp[f_1] & \ell_i &= (f_3 - y_i)^2.\end{aligned}$$

- The second of these derivatives is computed via the chain rule

$$\frac{\partial \ell_i}{\partial h_3} = \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3}$$



How does a small change in h_3 change ℓ_i ?

Backward pass

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{aligned}f_0 &= \beta_0 + \omega_0 \cdot x_i & f_2 &= \beta_2 + \omega_2 \cdot h_2 \\h_1 &= \sin[f_0] & h_3 &= \cos[f_2] \\f_1 &= \beta_1 + \omega_1 \cdot h_1 & f_3 &= \beta_3 + \omega_3 \cdot h_3 \\h_2 &= \exp[f_1] & \ell_i &= (f_3 - y_i)^2.\end{aligned}$$

- The second derivative is computed via the chain rule

$$\frac{\partial \ell_i}{\partial h_3} = \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3}$$

How does a small change in h_3 change ℓ_i ?

How does a small change in h_3 change f_3 ?

How does a small change in f_3 change ℓ_i ?

Backward pass

- 1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$h_1 = \sin[f_0]$$

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

$$h_2 = \exp[f_1]$$

$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_3 = \cos[f_2]$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$\ell_i = (f_3 - y_i)^2.$$

- The second of these derivatives is computed via the chain rule

$$\frac{\partial \ell_i}{\partial h_3} = \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3}$$

How does a small change in h_3 change ℓ_i ?

ω_3

Already computed!

Backward pass

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{aligned} f_0 &= \beta_0 + \omega_0 \cdot x_i & f_2 &= \beta_2 + \omega_2 \cdot h_2 \\ h_1 &= \sin[f_0] & h_3 &= \cos[f_2] \\ f_1 &= \beta_1 + \omega_1 \cdot h_1 & f_3 &= \beta_3 + \omega_3 \cdot h_3 \\ h_2 &= \exp[f_1] & \ell_i &= (f_3 - y_i)^2. \end{aligned}$$

- The remaining derivatives also calculated by further use of chain rule

$$\frac{\partial \ell_i}{\partial f_2} = \frac{\partial h_3}{\partial f_2} \left(\frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right)$$

Backward pass

- 1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{array}{ll} f_0 = \beta_0 + \omega_0 \cdot x_i & f_2 = \beta_2 + \omega_2 \cdot h_2 \\ h_1 = \sin[f_0] & h_3 = \cos[f_2] \\ f_1 = \beta_1 + \omega_1 \cdot h_1 & f_3 = \beta_3 + \omega_3 \cdot h_3 \\ h_2 = \exp[f_1] & \ell_i = (f_3 - y_i)^2. \end{array}$$

- The remaining derivatives also calculated by further use of chain rule

$$\frac{\partial \ell_i}{\partial f_2} = \frac{\partial h_3}{\partial f_2} \left(\frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right)$$

- $\sin[f_2]$

Already computed!

Backward pass

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{aligned} f_0 &= \beta_0 + \omega_0 \cdot x_i & f_2 &= \beta_2 + \omega_2 \cdot h_2 \\ h_1 &= \sin[f_0] & h_3 &= \cos[f_2] \\ f_1 &= \beta_1 + \omega_1 \cdot h_1 & f_3 &= \beta_3 + \omega_3 \cdot h_3 \\ h_2 &= \exp[f_1] & \ell_i &= (f_3 - y_i)^2. \end{aligned}$$

- The remaining derivatives also calculated by further use of chain rule

$$\begin{aligned} \frac{\partial \ell_i}{\partial f_2} &= \frac{\partial h_3}{\partial f_2} \left(\frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \\ \frac{\partial \ell_i}{\partial h_2} &= \frac{\partial f_2}{\partial h_2} \left(\frac{\partial h_3}{\partial f_2} \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \end{aligned}$$

Backward pass

- 1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$\begin{array}{ll} f_0 = \beta_0 + \omega_0 \cdot x_i & f_2 = \beta_2 + \omega_2 \cdot h_2 \\ h_1 = \sin[f_0] & h_3 = \cos[f_2] \\ f_1 = \beta_1 + \omega_1 \cdot h_1 & f_3 = \beta_3 + \omega_3 \cdot h_3 \\ h_2 = \exp[f_1] & \ell_i = (f_3 - y_i)^2. \end{array}$$

- The remaining derivatives also calculated by further use of chain rule

$$\begin{aligned} \frac{\partial \ell_i}{\partial f_2} &= \frac{\partial h_3}{\partial f_2} \left(\frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \\ \frac{\partial \ell_i}{\partial h_2} &= \frac{\partial f_2}{\partial h_2} \left(\frac{\partial h_3}{\partial f_2} \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \\ \frac{\partial \ell_i}{\partial f_1} &= \frac{\partial h_2}{\partial f_1} \left(\frac{\partial f_2}{\partial h_2} \frac{\partial h_3}{\partial f_2} \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \\ \frac{\partial \ell_i}{\partial h_1} &= \frac{\partial f_1}{\partial h_1} \left(\frac{\partial h_2}{\partial f_1} \frac{\partial f_2}{\partial h_2} \frac{\partial h_3}{\partial f_2} \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \\ \frac{\partial \ell_i}{\partial f_0} &= \frac{\partial h_1}{\partial f_0} \left(\frac{\partial f_1}{\partial h_1} \frac{\partial h_2}{\partial f_1} \frac{\partial f_2}{\partial h_2} \frac{\partial h_3}{\partial f_2} \frac{\partial f_3}{\partial h_3} \frac{\partial \ell_i}{\partial f_3} \right) \end{aligned}$$

Backward pass

1. Compute the derivatives of the loss with respect to these intermediate quantities, but in reverse order.

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$h_1 = \sin[f_0]$$

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

$$h_2 = \exp[f_1]$$

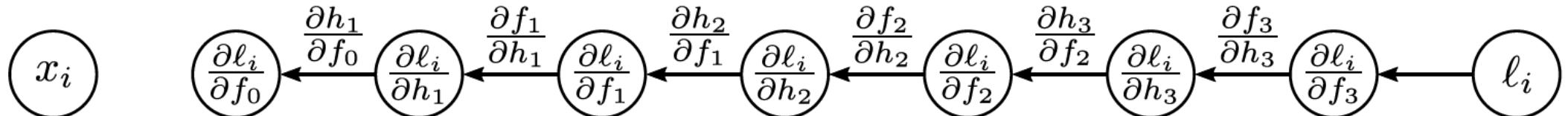
$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_3 = \cos[f_2]$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$\ell_i = (f_3 - y_i)^2.$$

- The remaining derivatives also calculated by further use of chain rule



Backward pass

- 2. Find how the loss changes as a function of the parameters β and ω .

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$h_1 = \sin[f_0]$$

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

$$h_2 = \exp[f_1]$$

$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_3 = \cos[f_2]$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$\ell_i = (f_3 - y_i)^2.$$

- Another application of the chain rule

$$\frac{\partial \ell_i}{\partial \omega_k} = \frac{\partial f_k}{\partial \omega_k} \frac{\partial \ell_i}{\partial f_k}$$

How does a small change in ω_k change ℓ_i ?

How does a small change in ω_k change f_k ?

How does a small change in f_k change ℓ_i ?

Backward pass

- 2. Find how the loss changes as a function of the parameters β and ω .

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$h_1 = \sin[f_0]$$

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

$$h_2 = \exp[f_1]$$

$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_3 = \cos[f_2]$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$\ell_i = (f_3 - y_i)^2.$$

- Another application of the chain rule

$$\frac{\partial \ell_i}{\partial \omega_k} = \frac{\partial f_k}{\partial \omega_k} \frac{\partial \ell_i}{\partial f_k}$$

How does a small change in ω_k change ℓ_i ?

h_k

Already calculated in part 1.

Backward pass

2. Find how the loss changes as a function of the parameters β and ω .

$$\begin{array}{ll} f_0 = \beta_0 + \omega_0 \cdot x_i & f_2 = \beta_2 + \omega_2 \cdot h_2 \\ h_1 = \sin[f_0] & h_3 = \cos[f_2] \\ f_1 = \beta_1 + \omega_1 \cdot h_1 & f_3 = \beta_3 + \omega_3 \cdot h_3 \\ h_2 = \exp[f_1] & \ell_i = (f_3 - y_i)^2. \end{array}$$

- Another application of the chain rule
- Similarly for β parameters

$$\frac{\partial \ell_i}{\partial \omega_k} = \frac{\partial f_k}{\partial \omega_k} \frac{\partial \ell_i}{\partial f_k}$$

$$\frac{\partial \ell_i}{\partial \beta_k} = \frac{\partial f_k}{\partial \beta_k} \frac{\partial \ell_i}{\partial f_k}$$

Backward pass

2. Find how the loss changes as a function of the parameters β and ω .

$$f_0 = \beta_0 + \omega_0 \cdot x_i$$

$$h_1 = \sin[f_0]$$

$$f_1 = \beta_1 + \omega_1 \cdot h_1$$

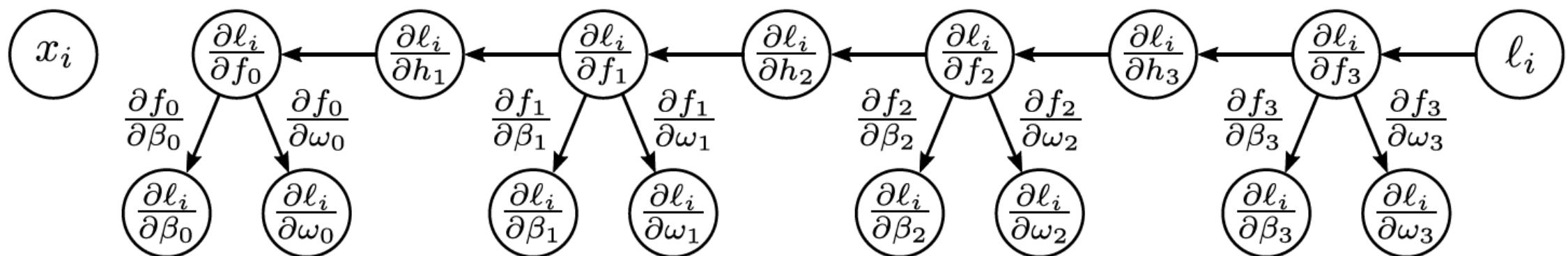
$$h_2 = \exp[f_1]$$

$$f_2 = \beta_2 + \omega_2 \cdot h_2$$

$$h_3 = \cos[f_2]$$

$$f_3 = \beta_3 + \omega_3 \cdot h_3$$

$$\ell_i = (f_3 - y_i)^2.$$



Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- Code

Matrix calculus

Scalar function $f[]$ of a vector \mathbf{a}

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

$$\frac{\partial f}{\partial \mathbf{a}} = \begin{bmatrix} \frac{\partial f}{\partial a_1} \\ \frac{\partial f}{\partial a_2} \\ \frac{\partial f}{\partial a_3} \\ \frac{\partial f}{\partial a_4} \end{bmatrix}$$

Matrix calculus

Scalar function $f()$ of a matrix \mathbf{A}

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\frac{\partial f}{\partial \mathbf{A}} = \begin{bmatrix} \frac{\partial f}{\partial a_{11}} & \frac{\partial f}{\partial a_{12}} & \frac{\partial f}{\partial a_{13}} \\ \frac{\partial f}{\partial a_{21}} & \frac{\partial f}{\partial a_{22}} & \frac{\partial f}{\partial a_{23}} \\ \frac{\partial f}{\partial a_{31}} & \frac{\partial f}{\partial a_{32}} & \frac{\partial f}{\partial a_{33}} \\ \frac{\partial f}{\partial a_{41}} & \frac{\partial f}{\partial a_{42}} & \frac{\partial f}{\partial a_{43}} \end{bmatrix}$$

