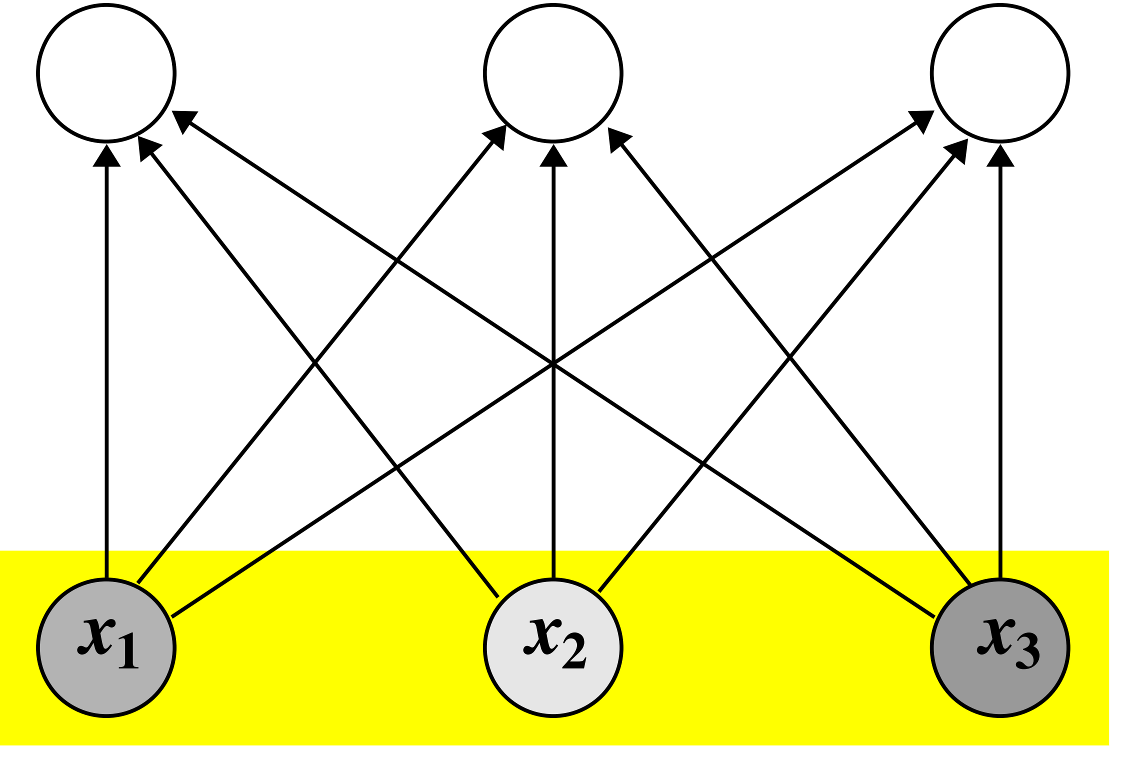
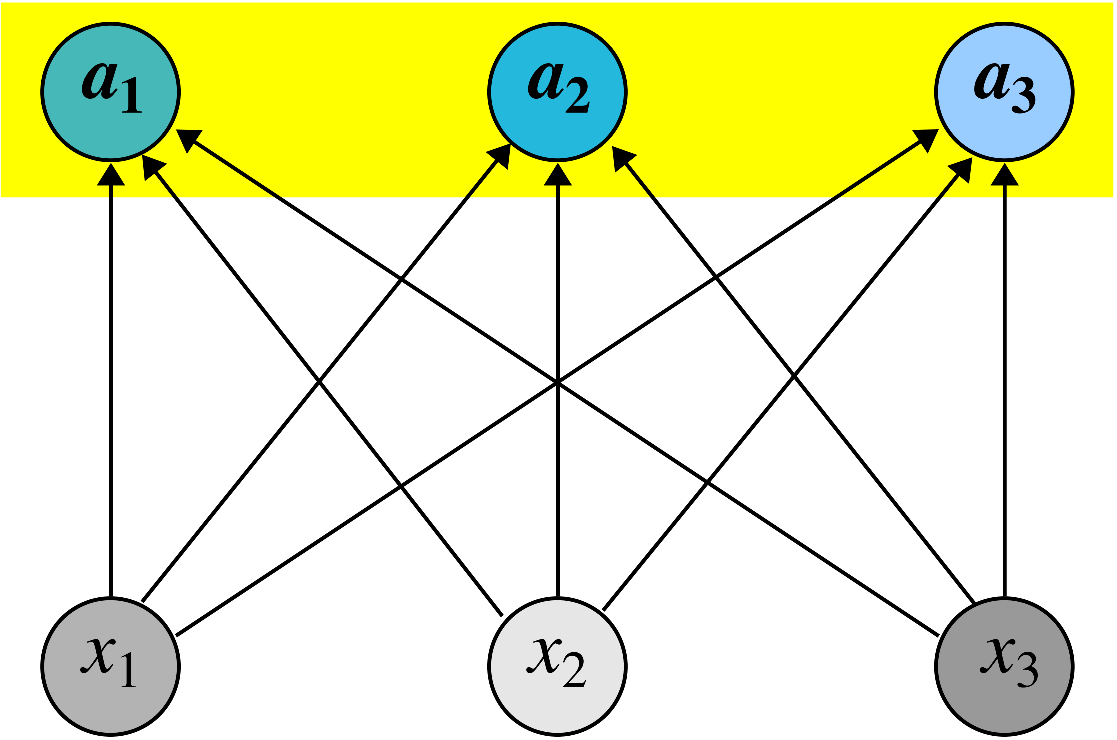
Derivation of Backpropagation