Matrix calculus

Vector function $\mathbf{f}[]$ of vector \mathbf{a}

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \quad \frac{\partial \mathbf{f}}{\partial \mathbf{a}} = \begin{bmatrix} \frac{\partial f_1}{\partial a_1} & \frac{\partial f_2}{\partial a_1} & \frac{\partial f_3}{\partial a_1} \\ \frac{\partial f_1}{\partial a_2} & \frac{\partial f_2}{\partial a_2} & \frac{\partial f_3}{\partial a_2} \\ \frac{\partial f_1}{\partial a_3} & \frac{\partial f_2}{\partial a_3} & \frac{\partial f_3}{\partial a_3} \\ \frac{\partial f_1}{\partial a_4} & \frac{\partial f_2}{\partial a_4} & \frac{\partial f_4}{\partial a_4} \end{bmatrix}$$

Comparing vector and matrix

Scalar derivatives:

$$f_3 = \beta_3 + \omega_3 h_3$$

$$\frac{\partial f_3}{\partial h_3} = \frac{\partial}{\partial h_3} (\beta_3 + \omega_3 h_3) = \omega_3$$

Comparing vector and matrix

Scalar derivatives:

$$f_3 = \beta_3 + \omega_3 h_3$$

$$\frac{\partial f_3}{\partial h_3} = \frac{\partial}{\partial h_3} (\beta_3 + \omega_3 h_3) = \omega_3$$

Matrix derivatives:

$$\mathbf{f}_3 = \boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3$$

$$\frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} = \frac{\partial}{\partial \mathbf{h}_3} (\boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3) = \boldsymbol{\Omega}_3^T$$

Comparing vector and matrix

Scalar derivatives:

$$f_3 = \beta_3 + \omega_3 h_3$$

$$\frac{\partial f_3}{\partial \beta_3} = \frac{\partial}{\partial \omega_3} \beta_3 + \omega_3 h_3 = 1$$

Matrix derivatives:

$$\mathbf{f}_3 = \boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3$$

$$\frac{\partial \mathbf{f}_3}{\partial \boldsymbol{\beta}_3} = \frac{\partial}{\partial \boldsymbol{\beta}_3} (\boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3) = \mathbf{I}$$

Homework: (keeners only)

- Consider function:

$$\mathbf{f} = \mathbf{B}\mathbf{a}$$

- Can write as:

$$f_i = \sum_j B_{ij} a_j$$

- Now calculate:

$$\frac{\partial \mathbf{f}}{\partial \mathbf{a}} = \begin{bmatrix} \frac{\partial f_1}{\partial a_1} & \frac{\partial f_2}{\partial a_1} & \frac{\partial f_3}{\partial a_1} \\ \frac{\partial f_1}{\partial a_2} & \frac{\partial f_2}{\partial a_2} & \frac{\partial f_3}{\partial a_2} \\ \frac{\partial f_1}{\partial a_3} & \frac{\partial f_2}{\partial a_3} & \frac{\partial f_3}{\partial a_3} \\ \frac{\partial f_1}{\partial a_4} & \frac{\partial f_2}{\partial a_4} & \frac{\partial f_3}{\partial a_4} \end{bmatrix}$$

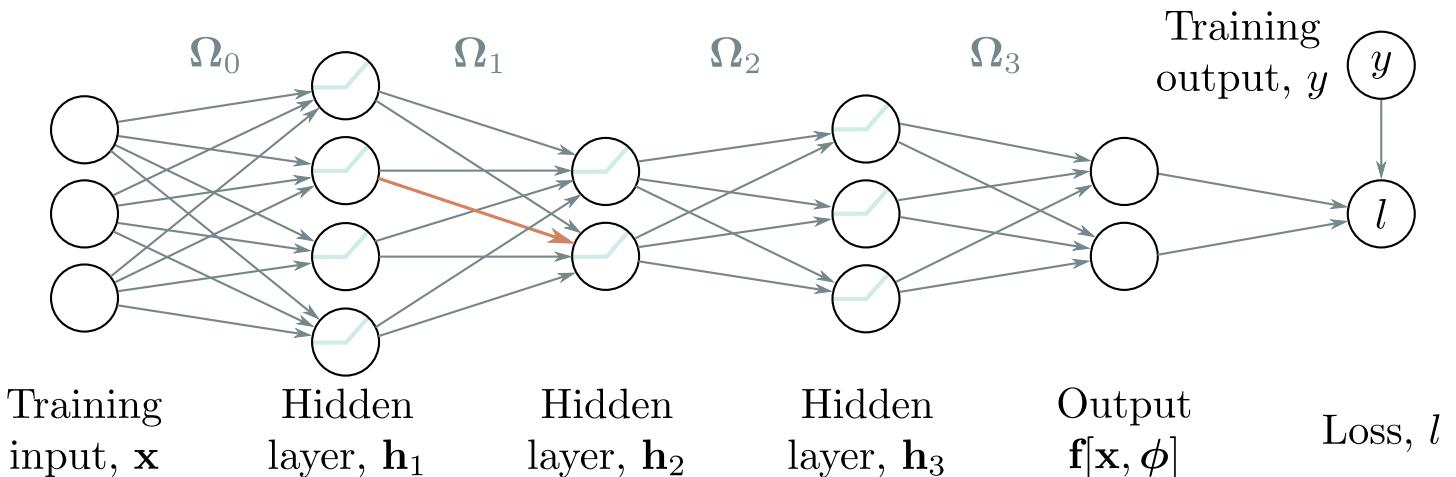
- Write final expression as a matrix

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- Code

The forward pass



1. Write this as a series of intermediate calculations

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

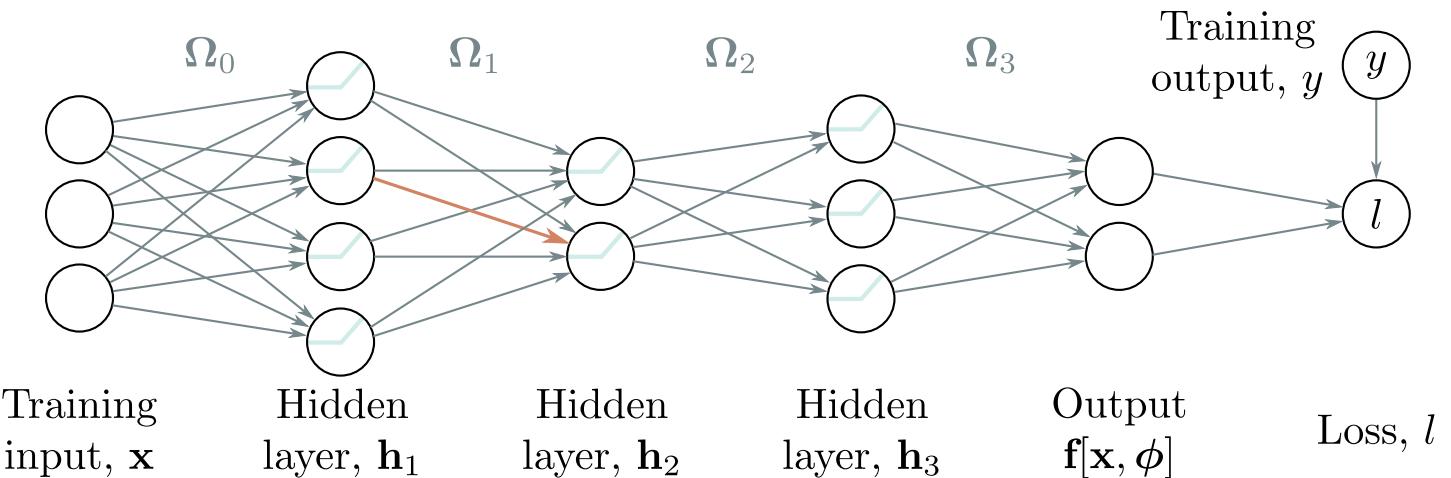
$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

$$\ell_i = l[\mathbf{f}_3, y_i]$$

The forward pass



1. Write this as a series of intermediate calculations

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

2. Compute these intermediate quantities

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

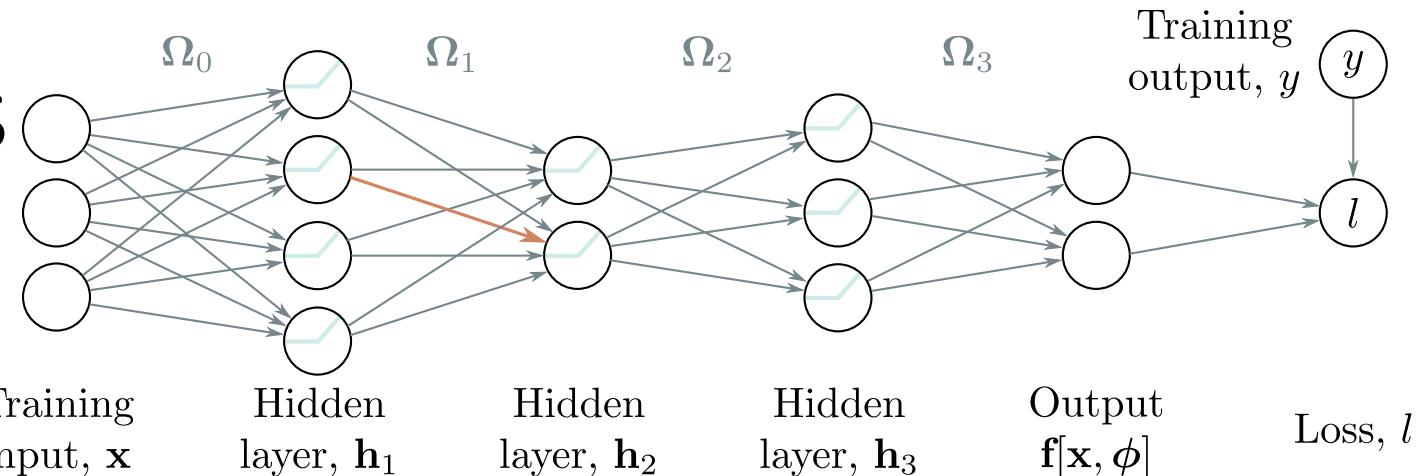
$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

$$\ell_i = l[\mathbf{f}_3, y_i]$$

The backward pass



1. Write this as a series of intermediate calculations

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

2. Compute these intermediate quantities

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_2} = \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_1} = \frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \left(\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

3. Take derivatives of output with respect to intermediate quantities

$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

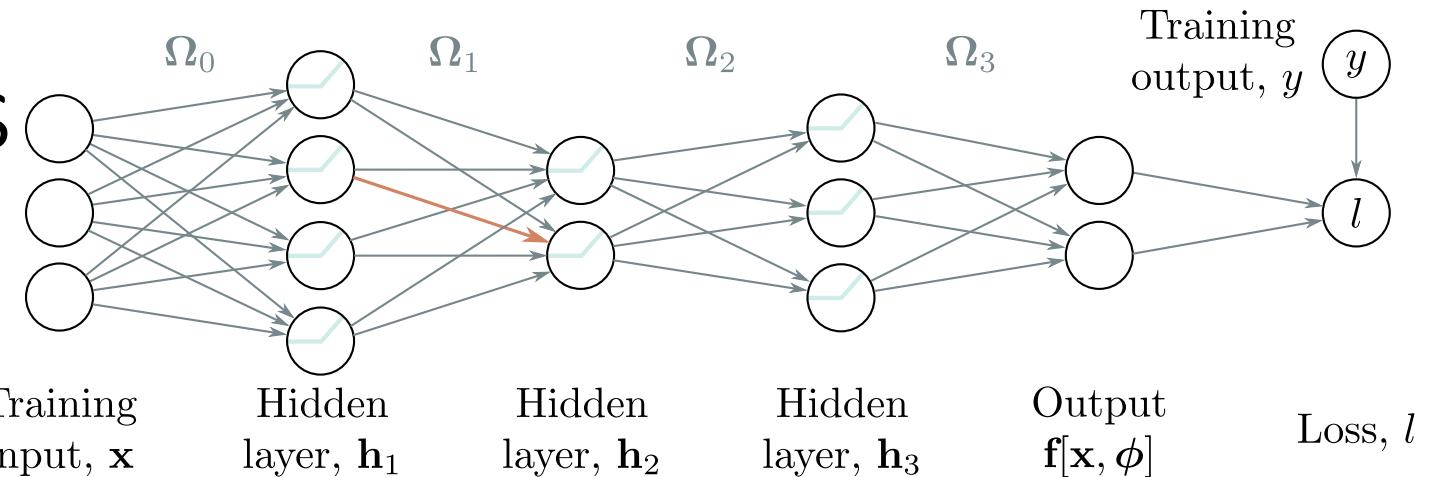
$$\ell_i = l[\mathbf{f}_3, y_i]$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_0} = \frac{\partial \mathbf{h}_1}{\partial \mathbf{f}_0} \frac{\partial \mathbf{f}_1}{\partial \mathbf{h}_1} \left(\frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- Code

The backward pass



1. Write this as a series of intermediate calculations
2. Compute these intermediate quantities
3. Take derivatives of output with respect to intermediate quantities

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

$$\ell_i = l[\mathbf{f}_3, y_i]$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_2} = \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \boxed{\frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3}} \frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_1} = \frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \left(\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_0} = \frac{\partial \mathbf{h}_1}{\partial \mathbf{f}_0} \frac{\partial \mathbf{f}_1}{\partial \mathbf{h}_1} \left(\frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

Yikes!