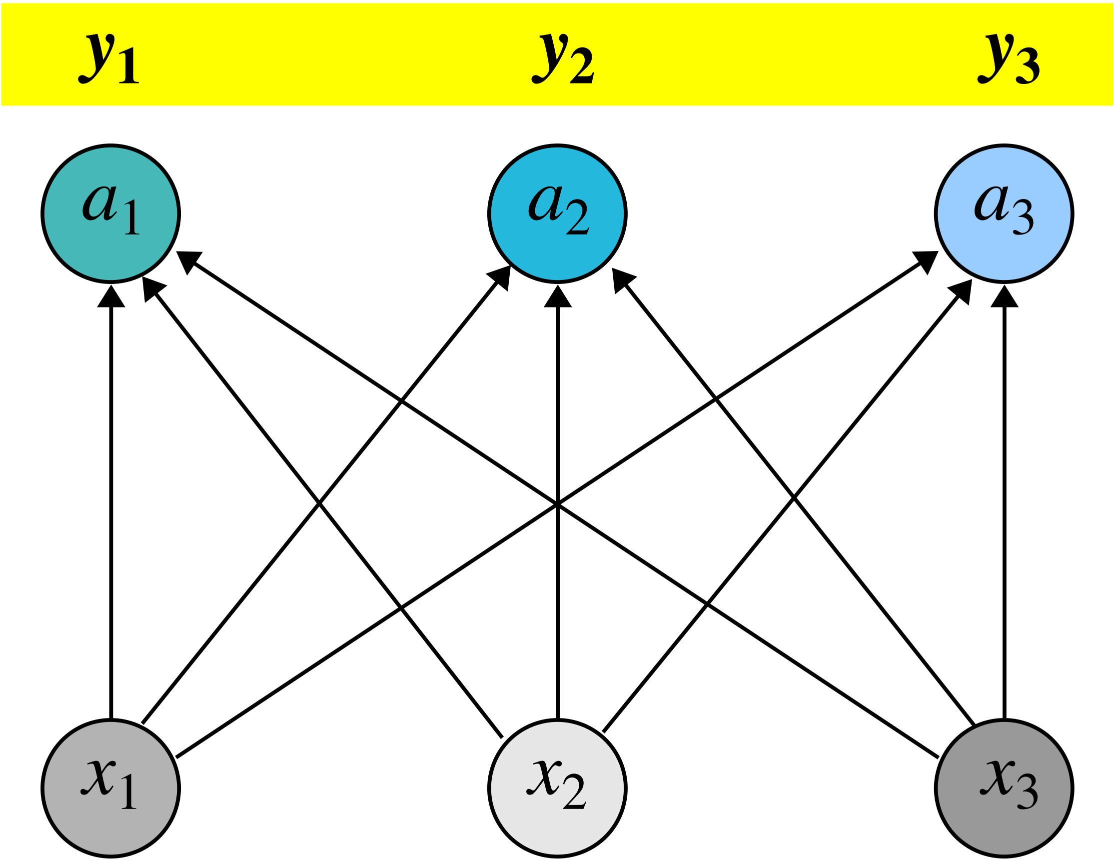
Input Values



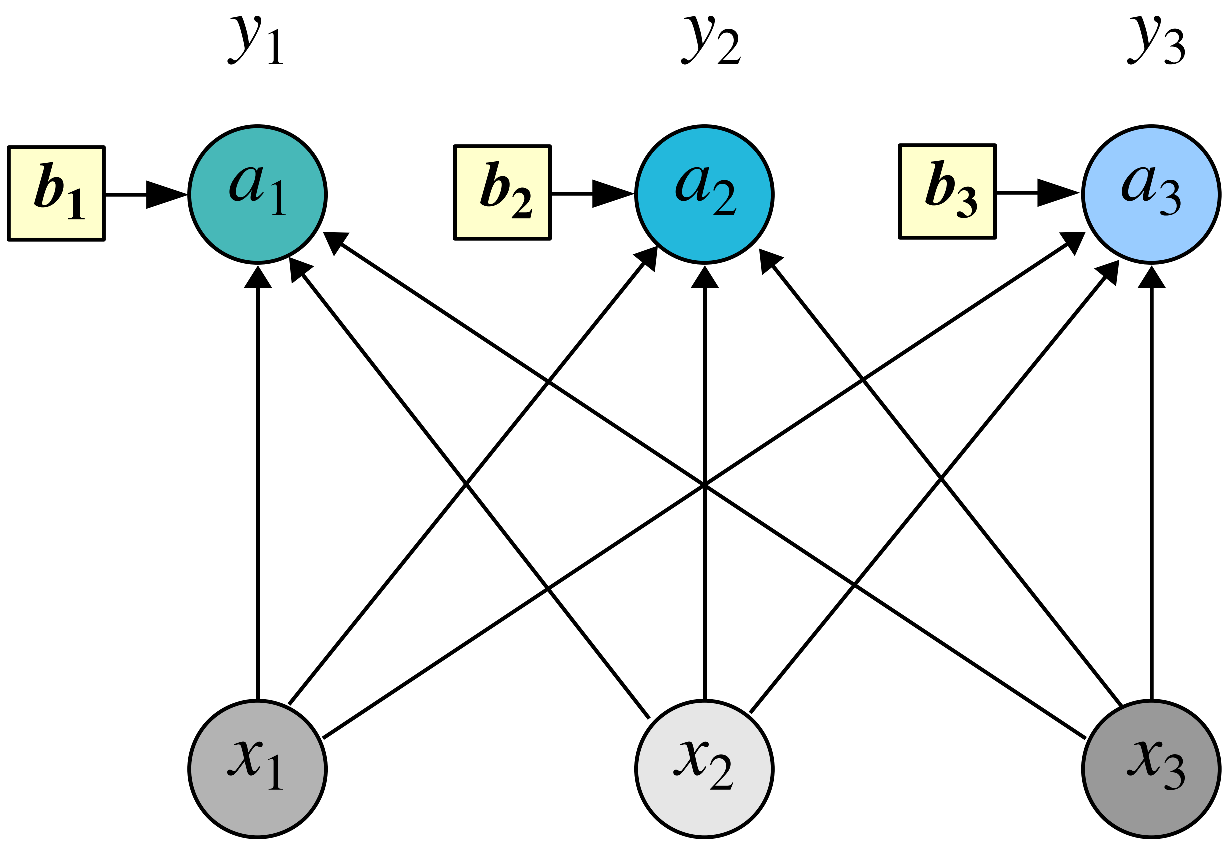
Activation Values



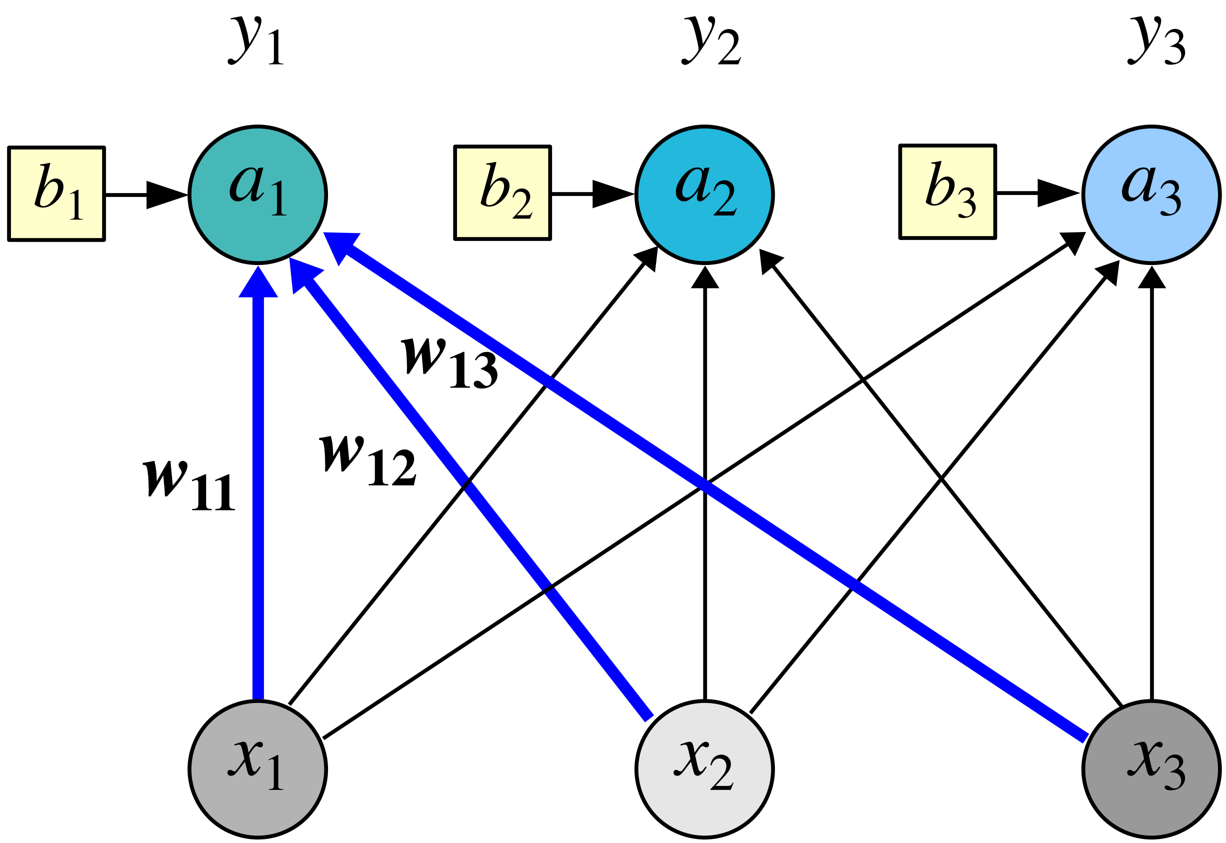
Target Values



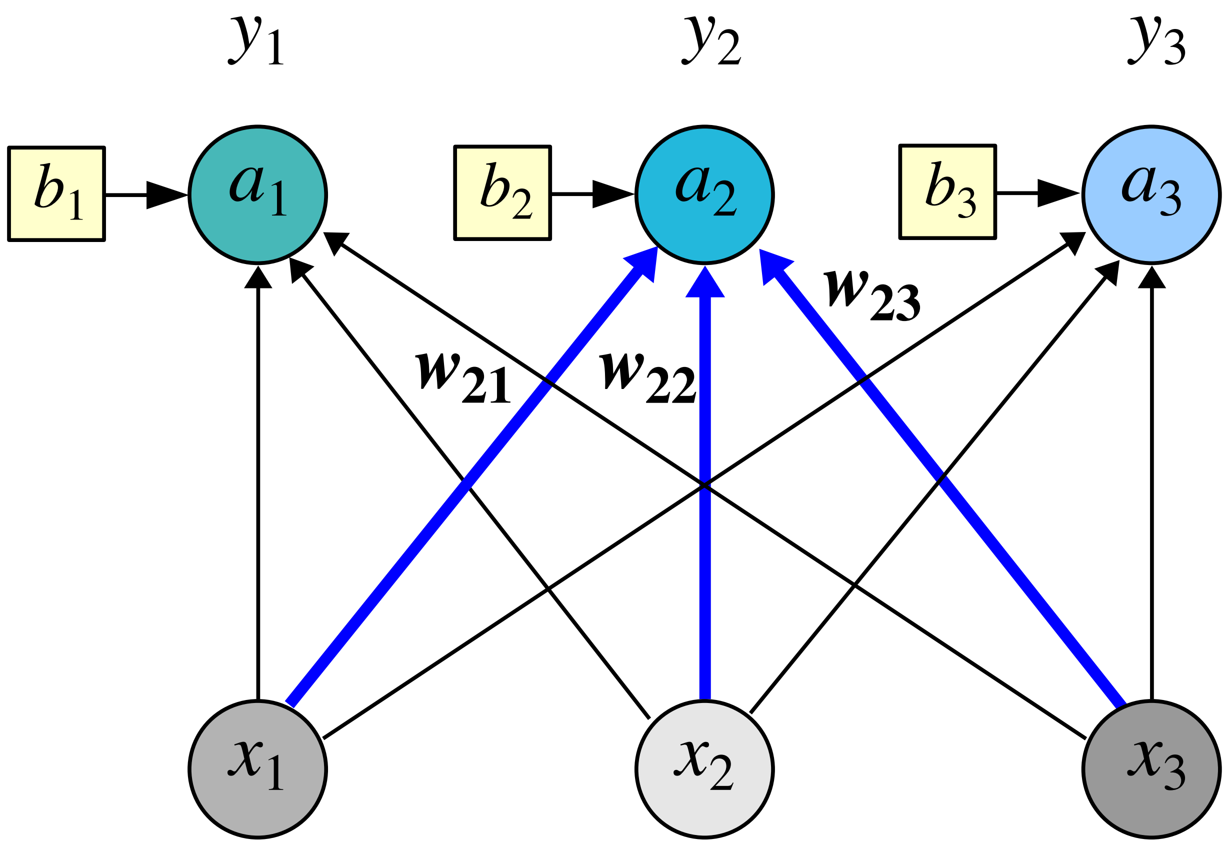
Biases



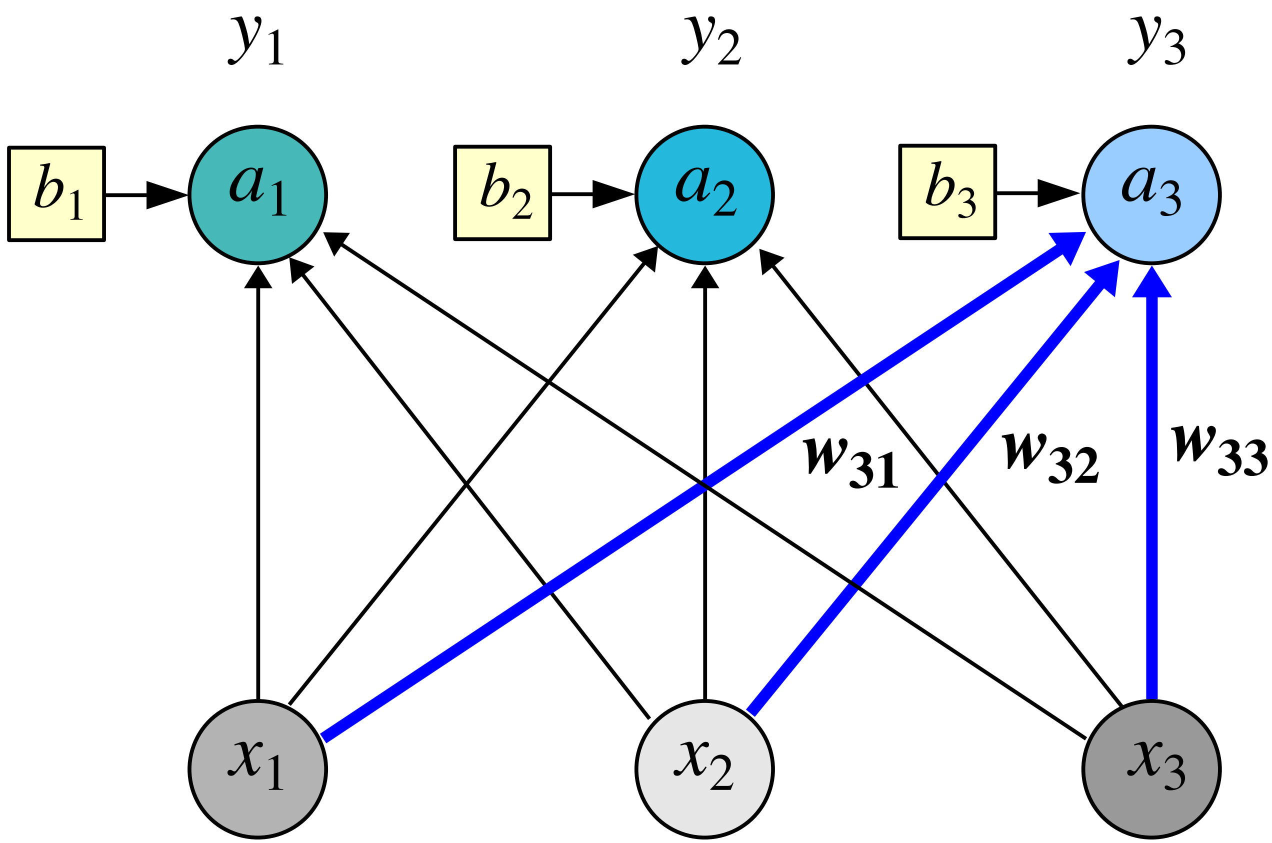
Connection Weights



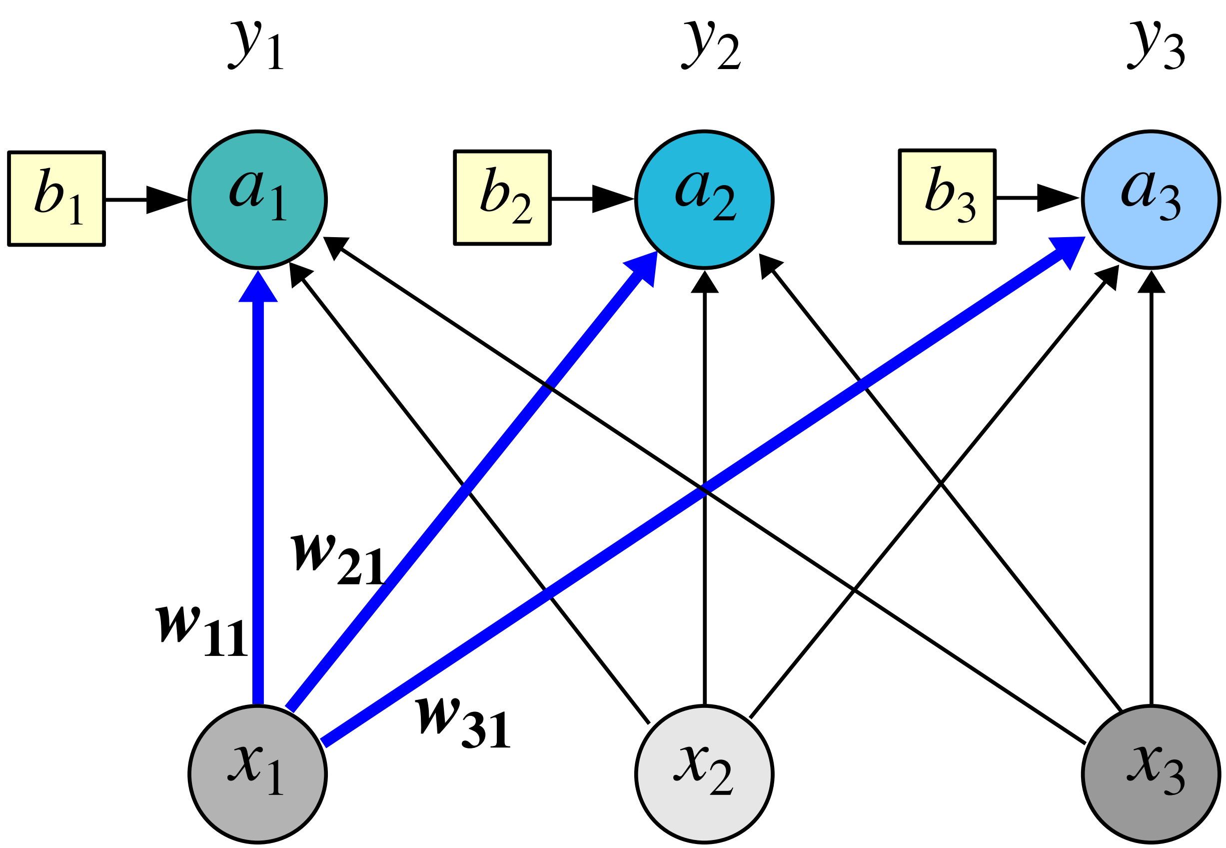
Connection Weights



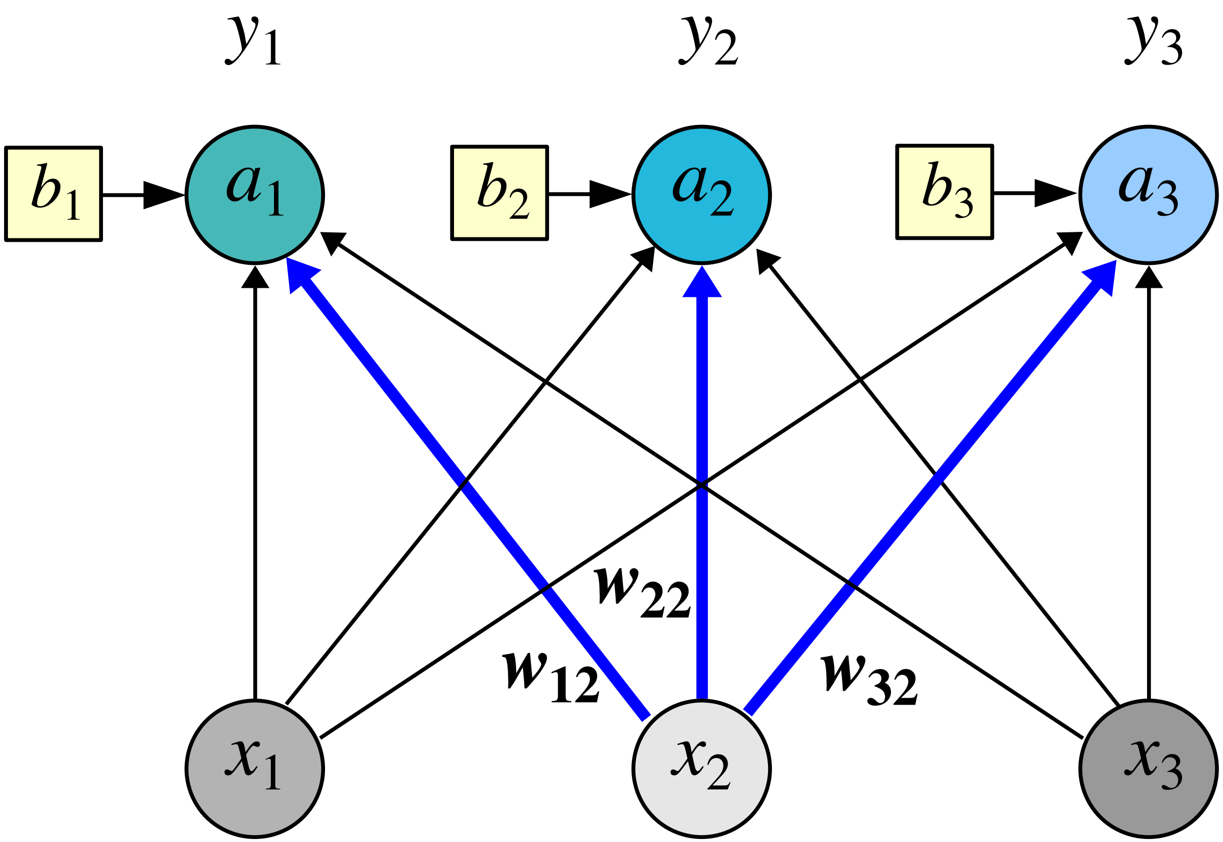
Connection Weights



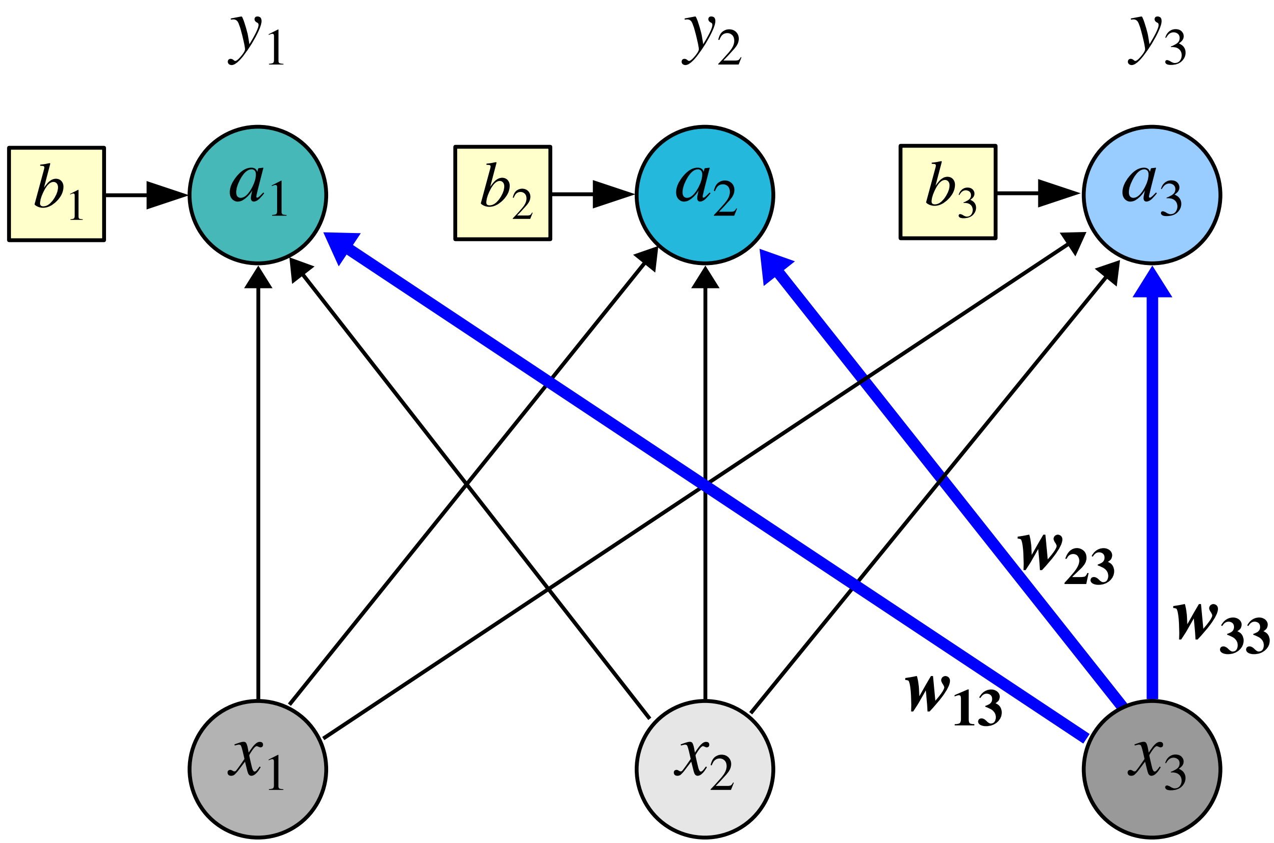
Connection Weights



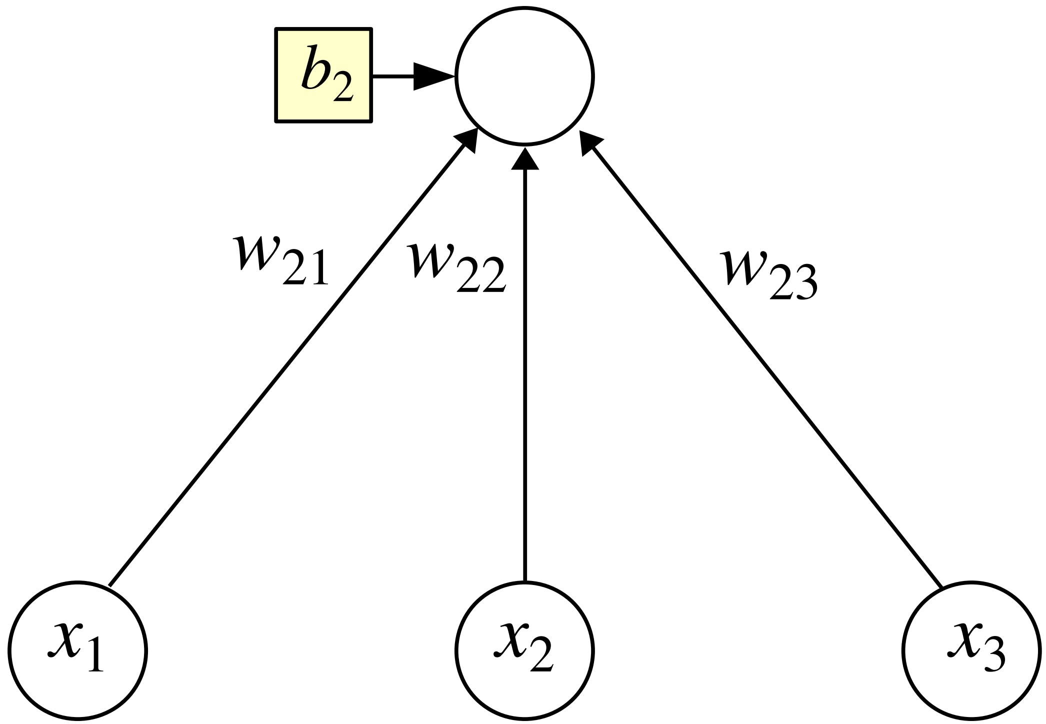
Connection Weights



Connection Weights



Compute Activation

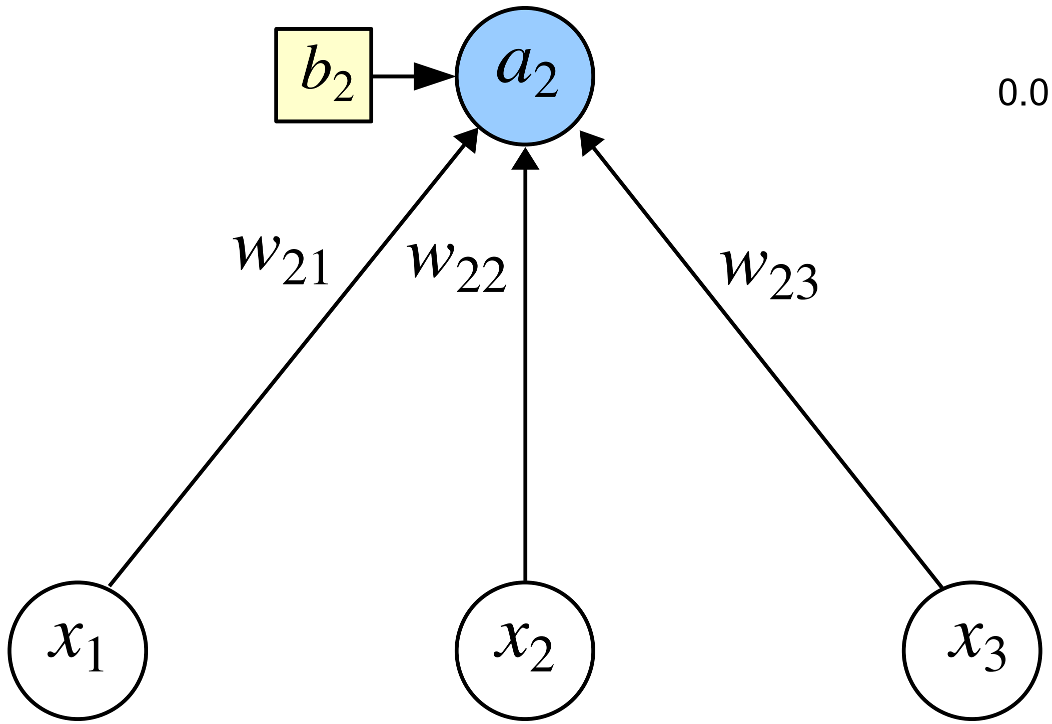


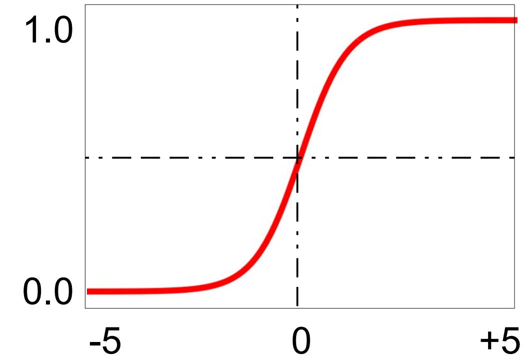
$$z_2 = w_{21}\,x_1 + w_{22}\,x_2 + w_{23}\,x_3 + b_2$$

$$z_2 = \left(\sum_k w_{2k}\,x_k \right) + b_2$$

Compute Activation

$$a_2 = \sigma(z_2)$$



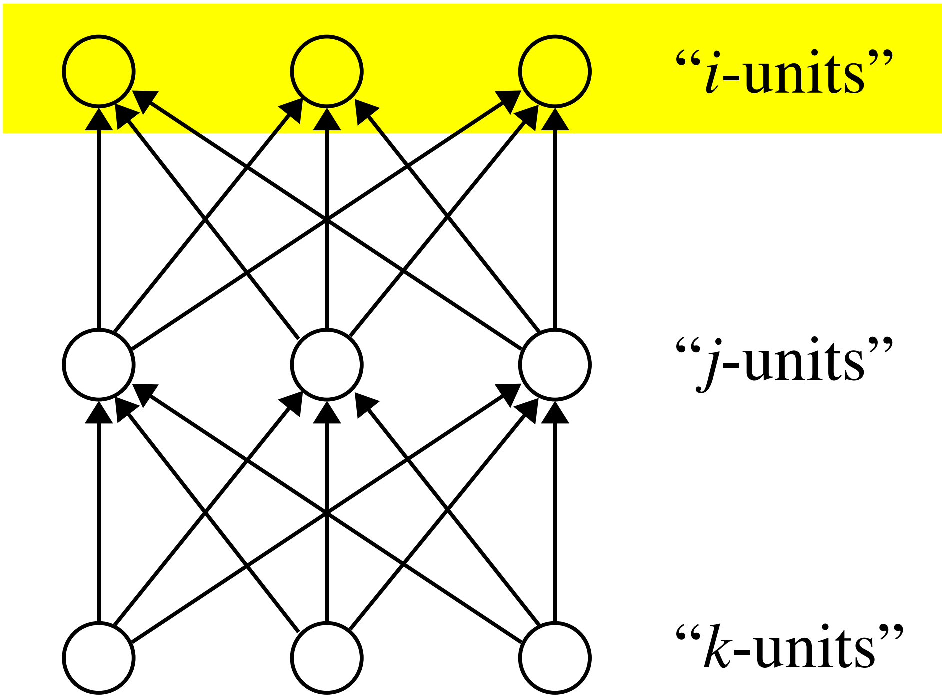


$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$$z_2 = w_{21}\,x_1 + w_{22}\,x_2 + w_{23}\,x_3 + b_2$$

$$z_2 = \left(\sum_k w_{2k}\,x_k \right) + b_2$$

Notation

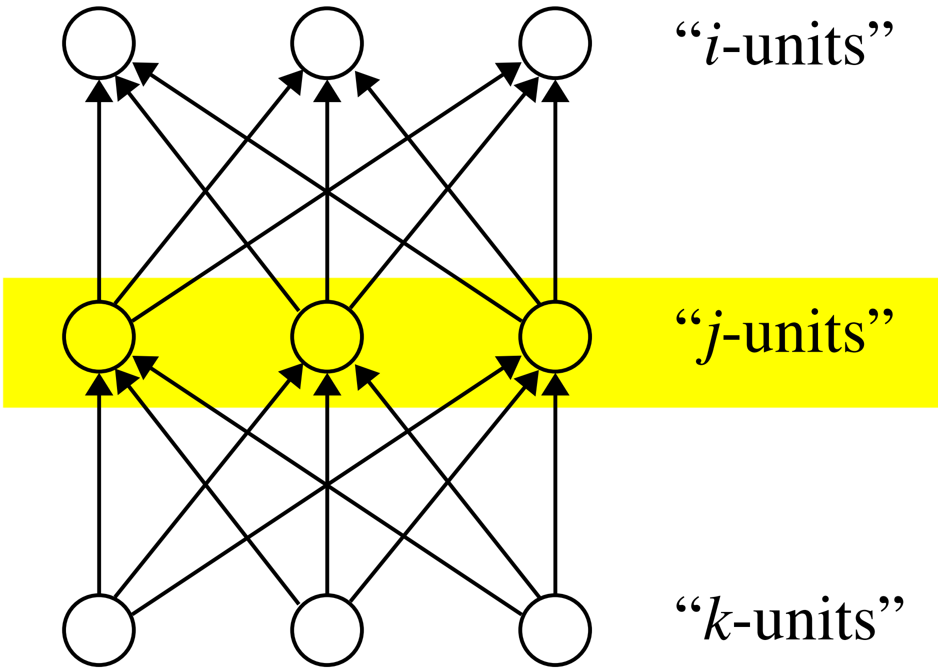


y_i = target value for output unit i

a_i = activation of output unit i

b_i = bias of output unit i

Notation



y_i = target value for output unit i

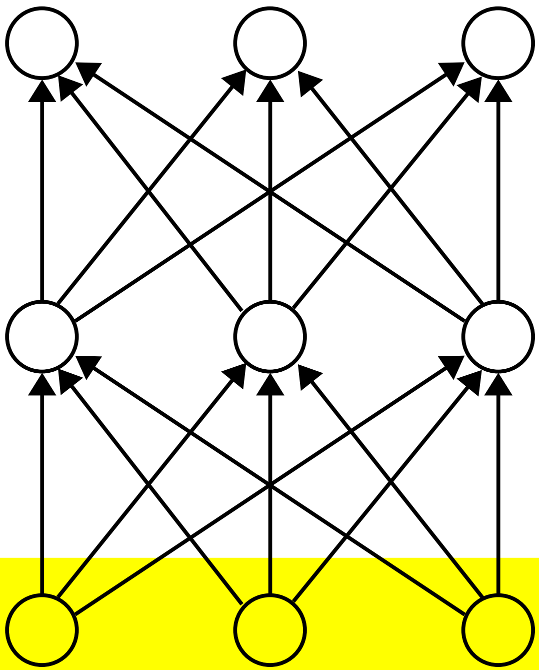
a_i = activation of output unit i

b_i = bias of output unit i

a_j = activation of hidden unit j

b_j = bias of hidden unit j

Notation



y_i = target value for output unit i

"i-units"

a_i = activation of output unit i

b_i = bias of output unit i

"j-units"

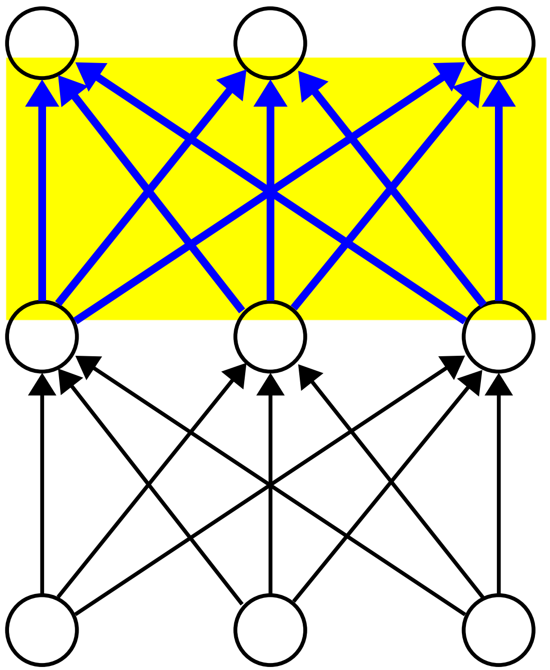
a_j = activation of hidden unit j

b_j = bias of hidden unit j

"k-units"

x_k = activation of input unit k

Notation



"i-units"

y_i = target value for output unit i

a_i = activation of output unit i

b_i = bias of output unit i

"j-units"

a_j = activation of hidden unit j

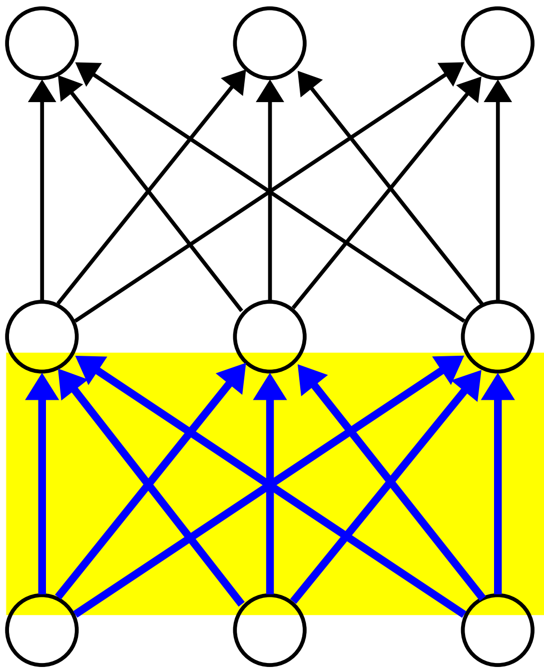
b_j = bias of hidden unit j

"k-units"

x_k = activation of input unit k

w_{ij} = connection weight from hidden unit j to output unit i

Notation



y_i = target value for output unit i

"i-units"

a_i = activation of output unit i

b_i = bias of output unit i

"j-units"

a_j = activation of hidden unit j

b_j = bias of hidden unit j

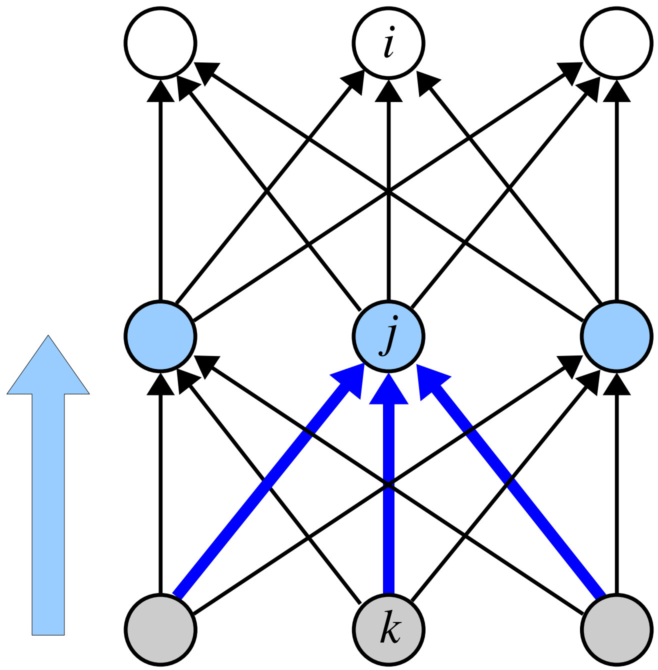
"k-units"

x_k = activation of input unit k

w_{ij} = connection weight from hidden unit j to output unit i

w_{jk} = connection weight from input unit k to hidden unit j

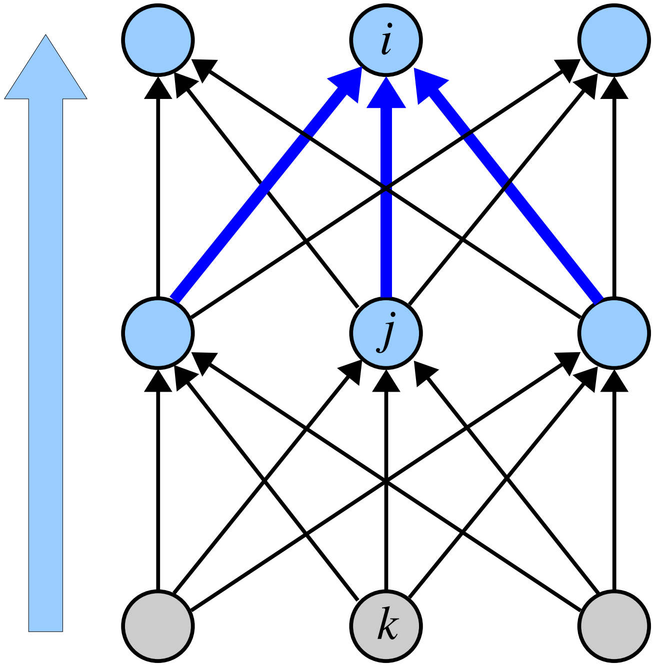
Forward Pass



$$z_j = \left(\sum_k w_{jk} x_k \right) + b_j \qquad a_j = \sigma(z_j)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Forward Pass



$$z_i = \left(\sum_j w_{ij}\, a_j \right) + b_i \qquad a_i = \sigma(z_i)$$

$$z_j = \left(\sum_k w_{jk}\, x_k \right) + b_j \qquad a_j = \sigma(z_j)$$

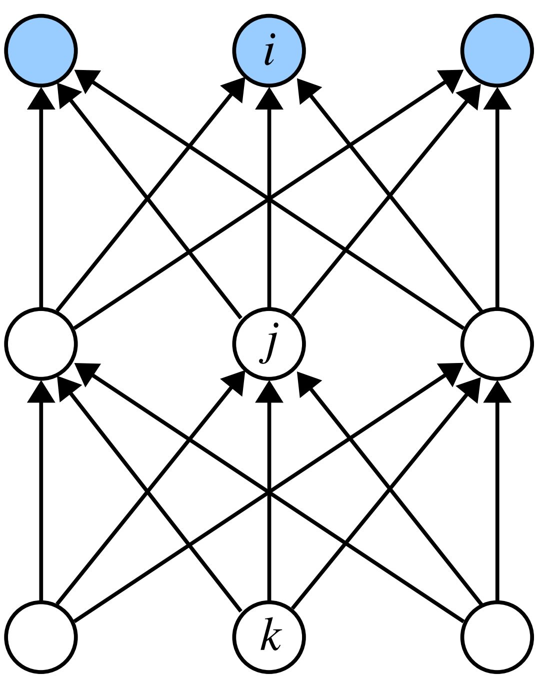
$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Cost/Loss for One Pattern

$$\frac{1}{2}(y_1 - a_1)^2 + \frac{1}{2}(y_2 - a_2)^2 + \ldots + \frac{1}{2}(y_i - a_i)^2 + \ldots$$