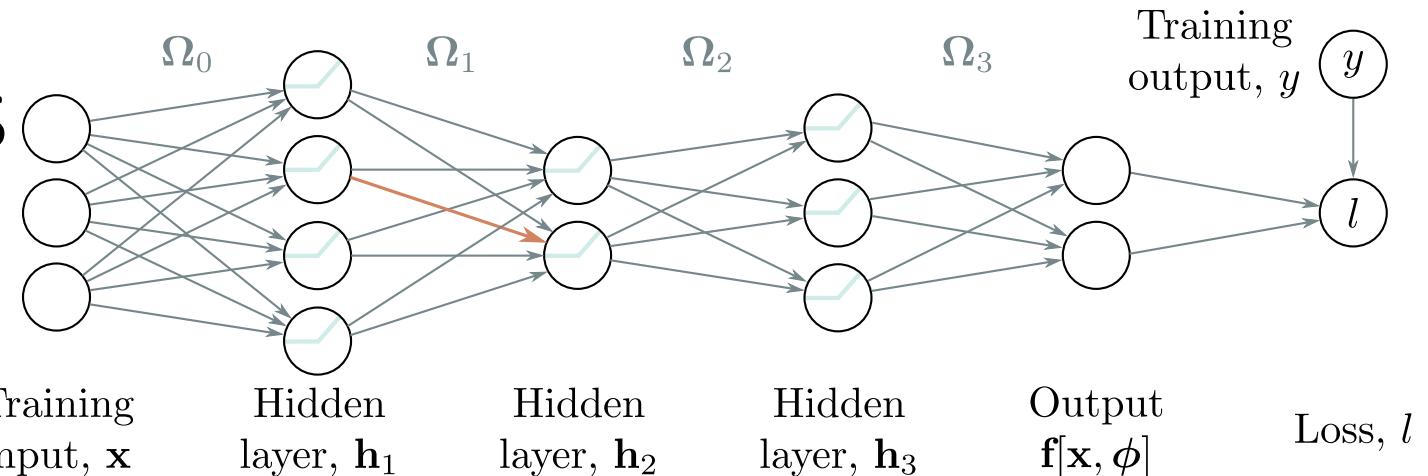
- But:

$$\frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} = \frac{\partial}{\partial \mathbf{h}_3} (\boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3) = \boldsymbol{\Omega}_3^T$$

- Quite similar to:

$$\frac{\partial f_3}{\partial h_3} = \frac{\partial}{\partial h_3} (\beta_3 + \omega_3 h_3) = \omega_3$$

The backward pass



1. Write this as a series of intermediate calculations

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

2. Compute these intermediate quantities

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

3. Take derivatives of output with respect to intermediate quantities

$$\ell_i = \mathbf{l}[\mathbf{f}_3, y_i]$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

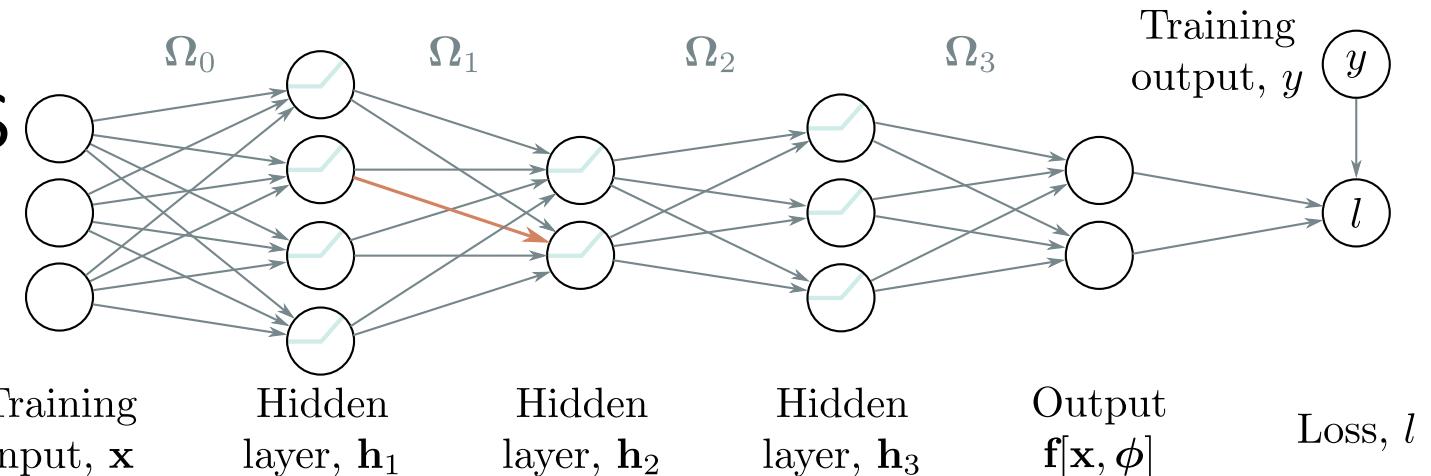
$$\boxed{\frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} = \frac{\partial}{\partial \mathbf{h}_3} (\beta_3 + \Omega_3 \mathbf{h}_3) = \Omega_3^T}$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_2} = \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \boxed{\frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3}} \frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_1} = \frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \left(\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_0} = \frac{\partial \mathbf{h}_1}{\partial \mathbf{f}_0} \frac{\partial \mathbf{f}_1}{\partial \mathbf{h}_1} \left(\frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

The backward pass



1. Write this as a series of intermediate calculations

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

2. Compute these intermediate quantities

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

3. Take derivatives of output with respect to intermediate quantities

$$\ell_i = l[\mathbf{f}_3, y_i]$$

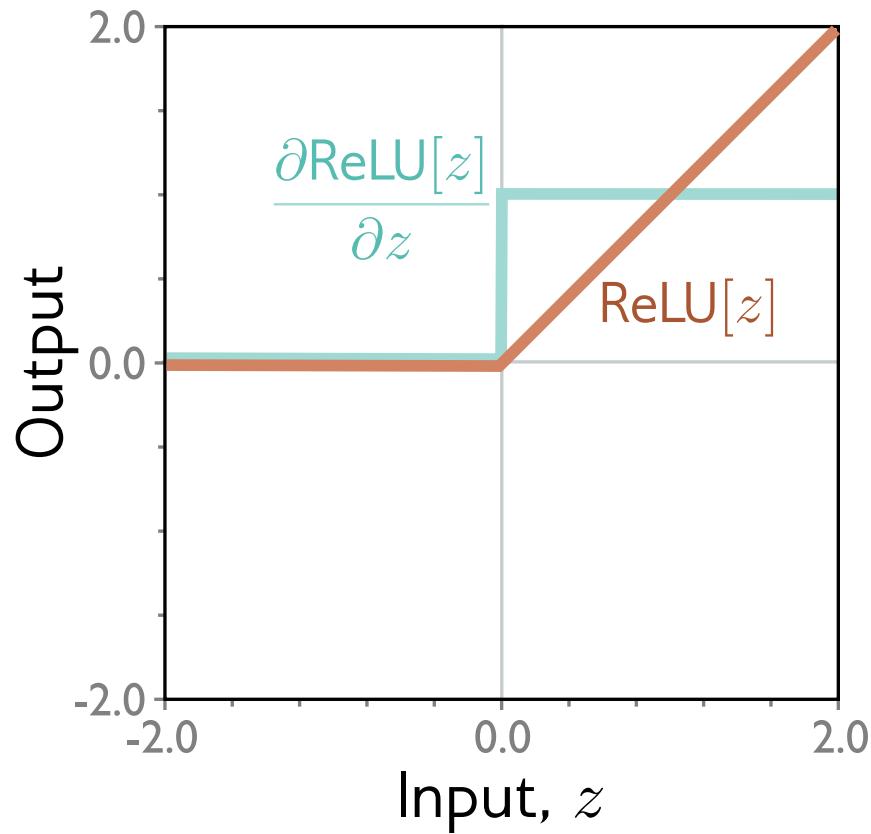
$$\frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_2} = \boxed{\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2}} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

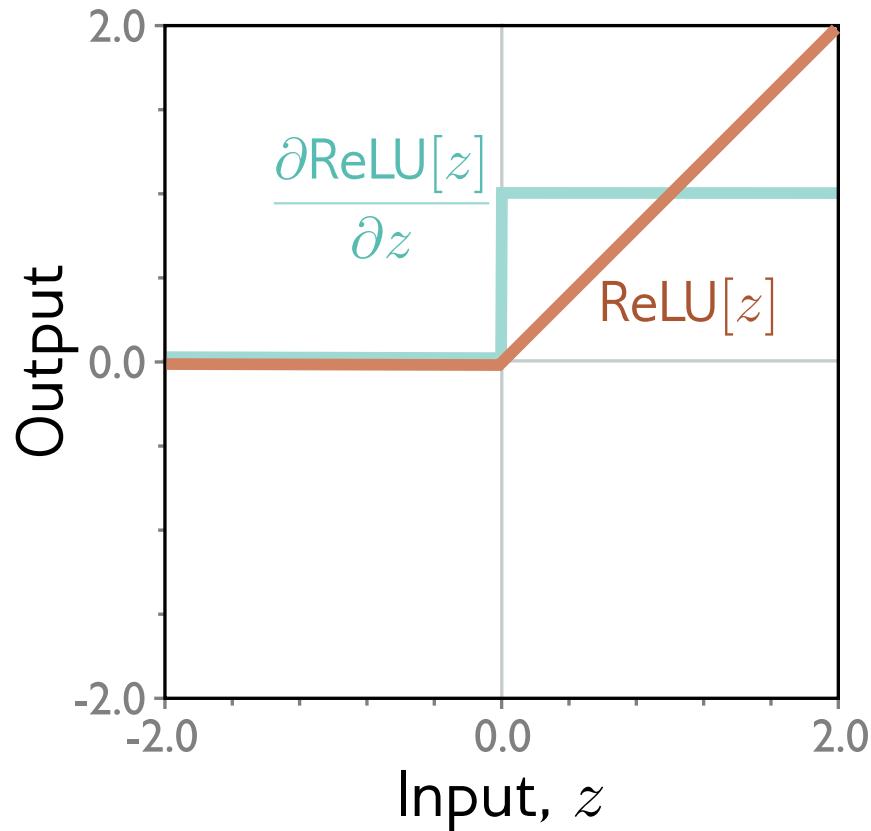
$$\frac{\partial \ell_i}{\partial \mathbf{f}_1} = \frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \left(\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_0} = \frac{\partial \mathbf{h}_1}{\partial \mathbf{f}_0} \frac{\partial \mathbf{f}_1}{\partial \mathbf{h}_1} \left(\frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