$y_1 \quad y_2 \quad \cdots \quad y_i$ ← target values

$a_1 \quad a_2 \quad \cdots \quad a_i$ ← output values



$$\sum_i \frac{1}{2}(y_i - a_i)^2$$

Cost/Loss for *n* Patterns

$$\frac{1}{2}(y_1 - a_1)^2 + \frac{1}{2}(y_2 - a_2)^2 + \ . \ . \ . + \frac{1}{2}(y_i - a_i)^2 + \ . \ . \ . \qquad \textit{(pattern 1)}$$

$$+ \ \frac{1}{2}(y_1 - a_1)^2 + \frac{1}{2}(y_2 - a_2)^2 + \ . \ . \ . + \frac{1}{2}(y_i - a_i)^2 + \ . \ . \ . \qquad \textit{(pattern 2)}$$

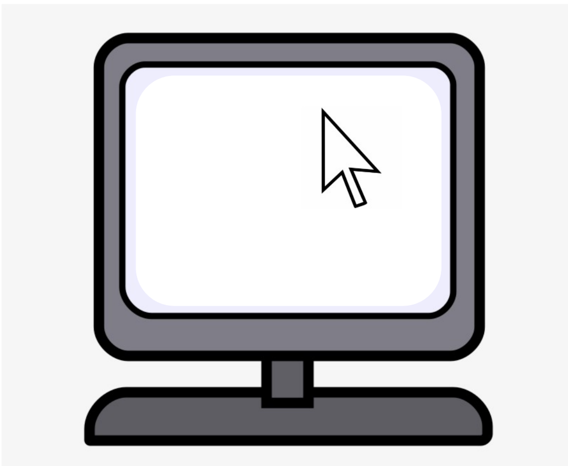
$$+ \ \frac{1}{2}(y_1 - a_1)^2 + \frac{1}{2}(y_2 - a_2)^2 + \ . \ . \ . + \frac{1}{2}(y_i - a_i)^2 + \ . \ . \ . \qquad \textit{(pattern 3)}$$

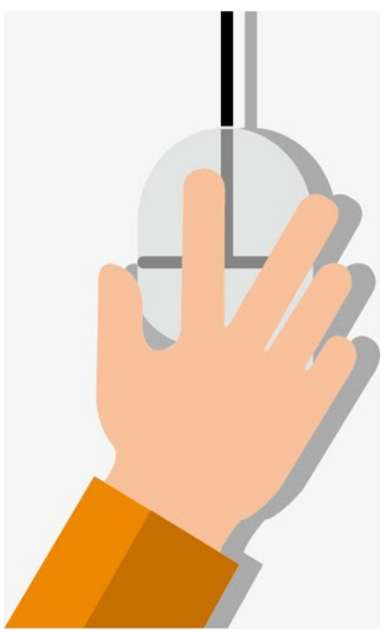
$$+ \ . \ . \ .$$

$$C = \frac{1}{n} \sum_p \sum_i \frac{1}{2}\left(y_i^{(p)} - a_i^{(p)}\right)^2$$

averaged over all *n* patterns

Variables





- *cursor*
- *mouse*
- *hand*

$$\frac{\partial\, mouse}{\partial\, hand}$$ Influence of *hand* on *mouse*

$$\frac{\partial\, cursor}{\partial\, mouse}$$ Influence of *mouse* on *cursor*

$$\frac{\partial\, cursor}{\partial\, hand}$$ Influence of *hand* on *cursor*

Chain Rule

$$\frac{\partial\, cursor}{\partial\, hand} = \frac{\partial\, mouse}{\partial\, hand} \times \frac{\partial\, cursor}{\partial\, mouse}$$

"Influence of *hand* on *cursor* can act through *mouse*"

Partial derivative of *y* with respect to *x*

Partial derivative of *F* with respect to *x*

$$\frac{\partial F}{\partial x} = \frac{\partial y}{\partial x} \times \frac{\partial F}{\partial y}$$

Partial derivative of *F* with respect to *y*

"Influence of *x* on function *F* can act through *y*"

Influence of Weight w_{ij} on Cost Function C

"cost gradient" $$\frac{\partial C}{\partial w_{ij}}$$ ← could be any weight in the network

$$\frac{\partial C}{\partial w_{ij}} > 0$$

How will changing w_{ij} cause C to change?

Influence of Weight w_{ij} on Cost Function C

"cost gradient" $$\frac{\partial C}{\partial w_{ij}}$$ ← could be any weight in the network

$$\frac{\partial C}{\partial w_{ij}} > 0$$

means that **increasing** w_{ij} makes C get **bigger**, and **decreasing** w_{ij} makes C get **smaller**
so we should **decrease** w_{ij} by some amount

$$\frac{\partial C}{\partial w_{ij}} < 0$$

How will changing w_{ij} cause C to change?

Influence of Weight w_{ij} on Cost Function C

"cost gradient" $$\frac{\partial C}{\partial w_{ij}}$$ ← could be any weight in the network

$$\frac{\partial C}{\partial w_{ij}} > 0$$

means that **increasing** w_{ij} makes C get **bigger**,
and **decreasing** w_{ij} makes C get **smaller**
so we should **decrease** w_{ij} by some amount

$$\frac{\partial C}{\partial w_{ij}} < 0$$

means that **increasing** w_{ij} makes C get **smaller**,
and **decreasing** w_{ij} makes C get **bigger**
so we should **increase** w_{ij} by some amount

How to Update the Weights?

"cost gradient" $\frac{\partial C}{\partial w_{ij}}$ ← could be any weight in the network

$$\Delta w_{ij} = -\eta \frac{\partial C}{\partial w_{ij}}$$

Δw_{ij}: amount to **change** the weight

η: "learning rate" $0 < \eta < 1$

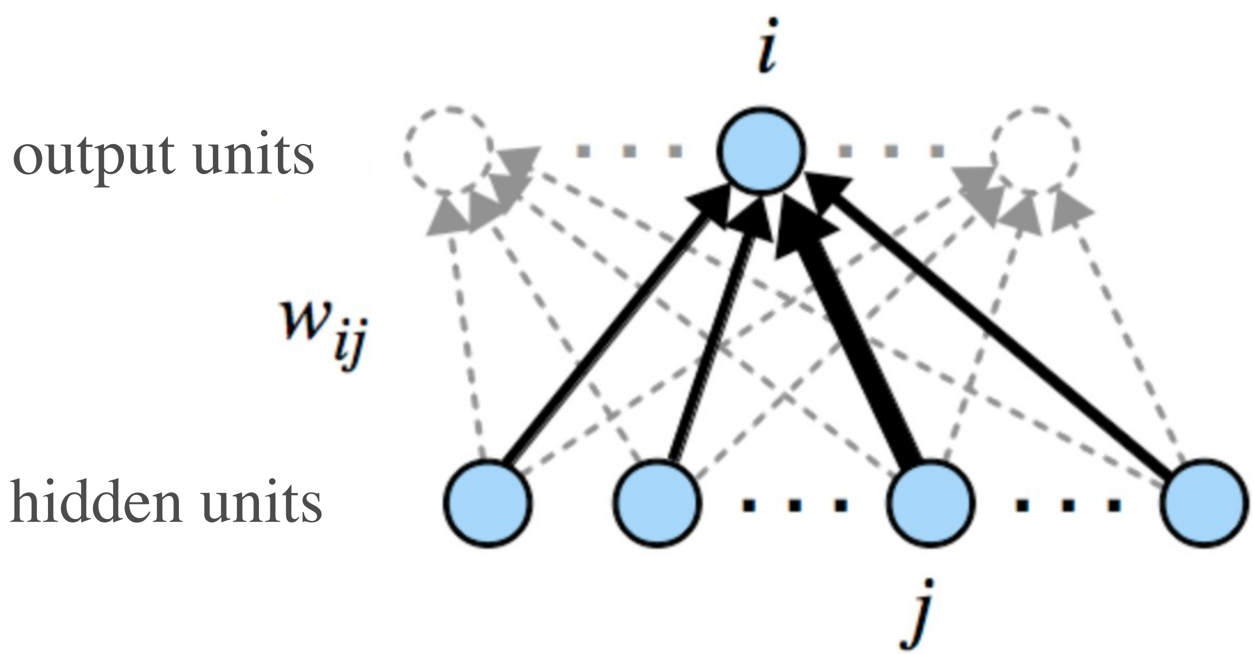
If the cost gradient is **positive**, we **decrease** the weight

If the cost gradient is **negative**, we **increase** the weight

Hidden → Output Weights

$$\frac{\partial C}{\partial w_{ij}} = \frac{\partial z_i}{\partial w_{ij}} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

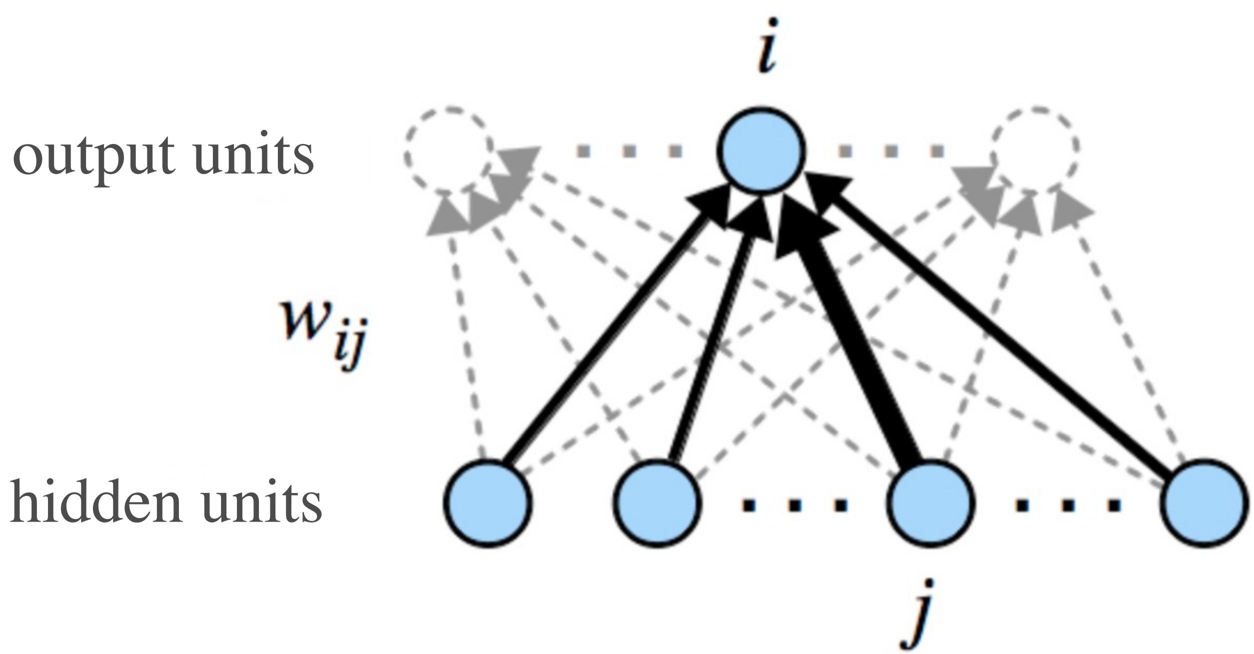
$$\boxed{\begin{array}{c}\text{influence of}\\ w_{ij} \text{ on } C\end{array}} = \boxed{\begin{array}{c}\text{influence of}\\ w_{ij} \text{ on } z_i\end{array}} \times \boxed{\begin{array}{c}\text{influence of}\\ z_i \text{ on } a_i\end{array}} \times \boxed{\begin{array}{c}\text{influence of}\\ a_i \text{ on } C\end{array}}$$



Hidden → Output Weights

$$\frac{\partial C}{\partial w_{ij}} = \frac{\partial z_i}{\partial w_{ij}} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

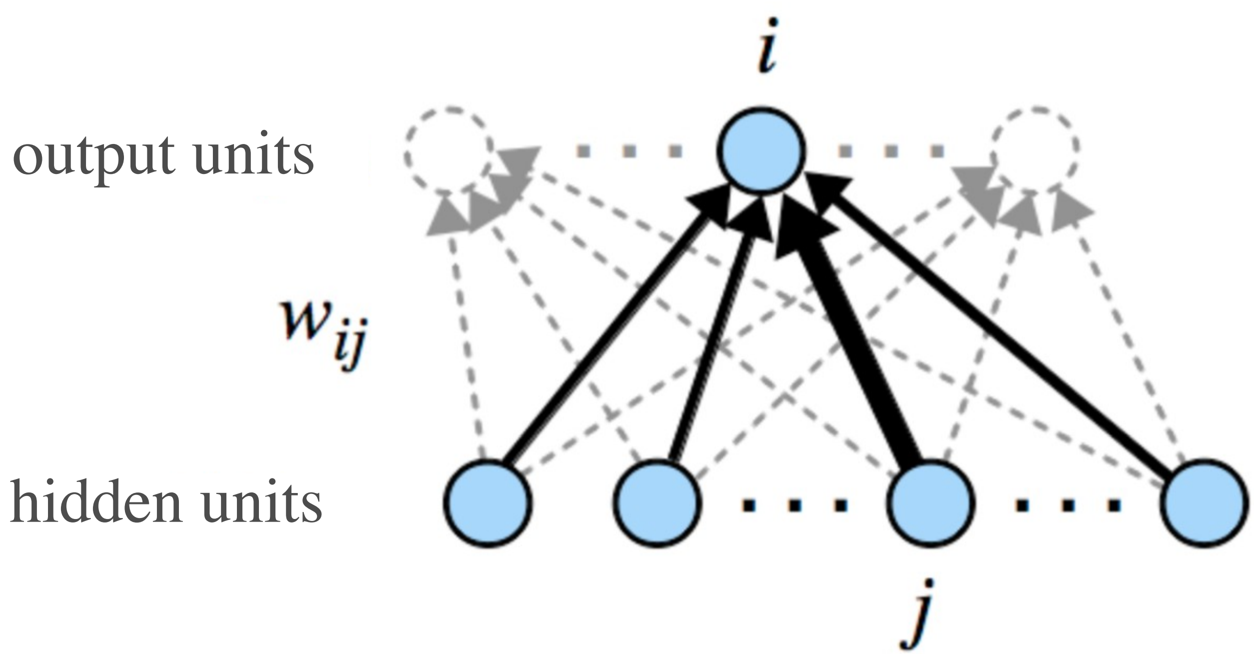
$$\boxed{\text{influence of } w_{ij} \text{ on } C} = a_j \times a_i(1 - a_i) \times (a_i - y_i)$$



Hidden → Output Weights

$$\frac{\partial C}{\partial w_{ij}} = \frac{\partial z_i}{\partial w_{ij}} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

$$\frac{\partial C}{\partial w_{ij}} = (a_i - y_i)\, a_i (1 - a_i)\, a_j$$



Hidden → Output Weights

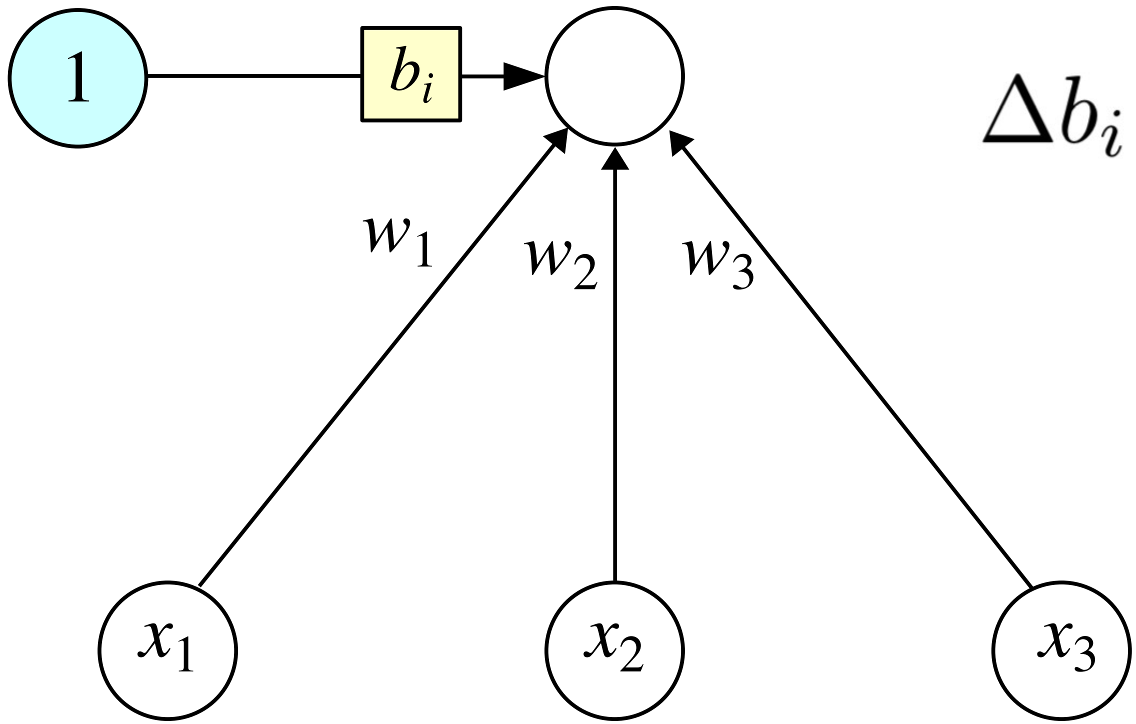
$$\frac{\partial C}{\partial w_{ij}} = \frac{\partial z_i}{\partial w_{ij}} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

$$\frac{\partial C}{\partial w_{ij}} = \boxed{(a_i - y_i)\, a_i (1 - a_i)}\, a_j$$

call this quantity δ_i

$$\Delta w_{ij} = -\eta \frac{\partial C}{\partial w_{ij}} = -\eta\, \delta_i\, a_j$$

How to Update the Bias?



$$\Delta w_{ij} = -\eta \, \delta_i \, a_j$$

$$\Delta b_i = -\eta \, \delta_i$$

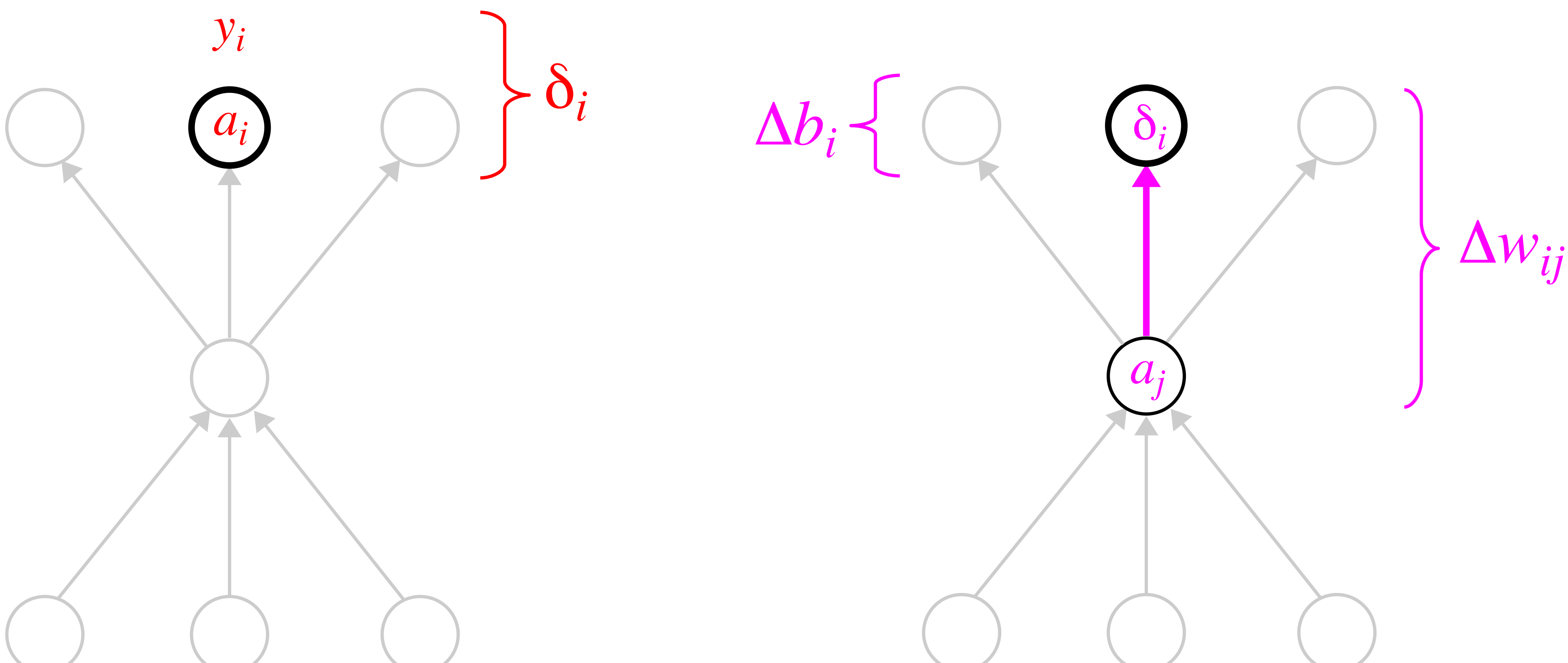
$$z_i = w_1 x_1 + w_2 x_2 + w_3 x_3 + b_i \cdot 1$$

Update Rule for Output Unit i

$0 < \eta < 1$

$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

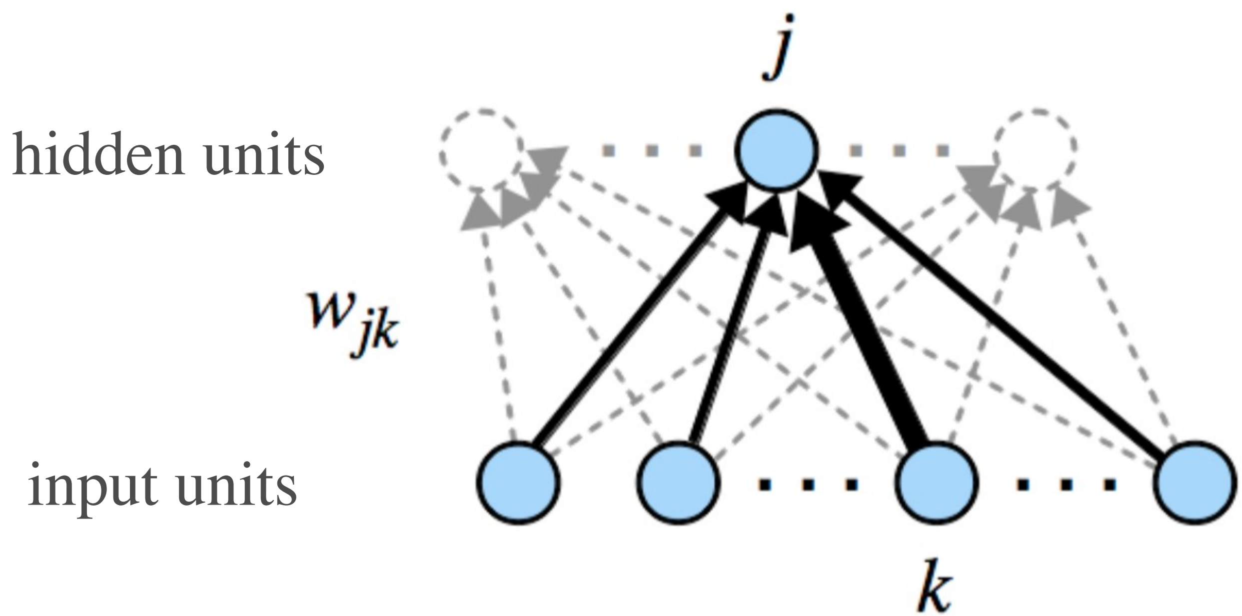
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$



Input → Hidden Weights

$$\frac{\partial C}{\partial w_{jk}} = \frac{\partial z_j}{\partial w_{jk}} \times \frac{\partial a_j}{\partial z_j} \times \frac{\partial C}{\partial a_j}$$

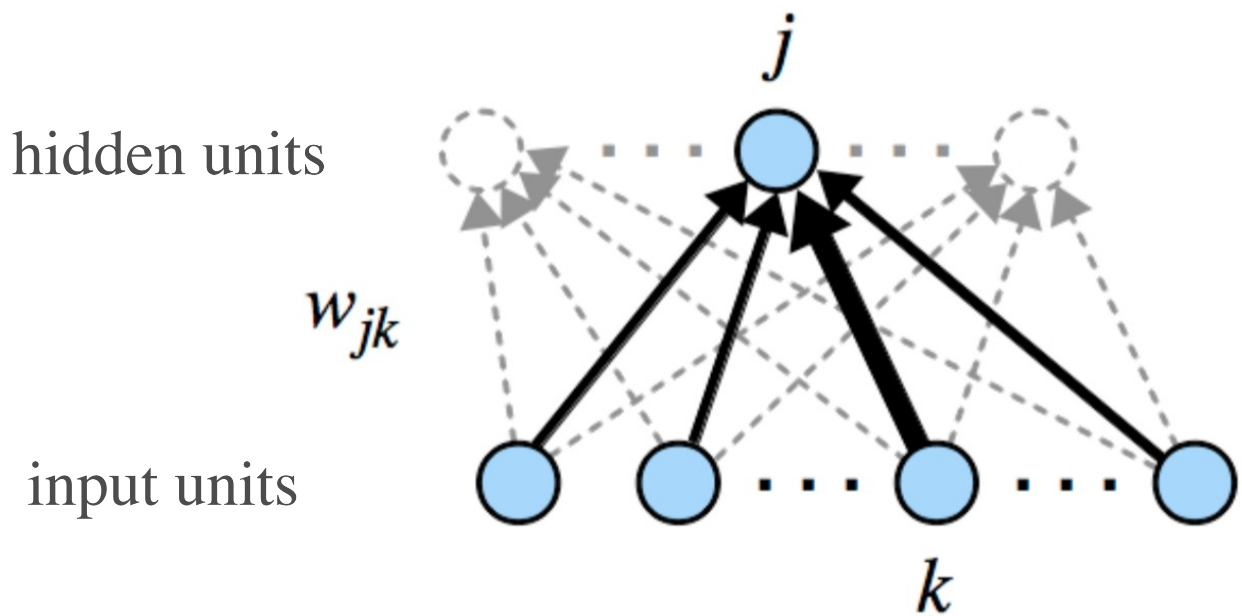
$$\boxed{\text{influence of } w_{jk} \text{ on } C} = \boxed{\text{influence of } w_{jk} \text{ on } z_j} \times \boxed{\text{influence of } z_j \text{ on } a_j} \times \boxed{\text{influence of } a_j \text{ on } C}$$



Input → Hidden Weights

$$\frac{\partial C}{\partial w_{jk}} = \frac{\partial z_j}{\partial w_{jk}} \times \frac{\partial a_j}{\partial z_j} \times \frac{\partial C}{\partial a_j}$$

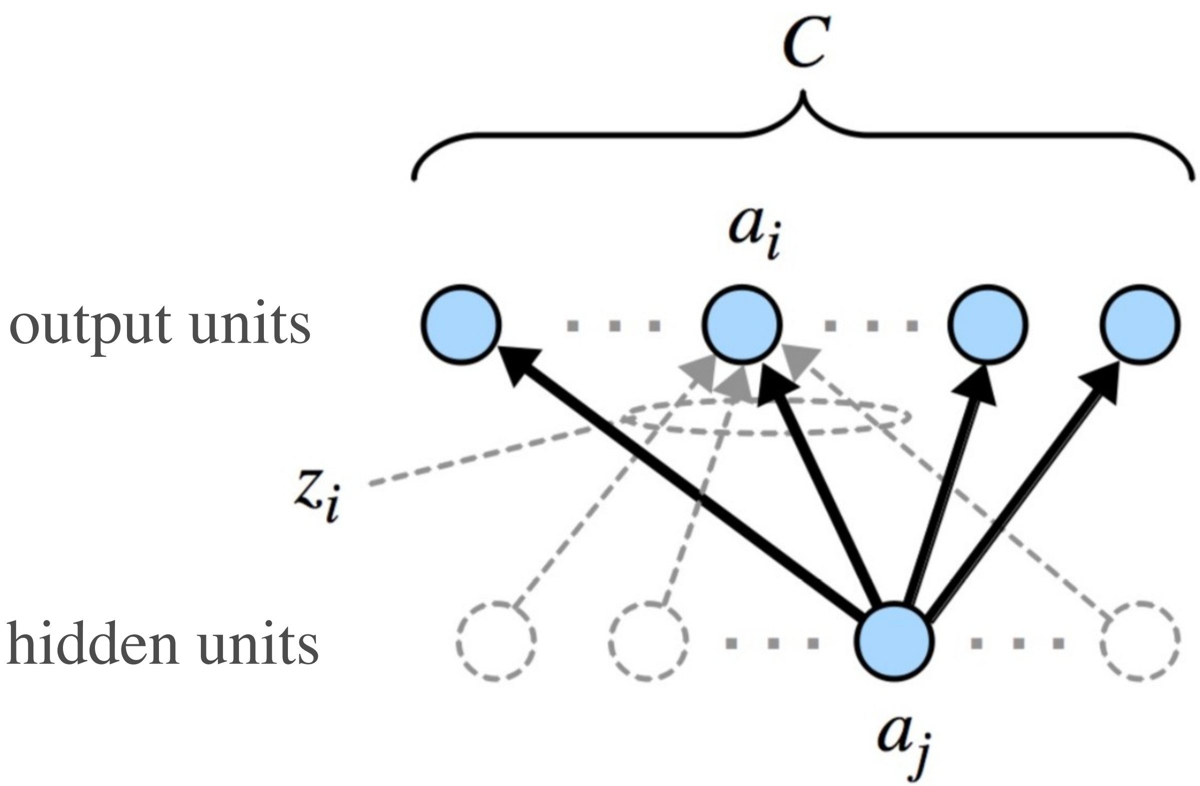
$$\boxed{\begin{array}{c}\text{influence of}\\ w_{jk} \text{ on } C\end{array}} = x_k \times a_j(1-a_j) \times \text{???}$$



Influence of Hidden Unit j on Cost Function

$$\frac{\partial C}{\partial a_j} = \sum_i \frac{\partial z_i}{\partial a_j} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

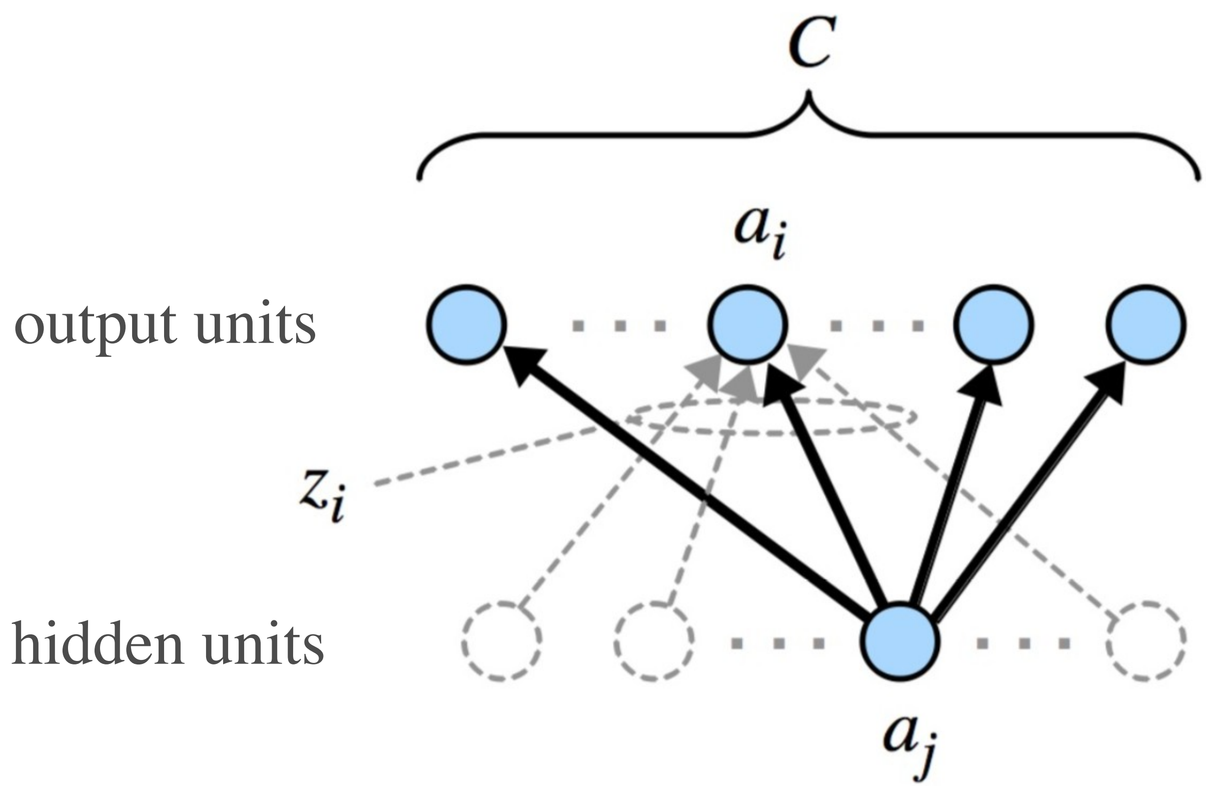
$$= \sum_i \boxed{\begin{array}{c}\text{influence of}\\ a_j \text{ on } z_i\end{array}} \times \boxed{\begin{array}{c}\text{influence of}\\ z_i \text{ on } a_i\end{array}} \times \boxed{\begin{array}{c}\text{influence of}\\ a_i \text{ on } C\end{array}}$$



Influence of Hidden Unit j on Cost Function

$$\frac{\partial C}{\partial a_j} = \sum_i \frac{\partial z_i}{\partial a_j} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