Derivative of ReLU



Derivative of ReLU



$$\mathbb{I}[z > 0]$$

“Indicator function”

Derivative of ReLU

1. Consider:

$$\mathbf{a} = \text{ReLU}[\mathbf{b}]$$

where:

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

2. We could equivalently write:

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \text{ReLU}[b_1] \\ \text{ReLU}[b_2] \\ \text{ReLU}[b_3] \end{bmatrix}$$

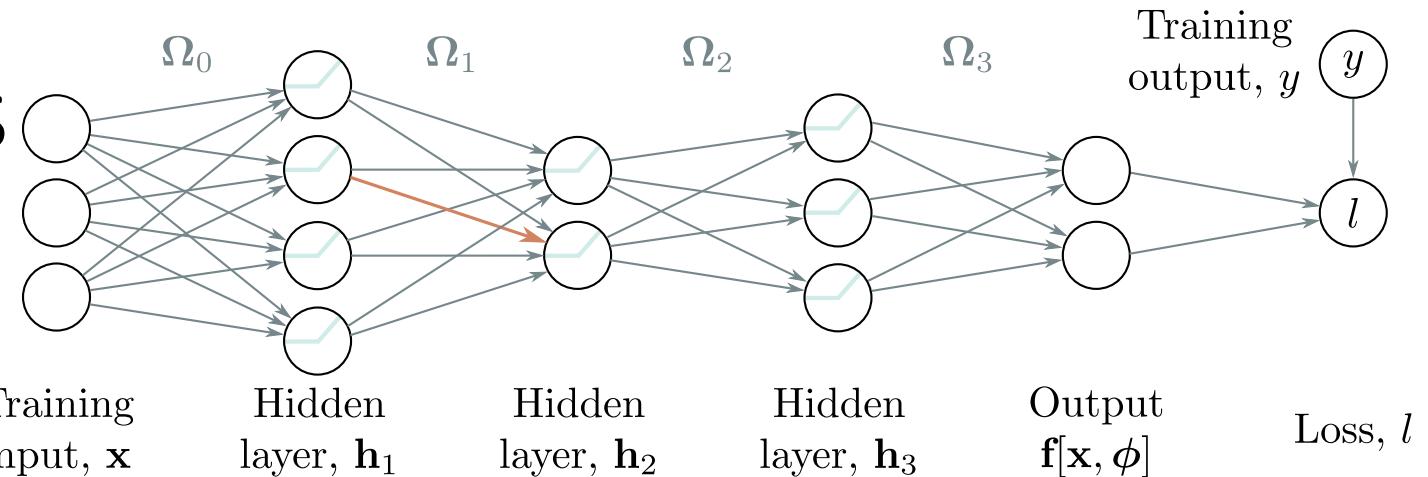
3. Taking the derivative

$$\frac{\partial \mathbf{a}}{\partial \mathbf{b}} = \begin{bmatrix} \frac{\partial a_1}{\partial b_1} & \frac{\partial a_2}{\partial b_1} & \frac{\partial a_3}{\partial b_1} \\ \frac{\partial a_1}{\partial b_2} & \frac{\partial a_2}{\partial b_2} & \frac{\partial a_3}{\partial b_2} \\ \frac{\partial a_1}{\partial b_3} & \frac{\partial a_2}{\partial b_3} & \frac{\partial a_3}{\partial b_3} \end{bmatrix} = \begin{bmatrix} \mathbb{I}[b_1 > 0] & 0 & 0 \\ 0 & \mathbb{I}[b_2 > 0] & 0 \\ 0 & 0 & \mathbb{I}[b_3 > 0] \end{bmatrix}$$

4. We can equivalently pointwise multiply by diagonal

$$\mathbb{I}[\mathbf{b} > 0] \odot$$

The backward pass



1. Write this as a series of intermediate calculations

$$\mathbf{f}_0 = \beta_0 + \Omega_0 \mathbf{x}_i$$

2. Compute these intermediate quantities

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \beta_1 + \Omega_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\mathbf{f}_2 = \beta_2 + \Omega_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \beta_3 + \Omega_3 \mathbf{h}_3$$

3. Take derivatives of output with respect to intermediate quantities

$$\ell_i = \mathbb{I}[\mathbf{f}_3, y_i]$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

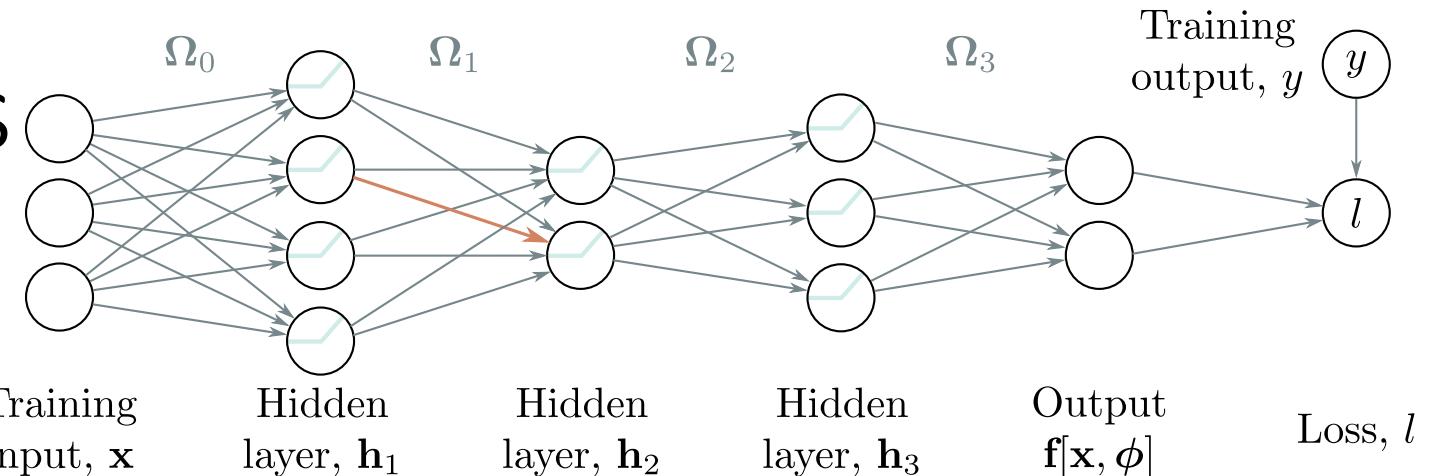
$$\frac{\partial \ell_i}{\partial \mathbf{f}_2} = \boxed{\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2}} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3}$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_1} = \frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \left(\frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

$$\frac{\partial \ell_i}{\partial \mathbf{f}_0} = \frac{\partial \mathbf{h}_1}{\partial \mathbf{f}_0} \frac{\partial \mathbf{f}_1}{\partial \mathbf{h}_1} \left(\frac{\partial \mathbf{h}_2}{\partial \mathbf{f}_1} \frac{\partial \mathbf{f}_2}{\partial \mathbf{h}_2} \frac{\partial \mathbf{h}_3}{\partial \mathbf{f}_2} \frac{\partial \mathbf{f}_3}{\partial \mathbf{h}_3} \frac{\partial \ell_i}{\partial \mathbf{f}_3} \right)$$

$\mathbb{I}[\mathbf{f}_2 > 0]$

The backward pass



1. Write this as a series of intermediate calculations
2. Compute these intermediate quantities
3. Take derivatives of output with respect to intermediate quantities
4. Take derivatives w.r.t. parameters

$$\mathbf{f}_0 = \boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}_i$$

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \boldsymbol{\beta}_1 + \boldsymbol{\Omega}_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\mathbf{f}_2 = \boldsymbol{\beta}_2 + \boldsymbol{\Omega}_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3$$

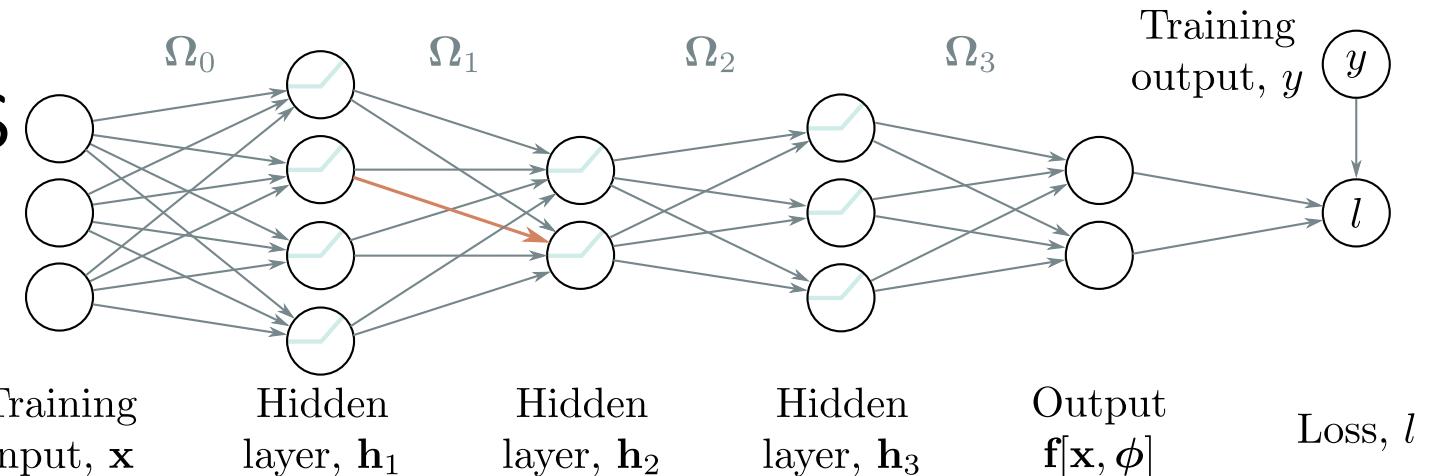
$$\ell_i = l[\mathbf{f}_3, y_i]$$

$$\frac{\partial \ell_i}{\partial \boldsymbol{\beta}_k} = \frac{\partial \mathbf{f}_k}{\partial \boldsymbol{\beta}_k} \frac{\partial \ell_i}{\partial \mathbf{f}_k}$$

$$= \frac{\partial}{\partial \boldsymbol{\beta}_k} (\boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k) \frac{\partial \ell_i}{\partial \mathbf{f}_k}$$

$$= \frac{\partial \ell_i}{\partial \mathbf{f}_k},$$

The backward pass



1. Write this as a series of intermediate calculations
2. Compute these intermediate quantities
3. Take derivatives of output with respect to intermediate quantities
4. Take derivatives w.r.t. parameters

$$\mathbf{f}_0 = \boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}_i$$

$$\mathbf{h}_1 = \mathbf{a}[\mathbf{f}_0]$$

$$\mathbf{f}_1 = \boldsymbol{\beta}_1 + \boldsymbol{\Omega}_1 \mathbf{h}_1$$

$$\mathbf{h}_2 = \mathbf{a}[\mathbf{f}_1]$$

$$\mathbf{f}_2 = \boldsymbol{\beta}_2 + \boldsymbol{\Omega}_2 \mathbf{h}_2$$

$$\mathbf{h}_3 = \mathbf{a}[\mathbf{f}_2]$$

$$\mathbf{f}_3 = \boldsymbol{\beta}_3 + \boldsymbol{\Omega}_3 \mathbf{h}_3$$

$$\ell_i = l[\mathbf{f}_3, y_i]$$

$$\frac{\partial \ell_i}{\partial \boldsymbol{\Omega}_k} = \frac{\partial \mathbf{f}_k}{\partial \boldsymbol{\Omega}_k} \frac{\partial \ell_i}{\partial \mathbf{f}_k}$$

$$= \frac{\partial}{\partial \boldsymbol{\Omega}_k} (\boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k) \frac{\partial \ell_i}{\partial \mathbf{f}_k}$$

$$= \frac{\partial \ell_i}{\partial \mathbf{f}_k} \mathbf{h}_k^T$$

Backprop summary

Forward pass: We compute and store the following quantities:

$$\begin{aligned}\mathbf{f}_0 &= \boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}_i \\ \mathbf{h}_k &= \mathbf{a}[\mathbf{f}_{k-1}] & k \in \{1, 2, \dots, K\} \\ \mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k. & k \in \{1, 2, \dots, K\}\end{aligned}$$

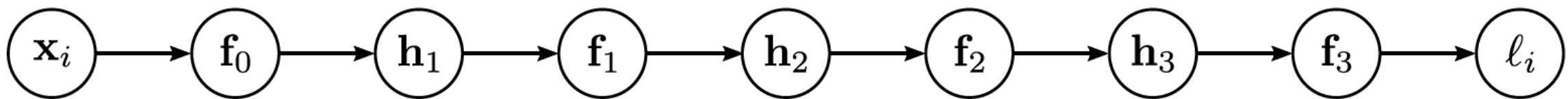
Backprop summary

Forward pass: We compute and store the following quantities:

$$\mathbf{f}_0 = \boldsymbol{\beta}_0 + \boldsymbol{\Omega}_0 \mathbf{x}_i$$

$$\mathbf{h}_k = \mathbf{a}[\mathbf{f}_{k-1}] \quad k \in \{1, 2, \dots, K\}$$

$$\mathbf{f}_k = \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k. \quad k \in \{1, 2, \dots, K\}$$



Backprop summary

Backward pass: We start with the derivative $\partial\ell_i/\partial\mathbf{f}_K$ of the loss function ℓ_i with respect to the network output \mathbf{f}_K and work backward through the network:

$$\begin{aligned}\frac{\partial\ell_i}{\partial\boldsymbol{\beta}_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} && k \in \{K, K-1, \dots, 1\} \\ \frac{\partial\ell_i}{\partial\boldsymbol{\Omega}_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} \mathbf{h}_k^T && k \in \{K, K-1, \dots, 1\} \\ \frac{\partial\ell_i}{\partial\mathbf{f}_{k-1}} &= \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial\ell_i}{\partial\mathbf{f}_k} \right), && k \in \{K, K-1, \dots, 1\}\end{aligned}\tag{7.13}$$

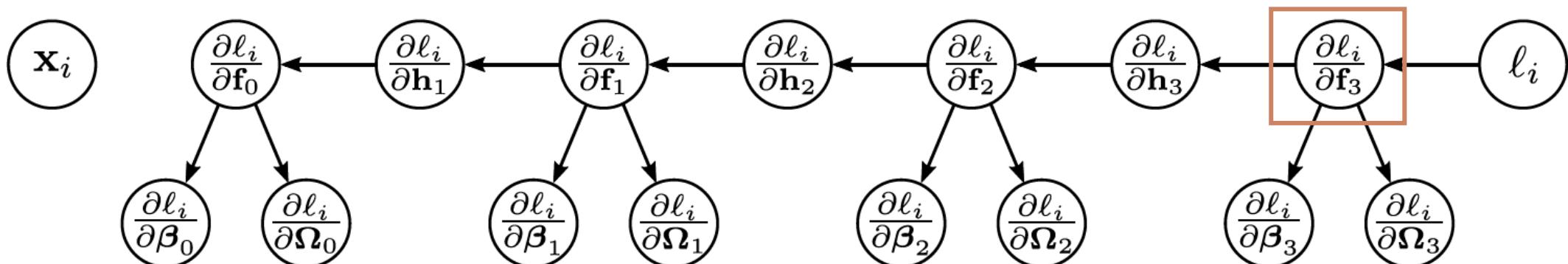
where \odot denotes pointwise multiplication and $\mathbb{I}[\mathbf{f}_{k-1} > 0]$ is a vector containing ones where \mathbf{f}_{k-1} is greater than zero and zeros elsewhere.

Backprop summary

Backward pass: We start with the derivative $\frac{\partial \ell_i}{\partial \mathbf{f}_K}$ of the loss function ℓ_i with respect to the network output \mathbf{f}_K and work backward through the network:

$$\begin{aligned}
 \frac{\partial \ell_i}{\partial \boldsymbol{\beta}_k} &= \frac{\partial \ell_i}{\partial \mathbf{f}_k} & k \in \{K, K-1, \dots, 1\} \\
 \frac{\partial \ell_i}{\partial \boldsymbol{\Omega}_k} &= \frac{\partial \ell_i}{\partial \mathbf{f}_k} \mathbf{h}_k^T & k \in \{K, K-1, \dots, 1\} \\
 \frac{\partial \ell_i}{\partial \mathbf{f}_{k-1}} &= \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial \ell_i}{\partial \mathbf{f}_k} \right), & k \in \{K, K-1, \dots, 1\}
 \end{aligned} \tag{7.13}$$

where \odot denotes pointwise multiplication and $\mathbb{I}[\mathbf{f}_{k-1} > 0]$ is a vector containing ones where \mathbf{f}_{k-1} is greater than zero and zeros elsewhere.

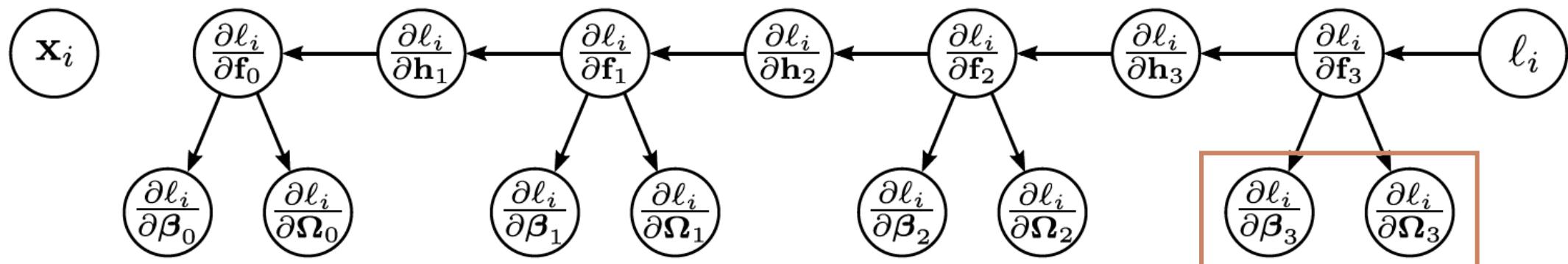


Backprop summary

Backward pass: We start with the derivative $\partial\ell_i/\partial\mathbf{f}_K$ of the loss function ℓ_i with respect to the network output \mathbf{f}_K and work backward through the network:

$$\begin{aligned}
 \frac{\partial\ell_i}{\partial\beta_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} & k \in \{K, K-1, \dots, 1\} \\
 \frac{\partial\ell_i}{\partial\Omega_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} \mathbf{h}_k^T & k \in \{K, K-1, \dots, 1\} \\
 \frac{\partial\ell_i}{\partial\mathbf{f}_{k-1}} &= \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial\ell_i}{\partial\mathbf{f}_k} \right), & k \in \{K, K-1, \dots, 1\} \tag{7.13}
 \end{aligned}$$

where \odot denotes pointwise multiplication and $\mathbb{I}[\mathbf{f}_{k-1} > 0]$ is a vector containing ones where \mathbf{f}_{k-1} is greater than zero and zeros elsewhere.



Backprop summary

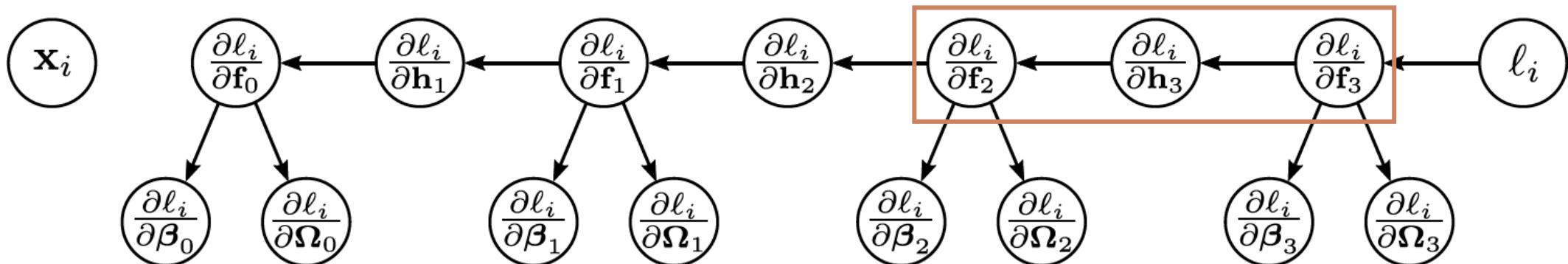
Backward pass: We start with the derivative $\partial\ell_i/\partial\mathbf{f}_K$ of the loss function ℓ_i with respect to the network output \mathbf{f}_K and work backward through the network:

$$\frac{\partial\ell_i}{\partial\beta_k} = \frac{\partial\ell_i}{\partial\mathbf{f}_k} \quad k \in \{K, K-1, \dots, 1\}$$

$$\frac{\partial\ell_i}{\partial\Omega_k} = \frac{\partial\ell_i}{\partial\mathbf{f}_k} \mathbf{h}_k^T \quad k \in \{K, K-1, \dots, 1\}$$

$$\boxed{\frac{\partial\ell_i}{\partial\mathbf{f}_{k-1}} = \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial\ell_i}{\partial\mathbf{f}_k} \right)}, \quad k \in \{K, K-1, \dots, 1\} \quad (7.13)$$

where \odot denotes pointwise multiplication and $\mathbb{I}[\mathbf{f}_{k-1} > 0]$ is a vector containing ones where \mathbf{f}_{k-1} is greater than zero and zeros elsewhere.



Backprop summary

Backward pass: We start with the derivative $\partial\ell_i/\partial\mathbf{f}_K$ of the loss function ℓ_i with respect to the network output \mathbf{f}_K and work backward through the network:

$$\begin{aligned}\frac{\partial\ell_i}{\partial\boldsymbol{\beta}_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} & k \in \{K, K-1, \dots, 1\} \\ \frac{\partial\ell_i}{\partial\boldsymbol{\Omega}_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} \mathbf{h}_k^T & k \in \{K, K-1, \dots, 1\} \\ \frac{\partial\ell_i}{\partial\mathbf{f}_{k-1}} &= \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial\ell_i}{\partial\mathbf{f}_k} \right), & k \in \{K, K-1, \dots, 1\}\end{aligned}\tag{7.13}$$

where \odot denotes pointwise multiplication and $\mathbb{I}[\mathbf{f}_{k-1} > 0]$ is a vector containing ones where \mathbf{f}_{k-1} is greater than zero and zeros elsewhere. Finally, we compute the derivatives with respect to the first set of biases and weights:

$$\begin{aligned}\frac{\partial\ell_i}{\partial\boldsymbol{\beta}_0} &= \frac{\partial\ell_i}{\partial\mathbf{f}_0} \\ \frac{\partial\ell_i}{\partial\boldsymbol{\Omega}_0} &= \frac{\partial\ell_i}{\partial\mathbf{f}_0} \mathbf{x}_i^T\end{aligned}$$

Pros and cons

- Extremely efficient
 - Only need matrix multiplication and thresholding for ReLU functions
- Memory hungry – must store all the intermediate quantities
- Sequential
 - can process multiple batches in parallel
 - but things get harder if the whole model doesn't fit on one machine.

Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- Code

Algorithmic differentiation

- Modern deep learning frameworks compute derivatives automatically
- You just have to specify the model and the loss
- How? Algorithmic differentiation
 - Each component knows how to compute its own derivative
 - ReLU knows how to compute deriv of output w.r.t. input
 - Linear function knows how to compute deriv of output w.r.t. input
 - Linear function knows how to compute deriv of output w.r.t. parameter
 - You specify how the order of the components
 - It can compute the chain of derivatives
- Works with branches as long as it's still an acyclic graph

Gradients

- Backpropagation intuition
- Toy model
- Background mathematics
- Backpropagation forward pass
- Backpropagation backward pass
- Algorithmic differentiation
- **Code**

PyTorch code

- Define a neural network
- Initialize params with He initialization
- Define loss function
- Choose optimization algorithm
- Choose initial learning rate
- Choose learning rates schedule
- Make some random data
- Train for 100 batches

```
import torch, torch.nn as nn
from torch.utils.data import TensorDataset, DataLoader
from torch.optim.lr_scheduler import StepLR

# define input size, hidden layer size, output size
D_i, D_k, D_o = 10, 40, 5
# create model with two hidden layers
model = nn.Sequential(
    nn.Linear(D_i, D_k),
    nn.ReLU(),
    nn.Linear(D_k, D_k),
    nn.ReLU(),
    nn.Linear(D_k, D_o))

# He initialization of weights
def weights_init(layer_in):
    if isinstance(layer_in, nn.Linear):
        nn.init.kaiming_uniform_(layer_in.weight)
        layer_in.bias.data.fill_(0.0)
model.apply(weights_init)

# choose least squares loss function
criterion = nn.MSELoss()
# construct SGD optimizer and initialize learning rate and momentum
optimizer = torch.optim.SGD(model.parameters(), lr = 0.01, momentum=0.9)
# object that decreases learning rate by half every 10 epochs
scheduler = StepLR(optimizer, step_size=10, gamma=0.5)

# create 100 dummy data points and store in data loader class
x = torch.randn(100, D_i)
y = torch.randn(100, D_o)
data_loader = DataLoader(TensorDataset(x,y), batch_size=10, shuffle=True)

# loop over the dataset 100 times
for epoch in range(100):
    epoch_loss = 0.0
    # loop over batches
    for i, data in enumerate(data_loader):
        # retrieve inputs and labels for this batch
        x_batch, y_batch = data
        # zero the parameter gradients
        optimizer.zero_grad()
        # forward pass
        pred = model(x_batch)
        loss = criterion(pred, y_batch)
        # backward pass
        loss.backward()
        # SGD update
        optimizer.step()
        # update statistics
        epoch_loss += loss.item()
    # print error
    print(f'Epoch {epoch:5d}, loss {epoch_loss:.3f}')
    # tell scheduler to consider updating learning rate
    scheduler.step()
```

PyTorch code

- Define a neural network
- Initialize params with He initialization
- Define loss function
- Choose optimization algorithm
- Choose initial learning rate
- Choose learning rates schedule
- Make some random data
- Train for 100 batches

```
import torch, torch.nn as nn
from torch.utils.data import TensorDataset, DataLoader
from torch.optim.lr_scheduler import StepLR

# define input size, hidden layer size, output size
D_i, D_k, D_o = 10, 40, 5
# create model with two hidden layers
model = nn.Sequential(
    nn.Linear(D_i, D_k),
    nn.ReLU(),
    nn.Linear(D_k, D_k),
    nn.ReLU(),
    nn.Linear(D_k, D_o))

# He initialization of weights
def weights_init(layer_in):
    if isinstance(layer_in, nn.Linear):
        nn.init.kaiming_uniform_(layer_in.weight)
        layer_in.bias.data.fill_(0.0)
model.apply(weights_init)

# choose least squares loss function
criterion = nn.MSELoss()
# construct SGD optimizer and initialize learning rate and momentum
optimizer = torch.optim.SGD(model.parameters(), lr = 0.01, momentum=0.9)
# object that decreases learning rate by half every 10 epochs
scheduler = StepLR(optimizer, step_size=10, gamma=0.5)

# create 100 dummy data points and store in data loader class
x = torch.randn(100, D_i)
y = torch.randn(100, D_o)
data_loader = DataLoader(TensorDataset(x,y), batch_size=10, shuffle=True)

# loop over the dataset 100 times
for epoch in range(100):
    epoch_loss = 0.0
    # loop over batches
```

PyTorch code

- Define a neural network
- Initialize params with He initialization
- Define loss function
- Choose optimization algorithm
- Choose initial learning rate
- Choose learning rates schedule
- Make some random data
- Train for 100 batches

```
model.apply(weights_init)

# choose least squares loss function
criterion = nn.MSELoss()
# construct SGD optimizer and initialize learning rate and momentum
optimizer = torch.optim.SGD(model.parameters(), lr = 0.01, momentum=0.9)
# object that decreases learning rate by half every 10 epochs
scheduler = StepLR(optimizer, step_size=10, gamma=0.5)

# create 100 dummy data points and store in data loader class
x = torch.randn(100, D_i)
y = torch.randn(100, D_o)
data_loader = DataLoader(TensorDataset(x,y), batch_size=10, shuffle=True)

# loop over the dataset 100 times
for epoch in range(100):
    epoch_loss = 0.0
    # loop over batches
    for i, data in enumerate(data_loader):
        # retrieve inputs and labels for this batch
        x_batch, y_batch = data
        # zero the parameter gradients
        optimizer.zero_grad()
        # forward pass
        pred = model(x_batch)
        loss = criterion(pred, y_batch)
        # backward pass
        loss.backward()
        # SGD update
        optimizer.step()
        # update statistics
        epoch_loss += loss.item()
    # print error
    print(f'Epoch {epoch:5d}, loss {epoch_loss:.3f}')
    # tell scheduler to consider updating learning rate
    scheduler.step()
```

CM20315 - Machine Learning

Prof. Simon Prince
7b Initialization



Initialization

- Need for initialization
- He initialization
- Interlude: Expectations
- Show that $\mathbb{E}[f'_i] = 0$
- Write variance of pre-activations f' in terms of activations h in previous layer

$$\sigma_{f'}^2 = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

- Write variance of pre-activations f' in terms of pre-activations f in previous layer

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

Initialization

- Consider standard building block of NN in terms of preactivations:

$$\begin{aligned}\mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k \\ &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{a}[\mathbf{f}_{k-1}]\end{aligned}$$

- How do we initialize the biases and weights?
- Equivalent to choosing starting point in Gabor/Linear regression models

Initialization

- Consider standard building block of NN in terms of *preativations*:

$$\begin{aligned}\mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k \\ &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{a}[\mathbf{f}_{k-1}]\end{aligned}$$

- Set all the biases to 0

$$\boldsymbol{\beta}_k = \mathbf{0}$$

- Weights normally distributed

- mean 0
 - variance σ_{Ω}^2

- What will happen as we move through the network if σ_{Ω}^2 is very small?
- What will happen as we move through the network if σ_{Ω}^2 is very large?

Backprop summary

Backward pass: We start with the derivative $\partial\ell_i/\partial\mathbf{f}_K$ of the loss function ℓ_i with respect to the network output \mathbf{f}_K and work backward through the network:

$$\begin{aligned}\frac{\partial\ell_i}{\partial\boldsymbol{\beta}_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} & k \in \{K, K-1, \dots, 1\} \\ \frac{\partial\ell_i}{\partial\boldsymbol{\Omega}_k} &= \frac{\partial\ell_i}{\partial\mathbf{f}_k} \mathbf{h}_k^T & k \in \{K, K-1, \dots, 1\} \\ \frac{\partial\ell_i}{\partial\mathbf{f}_{k-1}} &= \mathbb{I}[\mathbf{f}_{k-1} > 0] \odot \left(\boldsymbol{\Omega}_k^T \frac{\partial\ell_i}{\partial\mathbf{f}_k} \right), & k \in \{K, K-1, \dots, 1\}\end{aligned}\tag{7.13}$$

where \odot denotes pointwise multiplication and $\mathbb{I}[\mathbf{f}_{k-1} > 0]$ is a vector containing ones where \mathbf{f}_{k-1} is greater than zero and zeros elsewhere. Finally, we compute the derivatives with respect to the first set of biases and weights:

$$\begin{aligned}\frac{\partial\ell_i}{\partial\boldsymbol{\beta}_0} &= \frac{\partial\ell_i}{\partial\mathbf{f}_0} \\ \frac{\partial\ell_i}{\partial\boldsymbol{\Omega}_0} &= \frac{\partial\ell_i}{\partial\mathbf{f}_0} \mathbf{x}_i^T\end{aligned}$$

Initialization

- Need for initialization
- He initialization
- Interlude: Expectations
- Show that $\mathbb{E}[f'_i] = 0$
- Write variance of pre-activations f' in terms of activations h in previous layer

$$\sigma_{f'}^2 = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

- Write variance of pre-activations f' in terms of pre-activations f in previous layer

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

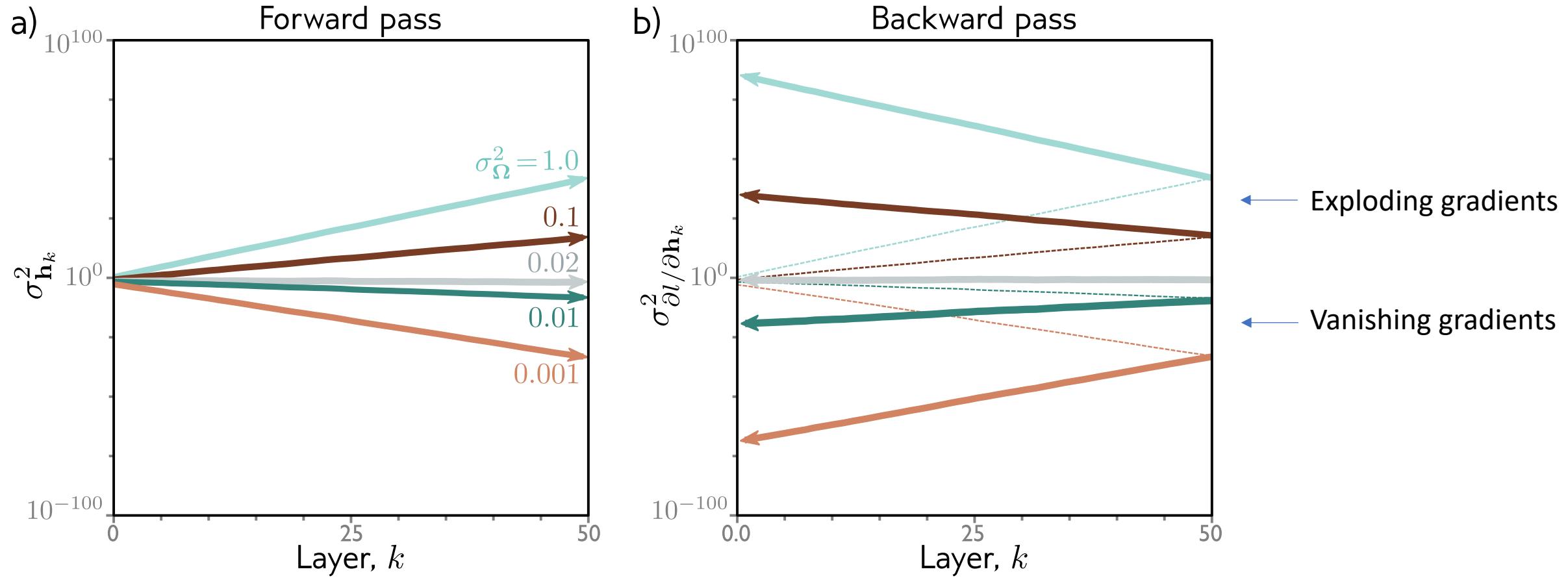


Figure 7.4 Weight initialization. Consider a deep network with 50 hidden layers and $D_h = 100$ hidden units per layer. The network has a 100 dimensional input \mathbf{x} initialized with values from a standard normal distribution, a single output fixed at $y = 0$, and a least squares loss function. The bias vectors β_k are initialized to zero and the weight matrices Ω_k are initialized with a normal distribution with mean zero and five different variances $\sigma_{\Omega}^2 \in \{0.001, 0.01, 0.02, 0.1, 1.0\}$. a)