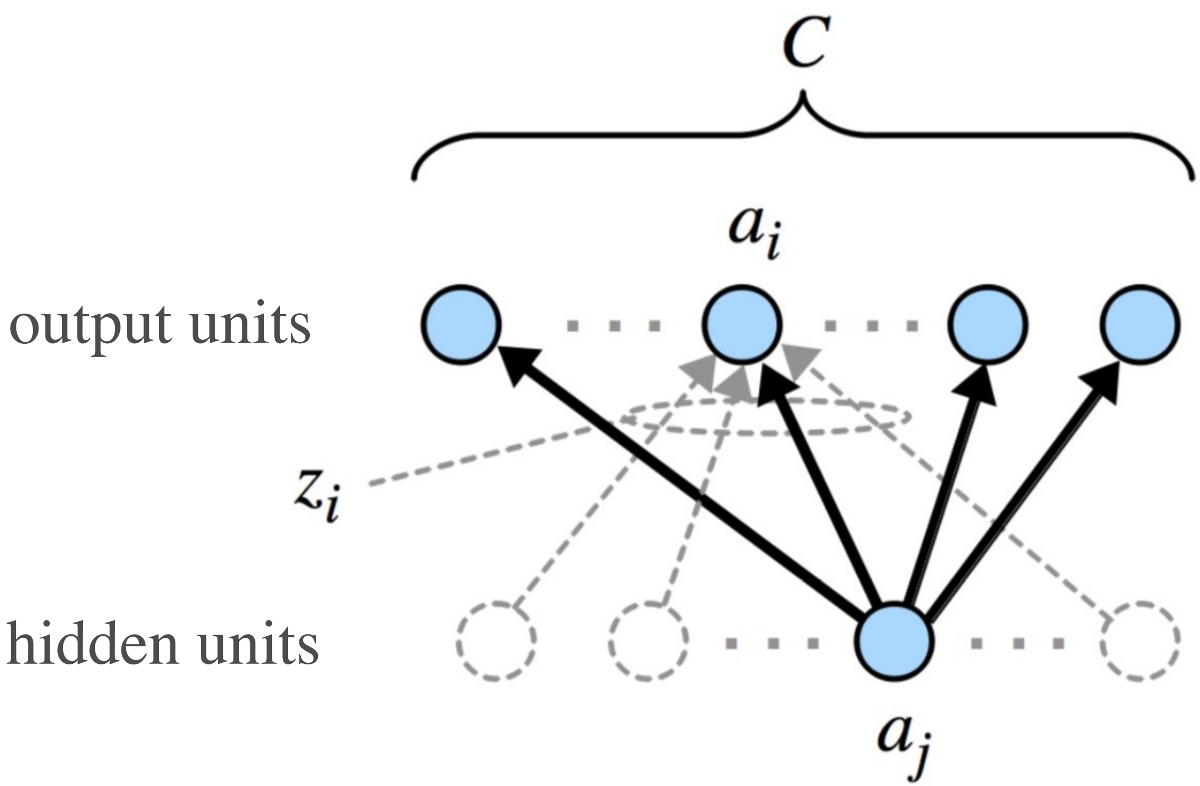
$$= \sum_i w_{ij} \times a_i(1-a_i) \times (a_i - y_i)$$



Influence of Hidden Unit j on Cost Function

$$\frac{\partial C}{\partial a_j} = \sum_i \frac{\partial z_i}{\partial a_j} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

$$= \sum_i w_{ij} \left(a_i - y_i\right) a_i \left(1 - a_i\right)$$



Influence of Hidden Unit j on Cost Function

$$\frac{\partial C}{\partial a_j} = \sum_i \frac{\partial z_i}{\partial a_j} \times \frac{\partial a_i}{\partial z_i} \times \frac{\partial C}{\partial a_i}$$

$$= \sum_i w_{ij} \boxed{(a_i - y_i)\, a_i\, (1 - a_i)}$$

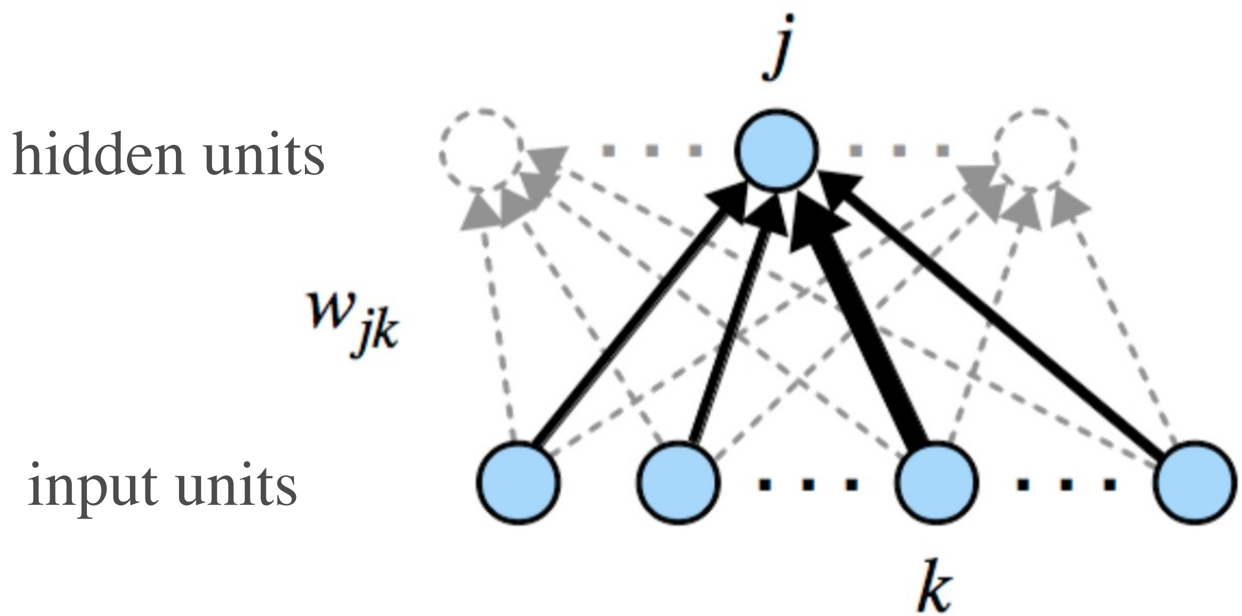
this is δ_i

$$= \sum_i w_{ij}\, \delta_i$$

Input → Hidden Weights

$$\frac{\partial C}{\partial w_{jk}} = \frac{\partial z_j}{\partial w_{jk}} \times \frac{\partial a_j}{\partial z_j} \times \frac{\partial C}{\partial a_j}$$

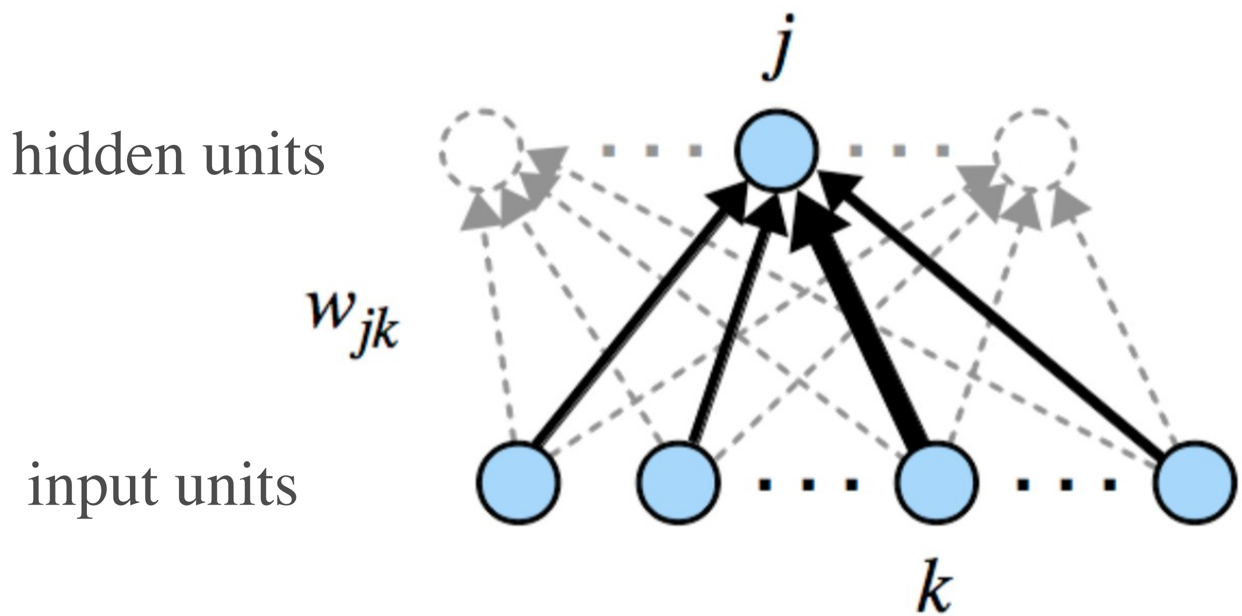
$$\boxed{\begin{array}{c}\text{influence of}\\ w_{jk} \text{ on } C\end{array}} = x_k \times a_j(1 - a_j) \times ???$$



Input → Hidden Weights

$$\frac{\partial C}{\partial w_{jk}} = \frac{\partial z_j}{\partial w_{jk}} \times \frac{\partial a_j}{\partial z_j} \times \frac{\partial C}{\partial a_j}$$

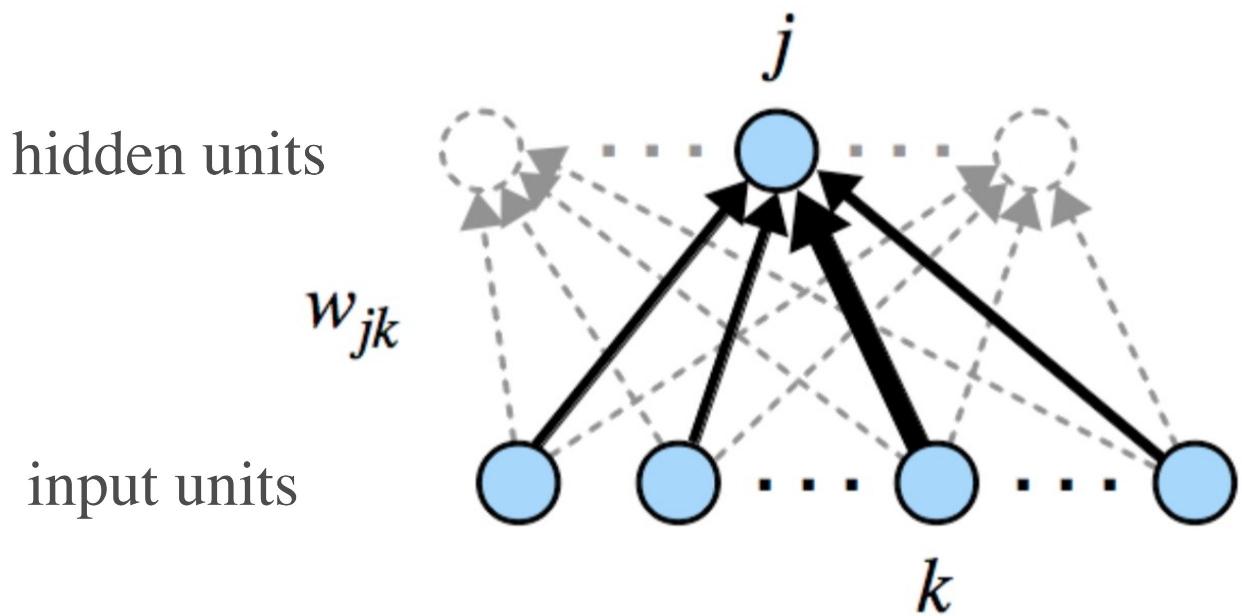
$$\boxed{\begin{array}{c}\text{influence of}\\ w_{jk} \text{ on } C\end{array}} = x_k \times a_j(1-a_j) \times \sum_i w_{ij}\,\delta_i$$



Input → Hidden Weights

$$\frac{\partial C}{\partial w_{jk}} = \frac{\partial z_j}{\partial w_{jk}} \times \frac{\partial a_j}{\partial z_j} \times \frac{\partial C}{\partial a_j}$$

$$\frac{\partial C}{\partial w_{jk}} = \left(\sum_i w_{ij}\, \delta_i \right) a_j\,(1 - a_j)\, x_k$$



Input → Hidden Weights

$$\frac{\partial C}{\partial w_{jk}} = \frac{\partial z_j}{\partial w_{jk}} \times \frac{\partial a_j}{\partial z_j} \times \frac{\partial C}{\partial a_j}$$

$$\frac{\partial C}{\partial w_{jk}} = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)\, x_k$$

call this quantity δ_j

$$\frac{\partial C}{\partial w_{jk}} = \delta_j\, x_k$$

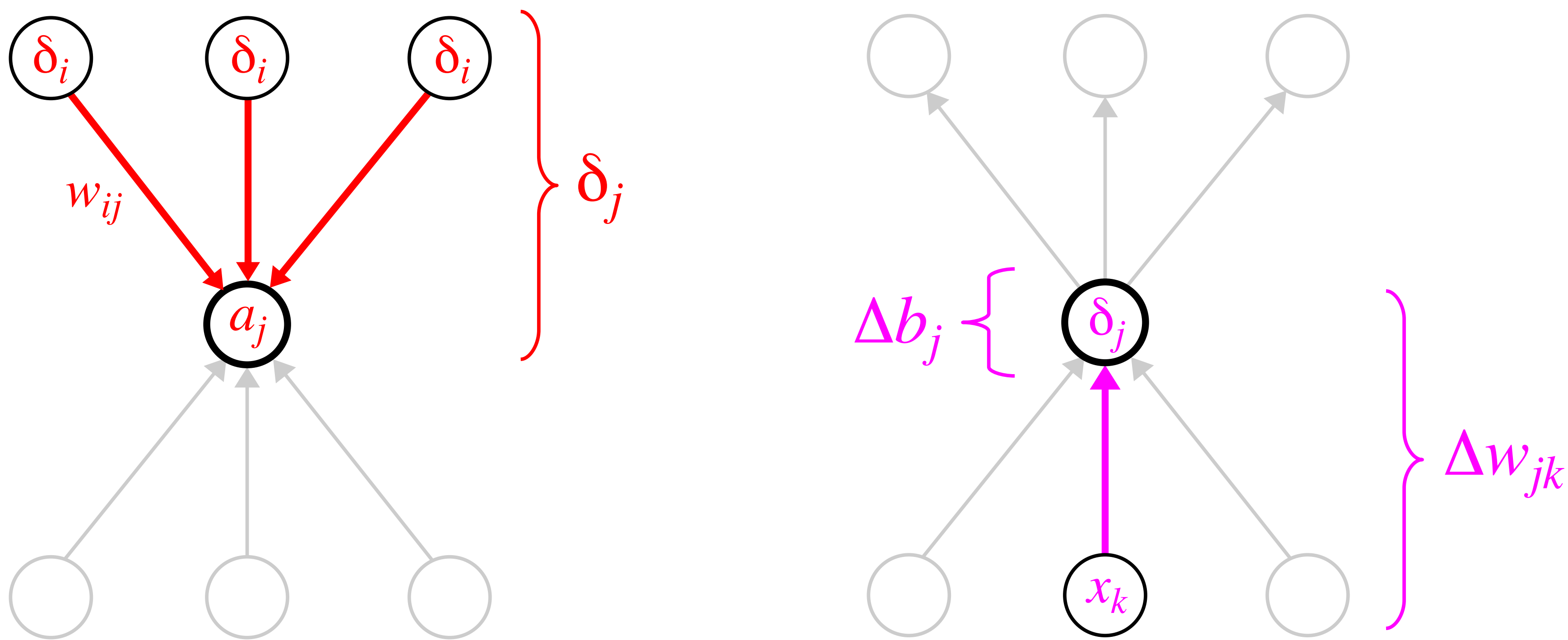
$$\Delta w_{jk} = -\eta\, \frac{\partial C}{\partial w_{jk}} = -\eta\, \delta_j\, x_k$$

Update Rule for Hidden Unit j

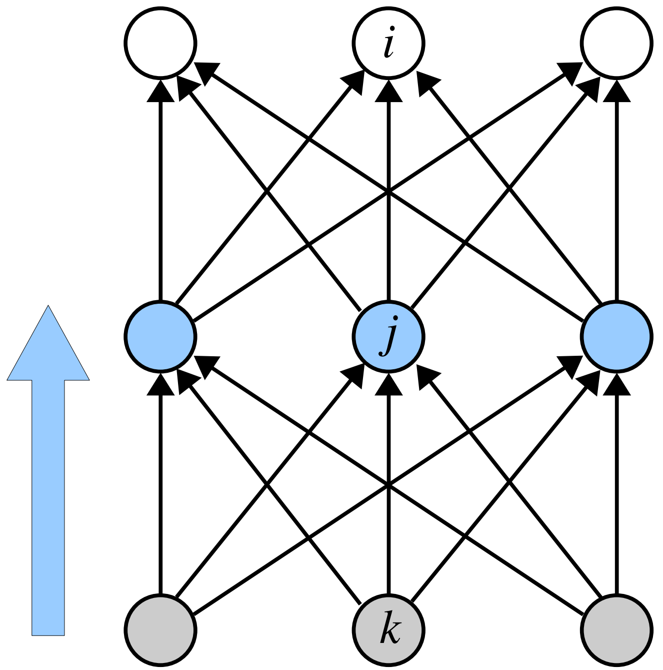
$0 < \eta < 1$

$$\delta_j = \left(\sum_i w_{ij}\,\delta_i\right) a_j\,(1 - a_j)$$

$$\Delta w_{jk} = -\eta\,\delta_j\,x_k \qquad\qquad \Delta b_j = -\eta\,\delta_j$$