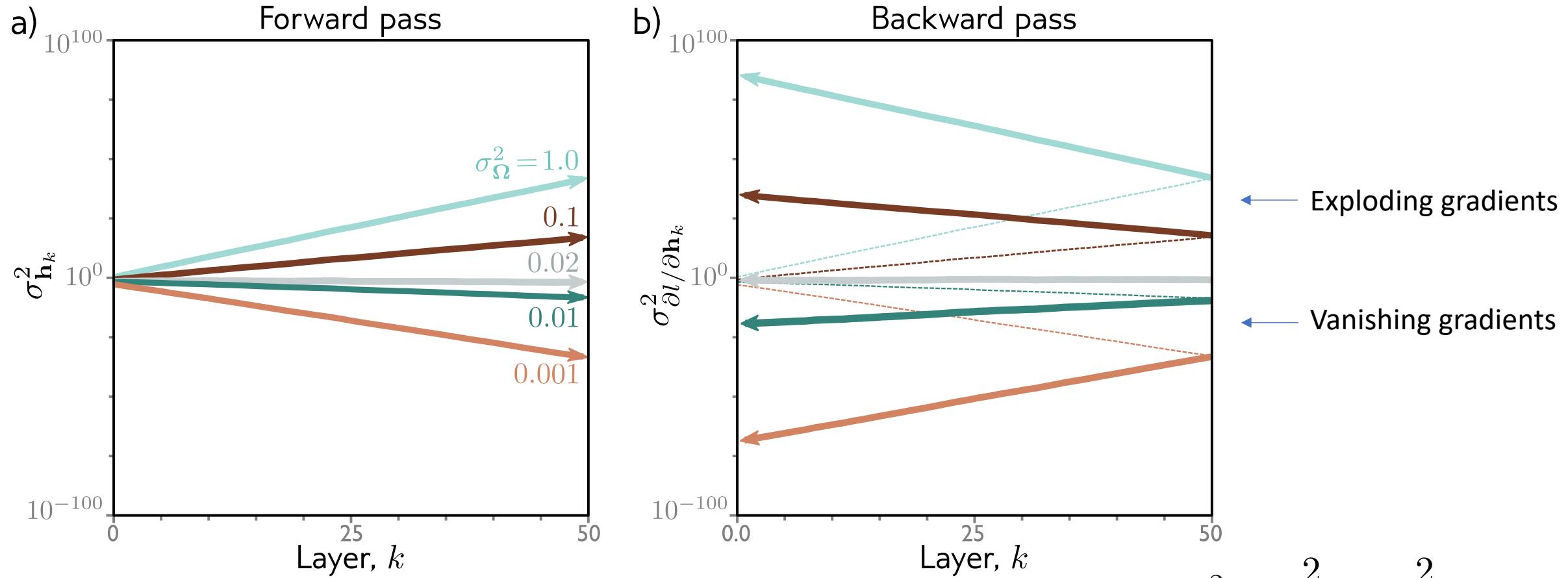
He initialization (assumes ReLU)

- Forward pass: want the variance of hidden unit activations in layer $k+1$ to be the same as variance of activations in layer k :

$$\sigma_{\Omega}^2 = \frac{2}{D_h} \quad \xleftarrow{\text{Number of units at layer } k}$$

- Backward pass: want the variance of gradients at layer k to be the same as variance of gradient in layer $k+1$:

$$\sigma_{\Omega}^2 = \frac{2}{D_{h'}} \quad \xleftarrow{\text{Number of units at layer } k+1}$$



$$\sigma_\Omega^2 = \frac{2}{D_h} = \frac{2}{100} = 0.02$$

Figure 7.4 Weight initialization. Consider a deep network with 50 hidden layers and $D_h = 100$ hidden units per layer. The network has a 100 dimensional input \mathbf{x} initialized with values from a standard normal distribution, a single output fixed at $y = 0$, and a least squares loss function. The bias vectors β_k are initialized to zero and the weight matrices Ω_k are initialized with a normal distribution with mean zero and five different variances $\sigma_\Omega^2 \in \{0.001, 0.01, 0.02, 0.1, 1.0\}$. a)

Initialization

- Need for initialization
- He initialization
- **Interlude: Expectations**
- Show that $\mathbb{E}[f'_i] = 0$
- Write variance of pre-activations f' in terms of activations h in previous layer

$$\sigma_{f'}^2 = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

- Write variance of pre-activations f' in terms of pre-activations f in previous layer

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

Expectations

$$\mathbb{E}[g[x]] = \int g[x] Pr(x) dx,$$

Interpretation: what is the average value of $g[x]$ when taking into account the probability of x ?

Could approximate, by sampling many values of x from the distribution, calculating $g[x]$, and taking average:

$$\mathbb{E}[g[x]] \approx \frac{1}{N} \sum_{n=1}^N g[x_n^*] \quad \text{where} \quad x_n^* \sim Pr(x)$$

Expectations

Function $g[\bullet]$	Expectation
x	mean, μ
x^k	k th moment about zero
$(x - \mu)^k$	k th moment about the mean
$(x - \mu)^2$	variance
$(x - \mu)^3$	skew
$(x - \mu)^4$	kurtosis

Table B.1 Special cases of expectation. For some functions $g[x]$, the expectation $\mathbb{E}[g[x]]$ is given a special name. Here we use the notation μ_x to represent the mean with respect to random variable x .

Rules for manipulating expectation

$$\mathbb{E}[k] = k$$

$$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]] \quad \text{if } x, y \text{ independent}$$

Rule 1

$$\mathbb{E}[g[x]] = \int g[x] Pr(x) dx,$$

$$\begin{aligned}\mathbb{E}[\kappa] &= \int \kappa Pr(x) dx \\ &= \kappa \int Pr(x) dx \\ &= \kappa.\end{aligned}$$

Rules for manipulating expectation

$$\mathbb{E}[k] = k$$

$$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]] \quad \text{if } x, y \text{ independent}$$

Rule 2

$$\mathbb{E}[g[x]] = \int g[x]Pr(x)dx,$$

$$\begin{aligned}\mathbb{E}[\kappa \cdot g[x]] &= \int \kappa \cdot g[x]Pr(x)dx \\ &= \kappa \cdot \int g[x]Pr(x)dx \\ &= \kappa \cdot \mathbb{E}[g[x]]\end{aligned}$$

Rules for manipulating expectation

$$\mathbb{E}[k] = k$$

$$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]] \quad \text{if } x, y \text{ independent}$$

Rule 3

$$\mathbb{E}[g[x]] = \int g[x] Pr(x) dx,$$

$$\begin{aligned}\mathbb{E}[f[x] + g[x]] &= \int (f[x] + g[x]) Pr(x) dx \\&= \int (f[x] Pr(x) + g[x] Pr(x)) dx \\&= \int f[x] Pr(x) dx + \int g[x] Pr(x) dx \\&= \mathbb{E}[f[x]] + \mathbb{E}[g[x]]\end{aligned}$$

Rules for manipulating expectation

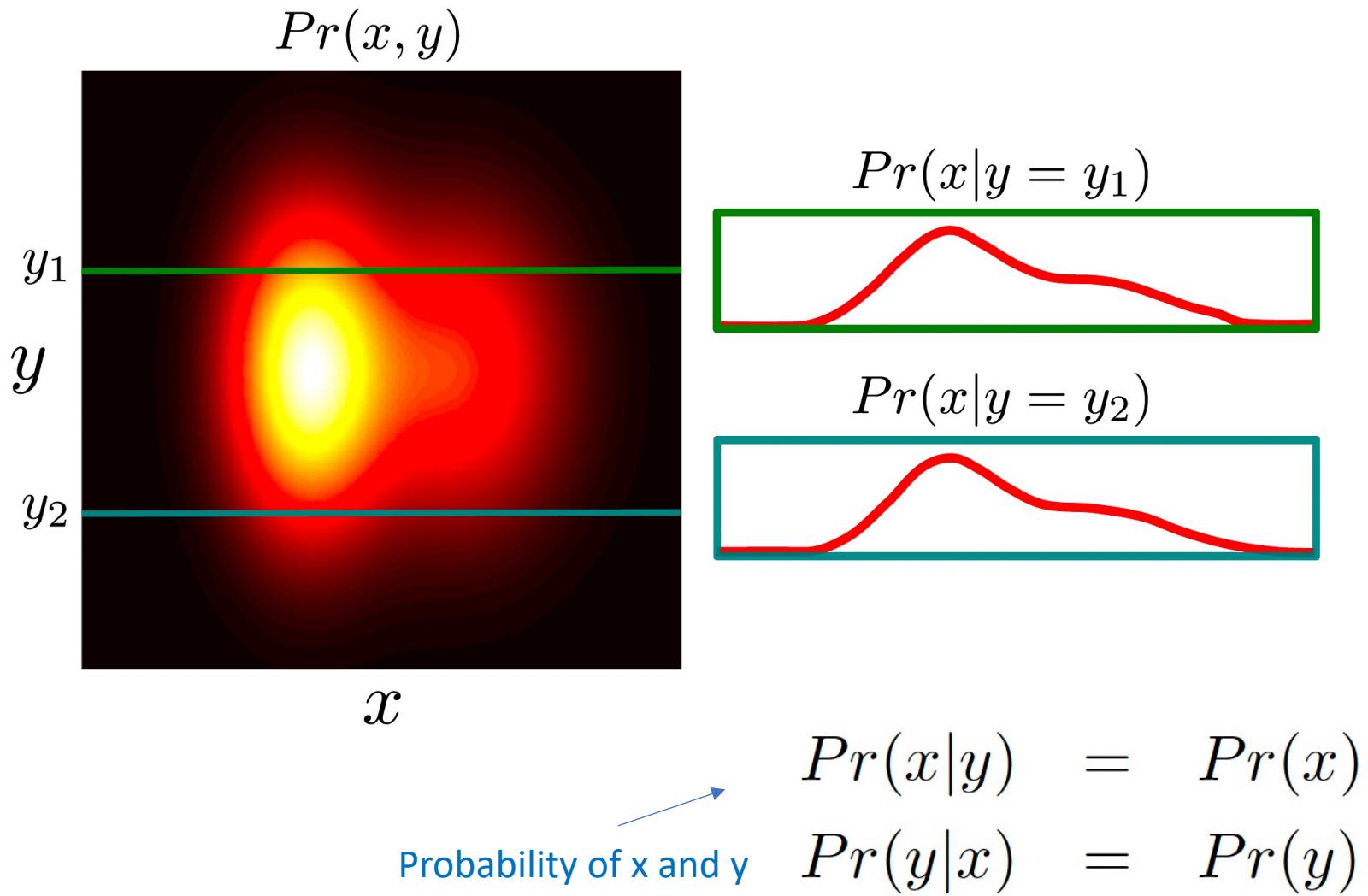
$$\mathbb{E}[k] = k$$

$$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$$

$$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$$

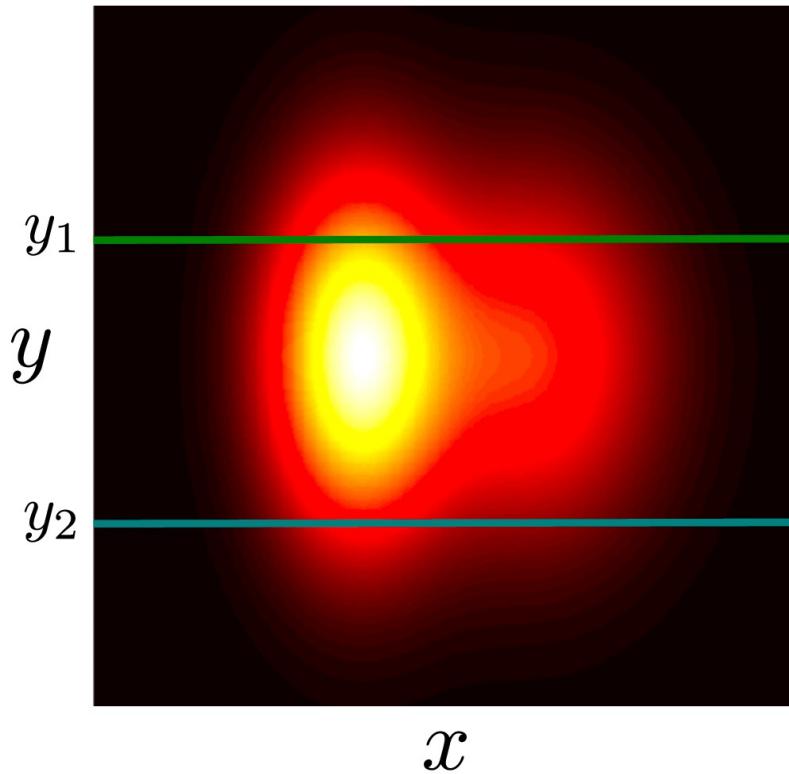
$$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]] \quad \text{if } x, y \text{ independent}$$

Independence



Independence

$$Pr(x, y)$$



$$Pr(x|y = y_1)$$

$$Pr(x|y = y_2)$$

$$Pr(x, y) = Pr(x)Pr(y)$$

Probability of x and y

Rule 4

$$\mathbb{E}[g[x]] = \int g[x]Pr(x)dx,$$

$$\begin{aligned}\mathbb{E}[f[x] \cdot g[y]] &= \int \int f[x] \cdot g[y] Pr(x, y) dx dy \\ &= \int \int f[x] \cdot g[y] Pr(x) Pr(y) dx dy \\ &= \int f[x] Pr(x) dx \int g[y] Pr(y) dy \\ &= \mathbb{E}[f[x]] \mathbb{E}[g[y]]\end{aligned}$$

if x, y independent

Because
independent

Now let's prove:

$$\mathbb{E}[(x - \mu)^2] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$$

Keeping in mind:

$$\mathbb{E}[x] = \mu$$

Now let's prove:

$$\mathbb{E}[(x - \mu)^2] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$$

Keeping in mind:

$$\mathbb{E}[x] = \mu$$

$$\text{Rule 1: } \mathbb{E}[k] = k$$

$$\text{Rule 2: } \mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$$

$$\text{Rule 3: } \mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$$

$$\text{Def'n } \mathbb{E}[x] = \mu$$

$$\mathbb{E}[(x - \mu^2)] = \mathbb{E}[x^2 - 2x\mu + \mu^2]$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$



$$\begin{aligned}\mathbb{E}[(x - \mu^2)] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\ &= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2]\end{aligned}$$

Rule 1:	$\mathbb{E}[k] = k$	
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$	
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$	
Def'n	$\mathbb{E}[x] = \mu$	

$$\begin{aligned}
 \mathbb{E}[(x - \mu^2)] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\
 &= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\
 &= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2
 \end{aligned}$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$



$$\begin{aligned}\mathbb{E}[(x - \mu^2)] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\&= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\&= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2 \\&= \mathbb{E}[x^2] - 2\mu^2 + \mu^2\end{aligned}$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Def'n $\mathbb{E}[x] = \mu$

$$\begin{aligned}\mathbb{E}[(x - \mu)^2] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\&= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\&= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2 \\&= \mathbb{E}[x^2] - 2\mu^2 + \mu^2 \\&= \mathbb{E}[x^2] - \mu^2\end{aligned}$$

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Def'n	$\mathbb{E}[x] = \mu$



$$\begin{aligned}
\mathbb{E}[(x - \mu^2)] &= \mathbb{E}[x^2 - 2x\mu + \mu^2] \\
&= \mathbb{E}[x^2] - \mathbb{E}[2x\mu] + \mathbb{E}[\mu^2] \\
&= \mathbb{E}[x^2] - 2\mu\mathbb{E}[x] + \mu^2 \\
&= \mathbb{E}[x^2] - 2\mu^2 + \mu^2 \\
&= \mathbb{E}[x^2] - \mu^2 \\
&= \mathbb{E}[x^2] - E[x]^2
\end{aligned}$$

Initialization

- Need for initialization
- He initialization
- Interlude: Expectations
- Show that $\mathbb{E}[f'_i] = 0$
- Write variance of pre-activations f' in terms of activations h in previous layer

$$\sigma_{f'}^2 = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

- Write variance of pre-activations f' in terms of pre-activations f in previous layer

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

Initialization

- Consider standard building block of NN in terms of *preactivations*:

$$\begin{aligned}\mathbf{f}_k &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{h}_k \\ &= \boldsymbol{\beta}_k + \boldsymbol{\Omega}_k \mathbf{a}[\mathbf{f}_{k-1}]\end{aligned}$$

- Set all the biases to 0

$$\boldsymbol{\beta}_k = \mathbf{0}$$

- Weights normally distributed
 - mean 0
 - variance σ_{Ω}^2
- What will happen as we move through the network if σ_{Ω}^2 is very small?
- What will happen as we move through the network if σ_{Ω}^2 is very large?

Aim: keep variance same between two layers

$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega} \mathbf{h}$$

Consider the mean of the pre-activations:

$$\mathbb{E}[f'_i] = \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right]$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Rule 4: $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent



$$\begin{aligned}\mathbb{E}[f'_i] &= \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right] \\ &= \mathbb{E} [\beta_i] + \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij} h_j]\end{aligned}$$

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Rule 4: $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent



$$\mathbb{E}[f'_i] = \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right]$$

$$= \mathbb{E} [\beta_i] + \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij} h_j]$$

$$= \mathbb{E} [\beta_i] + \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij}] \mathbb{E} [h_j]$$

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent

$$\mathbb{E}[f'_i] = \mathbb{E} \left[\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right]$$

$$= \mathbb{E}[\beta_i] + \sum_{j=1}^{D_h} \mathbb{E}[\Omega_{ij} h_j]$$

Set all the biases to 0

$$= \mathbb{E}[\beta_i] + \sum_{j=1}^{D_h} \mathbb{E}[\Omega_{ij}] \mathbb{E}[h_j]$$

Weights normally distributed

mean 0

variance σ_Ω^2

$$= 0 + \sum_{j=1}^{D_h} 0 \cdot \mathbb{E}[h_j] = 0$$

Initialization

- Need for initialization
- He initialization
- Interlude: Expectations
- Show that $\mathbb{E}[f'_i] = 0$
- Write variance of pre-activations f' in terms of activations h in previous layer

$$\sigma_{f'}^2 = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

- Write variance of pre-activations f' in terms of pre-activations f in previous layer

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

Aim: keep variance same between two layers

$$\mathbf{f}' = \boldsymbol{\beta} + \boldsymbol{\Omega} \mathbf{h}$$

$$\mathbf{h} = \mathbf{a}[\mathbf{f}],$$

$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2$$

$$\longrightarrow \mathbb{E}[(x - \mu)^2] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$$

- | | |
|---------|---|
| Rule 1: | $\mathbb{E}[k] = k$ |
| Rule 2: | $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$ |
| Rule 3: | $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$ |
| Rule 4: | $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent |

$$\begin{aligned}\sigma_{f'}^2 &= \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2 \\ &= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0\end{aligned}$$

Set all the biases to 0

Weights normally distributed
mean 0
variance σ_Ω^2

- | | |
|---------|---|
| Rule 1: | $\mathbb{E}[k] = k$ |
| Rule 2: | $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$ |
| Rule 3: | $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$ |
| Rule 4: | $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent |

$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2$$

$$= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0$$

$$= \mathbb{E} \left[\left(\sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right]$$

Set all the biases to 0



Weights normally distributed

mean 0

variance σ_Ω^2

Rule 1: $\mathbb{E}[k] = k$

Rule 2: $\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$

Rule 3: $\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$

Rule 4: $\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent



$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2$$

$$= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0$$

$$= \mathbb{E} \left[\left(\sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right]$$

$$= \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij}^2] \mathbb{E} [h_j^2]$$

Set all the biases to 0

Weights normally distributed
mean 0

variance σ_Ω^2

Rule 1:	$\mathbb{E}[k] = k$
Rule 2:	$\mathbb{E}[k \cdot g[x]] = k \cdot \mathbb{E}[g[x]]$
Rule 3:	$\mathbb{E}[f[x] + g[x]] = \mathbb{E}[f[x]] + \mathbb{E}[g[x]]$
Rule 4:	$\mathbb{E}[f[x]g[y]] = \mathbb{E}[f[x]]\mathbb{E}[g[y]]$ if x, y independent

$$\sigma_{f'}^2 = \mathbb{E}[f_i'^2] - \mathbb{E}[f_i']^2$$

$$= \mathbb{E} \left[\left(\beta_i + \sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right] - 0$$

$$= \mathbb{E} \left[\left(\sum_{j=1}^{D_h} \Omega_{ij} h_j \right)^2 \right]$$

$$= \sum_{j=1}^{D_h} \mathbb{E} [\Omega_{ij}^2] \mathbb{E} [h_j^2]$$

$$= \sum_{j=1}^{D_h} \sigma_\Omega^2 \mathbb{E} [h_j^2] = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

Set all the biases to 0

Weights normally distributed
mean 0

variance σ_Ω^2

Initialization

- Need for initialization
- He initialization
- Interlude: Expectations
- Show that $\mathbb{E}[f'_i] = 0$
- Write variance of pre-activations f' in terms of activations h in previous layer

$$\sigma_{f'}^2 = \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2]$$

- Write variance of pre-activations f' in terms of pre-activations f in previous layer

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

$$\begin{aligned}
\sigma_{f'}^2 &= \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [h_j^2] \\
&= \sigma_\Omega^2 \sum_{j=1}^{D_h} \mathbb{E} [\text{ReLU}[f_j]^2] \\
&= \sigma_\Omega^2 \sum_{j=1}^{D_h} \int_{-\infty}^{\infty} \text{ReLU}[f_j]^2 Pr(f_j) df_j \\
&= \sigma_\Omega^2 \sum_{j=1}^{D_h} \int_{-\infty}^{\infty} (\mathbb{I}[f_j > 0] f_j)^2 Pr(f_j) df_j \\
&= \sigma_\Omega^2 \sum_{j=1}^{D_h} \int_0^{\infty} f_j^2 Pr(f_j) df_j \\
&= \sigma_\Omega^2 \sum_{j=1}^{D_h} \frac{\sigma_f^2}{2} = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}
\end{aligned}$$

Aim: keep variance same between two layers

$$\sigma_{f'}^2 = \frac{D_h \sigma_\Omega^2 \sigma_f^2}{2}$$

Should choose:

$$\sigma_\Omega^2 = \frac{2}{D_h}$$

This is called He initialization.