Forward Pass

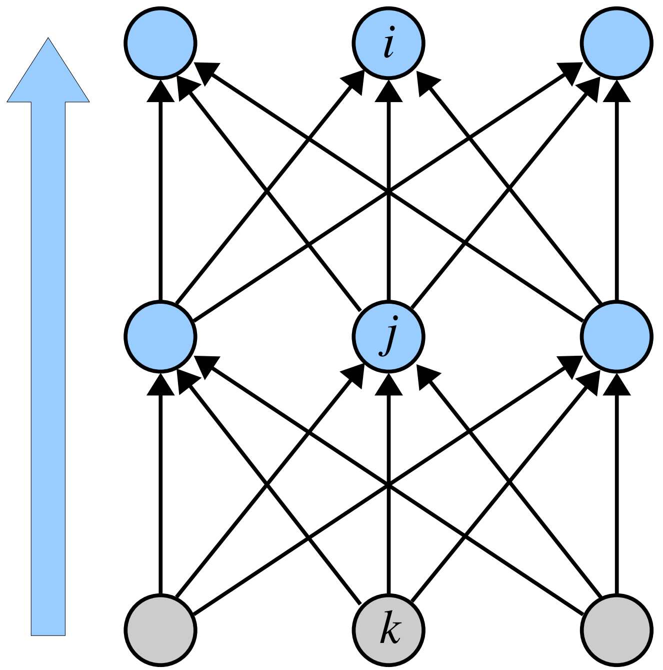


$$z_i = \left(\sum_j w_{ij}\, a_j \right) + b_i \qquad a_i = \sigma(z_i)$$

$$z_j = \left(\sum_k w_{jk}\, x_k \right) + b_j \qquad a_j = \sigma(z_j)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Forward Pass

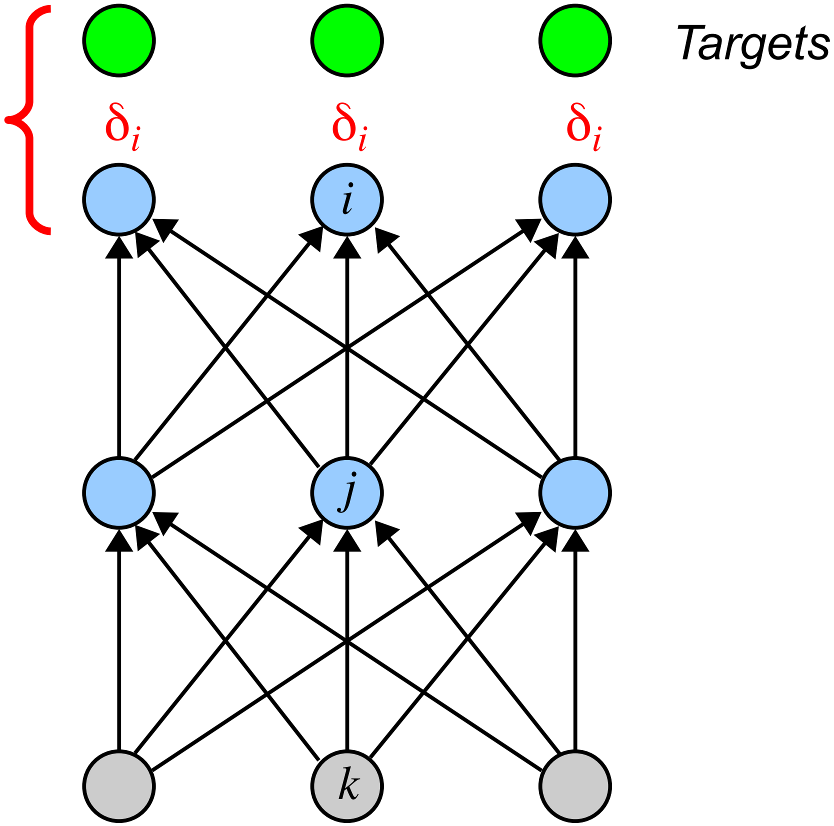


$$z_i = \left(\sum_j w_{ij}\, a_j \right) + b_i \qquad a_i = \sigma(z_i)$$

$$z_j = \left(\sum_k w_{jk}\, x_k \right) + b_j \qquad a_j = \sigma(z_j)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

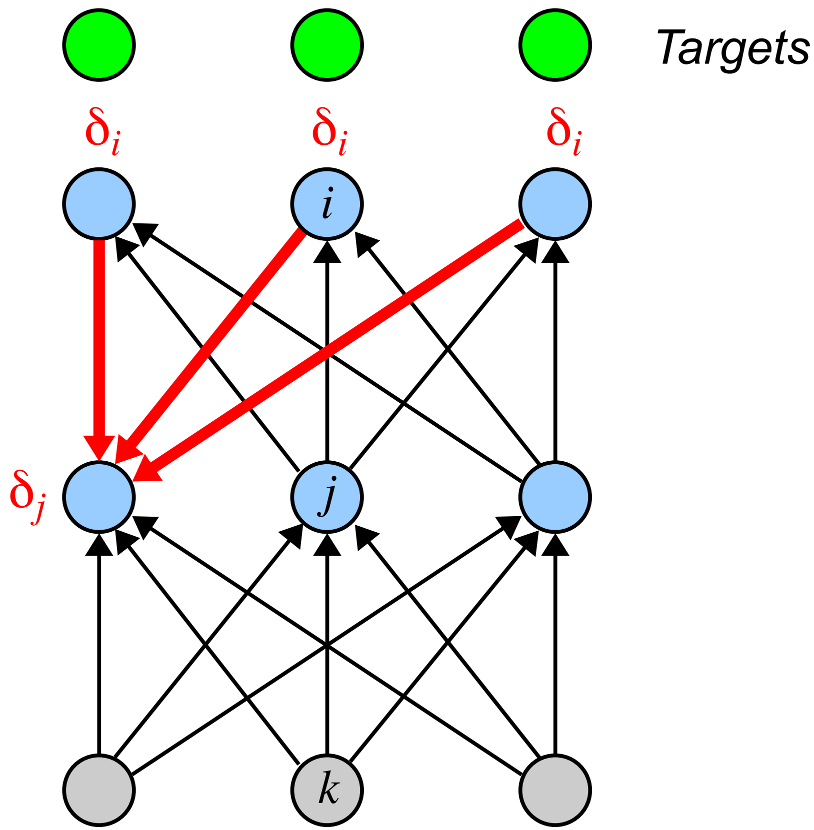
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

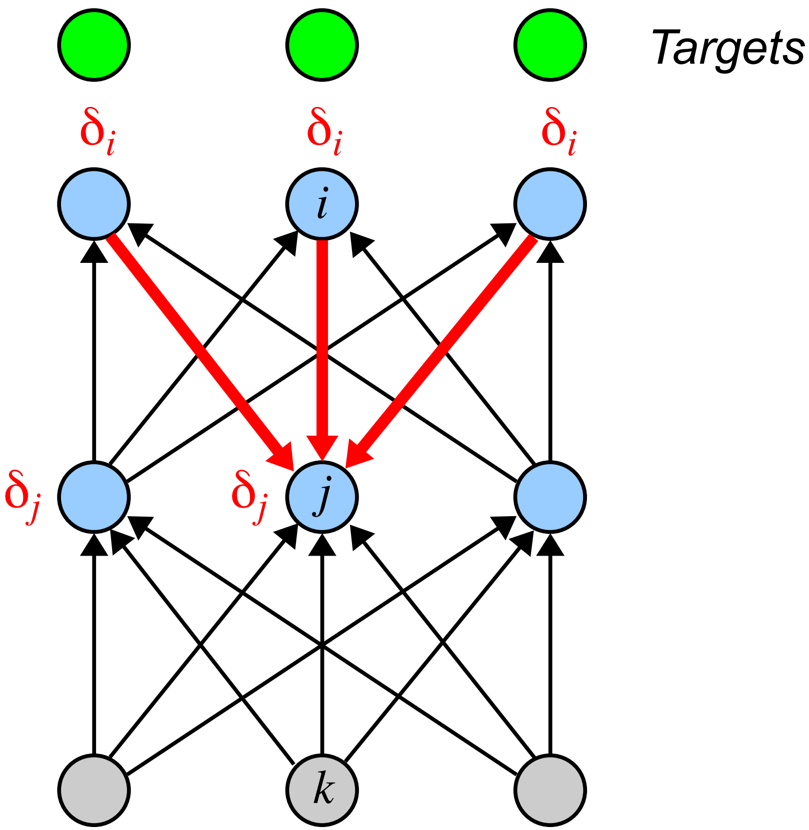
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

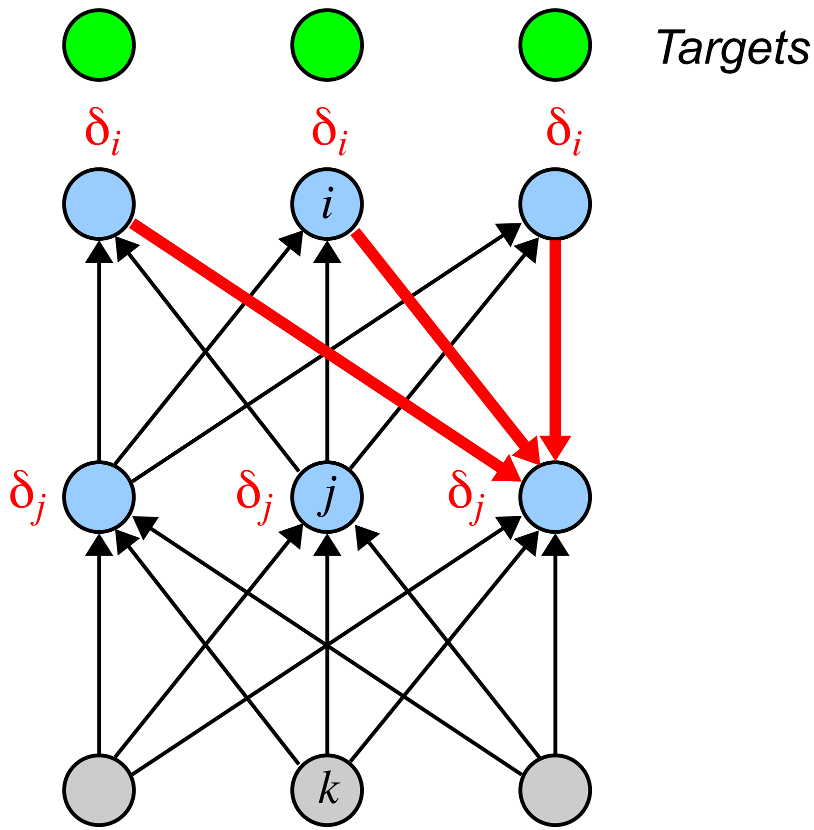
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

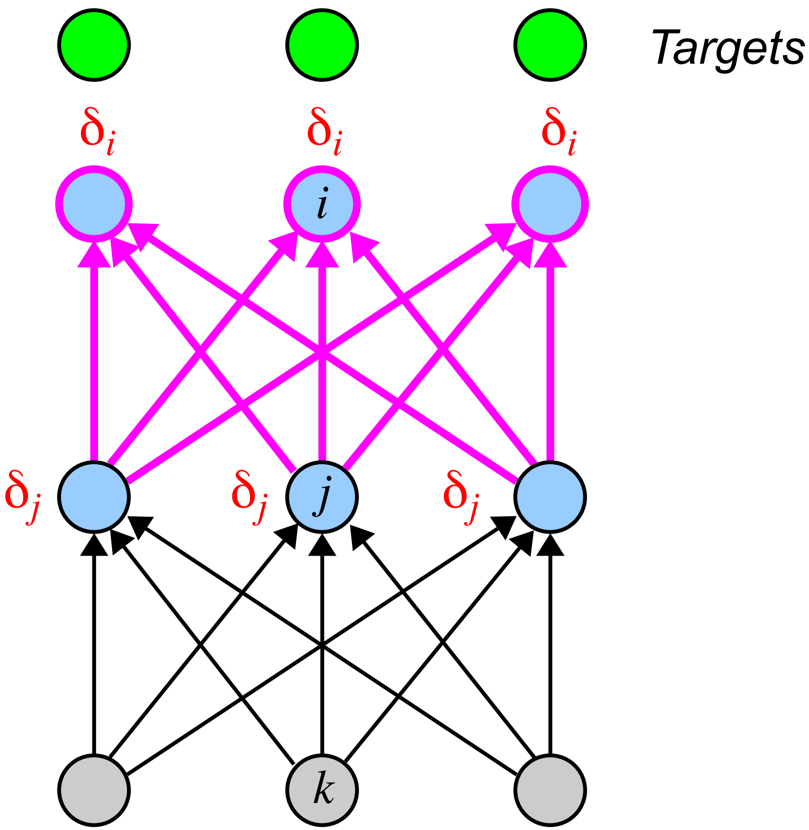
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

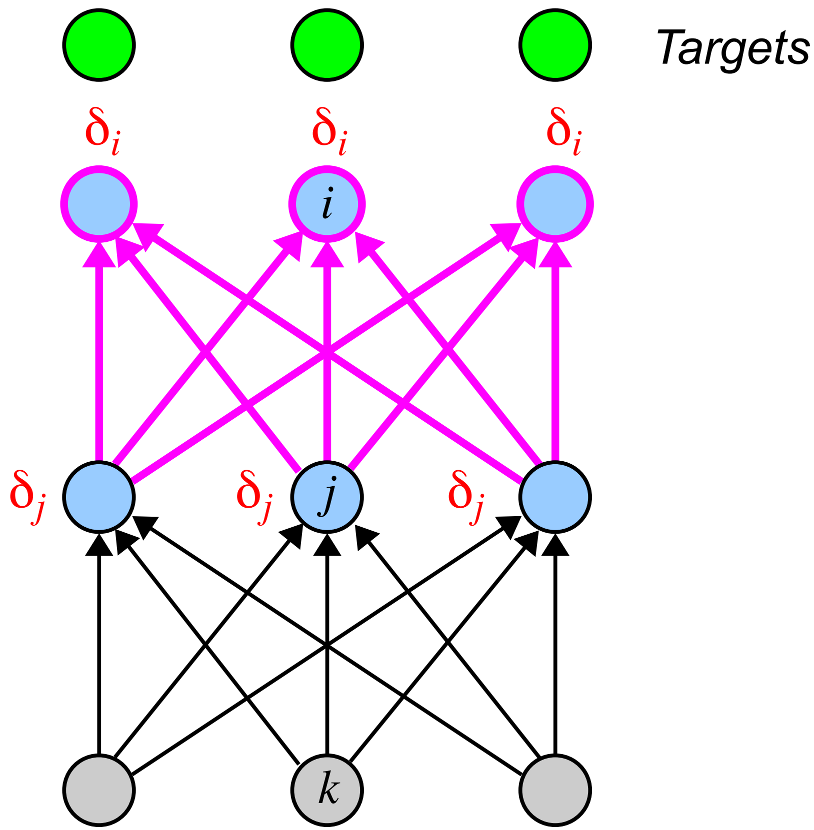
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

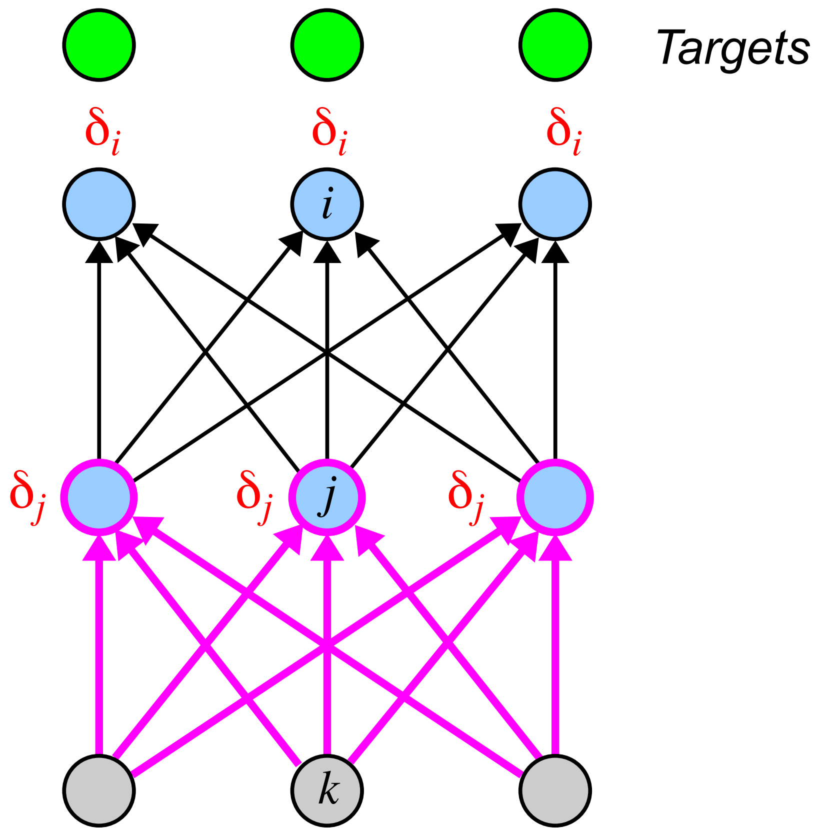
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

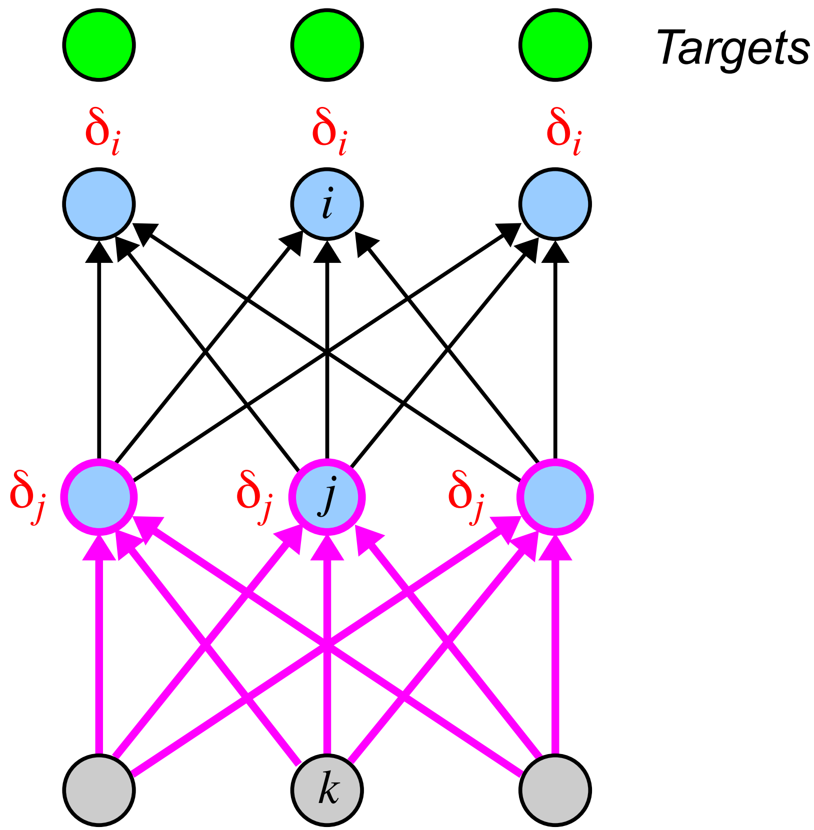
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass



$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

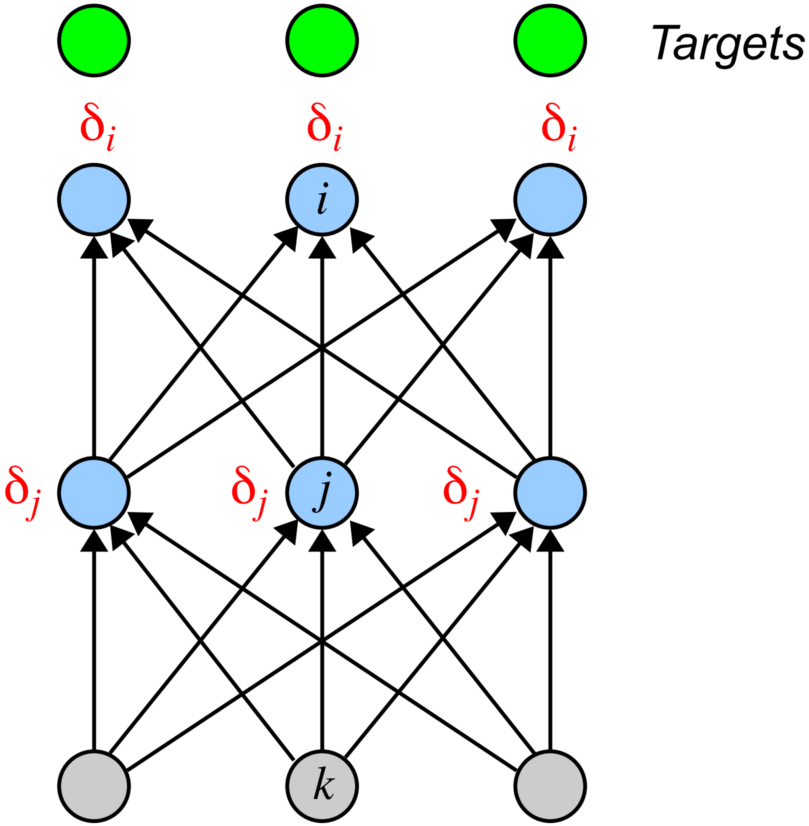
$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$

$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Backward Pass

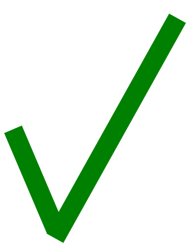


$$\delta_i = (a_i - y_i)\, a_i\, (1 - a_i)$$

$$\delta_j = \left(\sum_i w_{ij}\, \delta_i \right) a_j\, (1 - a_j)$$

$$\Delta w_{ij} = -\eta\, \delta_i\, a_j \qquad \Delta b_i = -\eta\, \delta_i$$

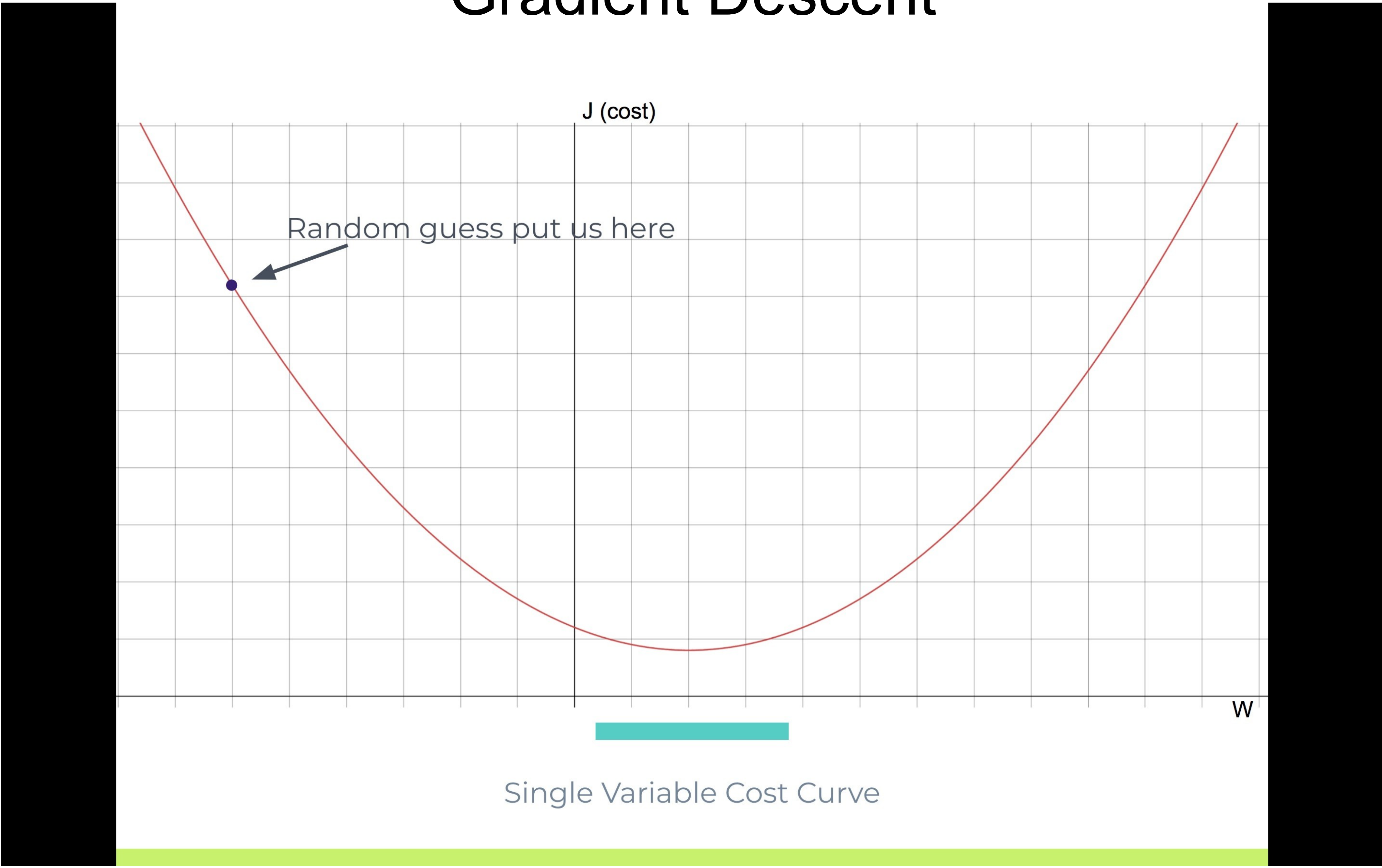
$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad b_i = b_i + \Delta b_i$$



$$\Delta w_{jk} = -\eta\, \delta_j\, x_k \qquad \Delta b_j = -\eta\, \delta_j$$

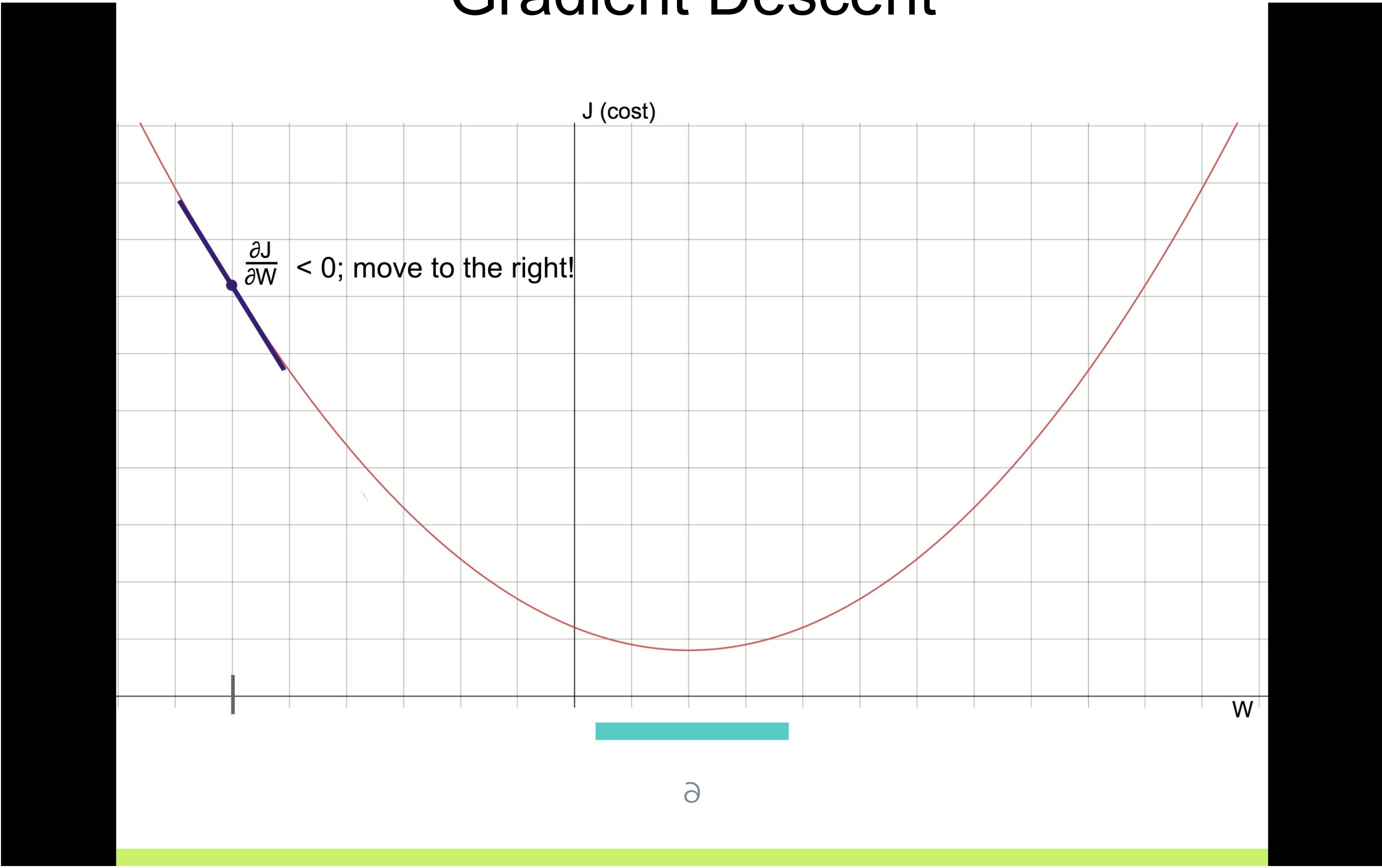
$$w_{jk} = w_{jk} + \Delta w_{jk} \qquad b_j = b_j + \Delta b_j$$

Gradient Descent



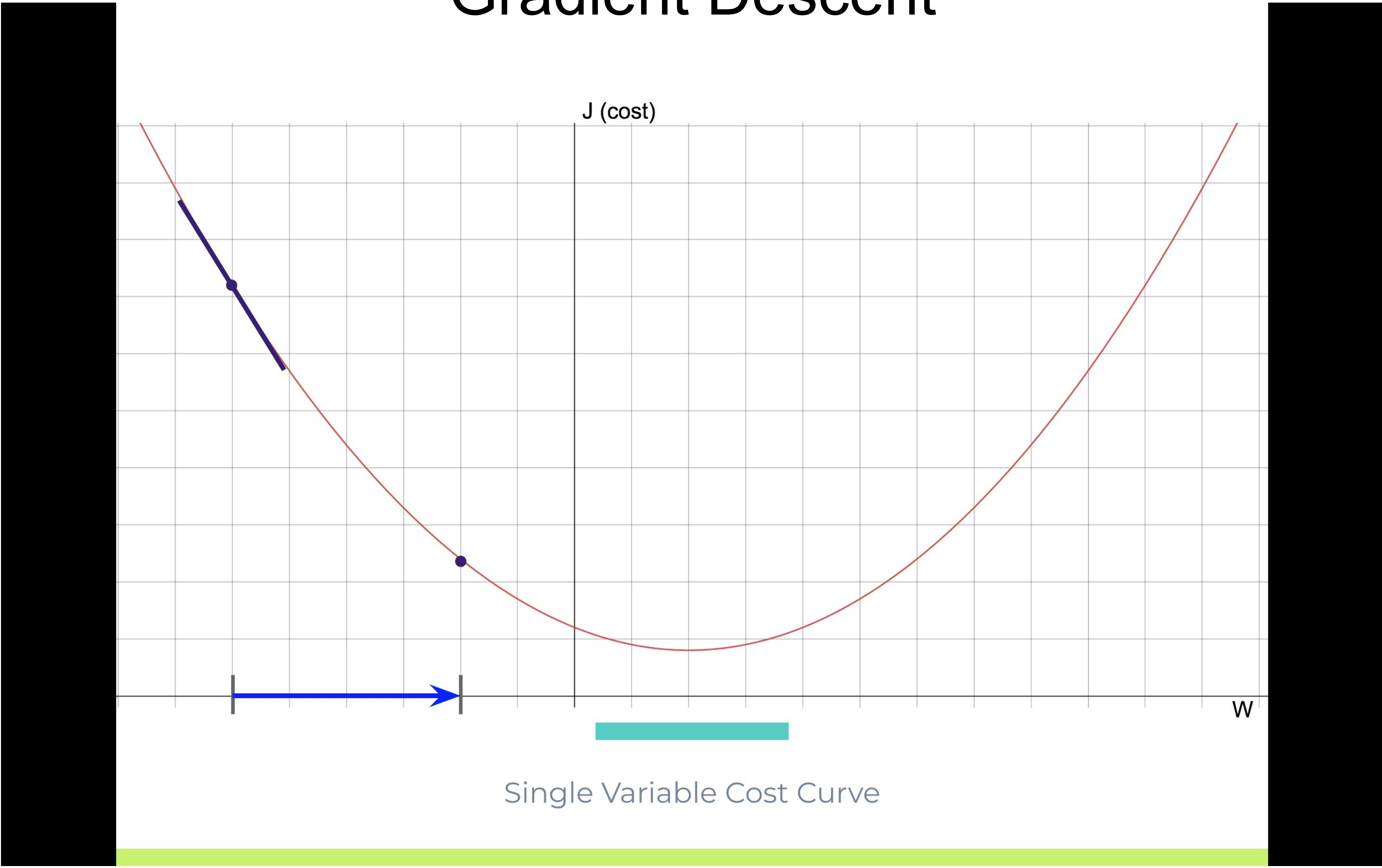
Source: Sam Abrahams (https://goo.gl/tKOvr7)

Gradient Descent



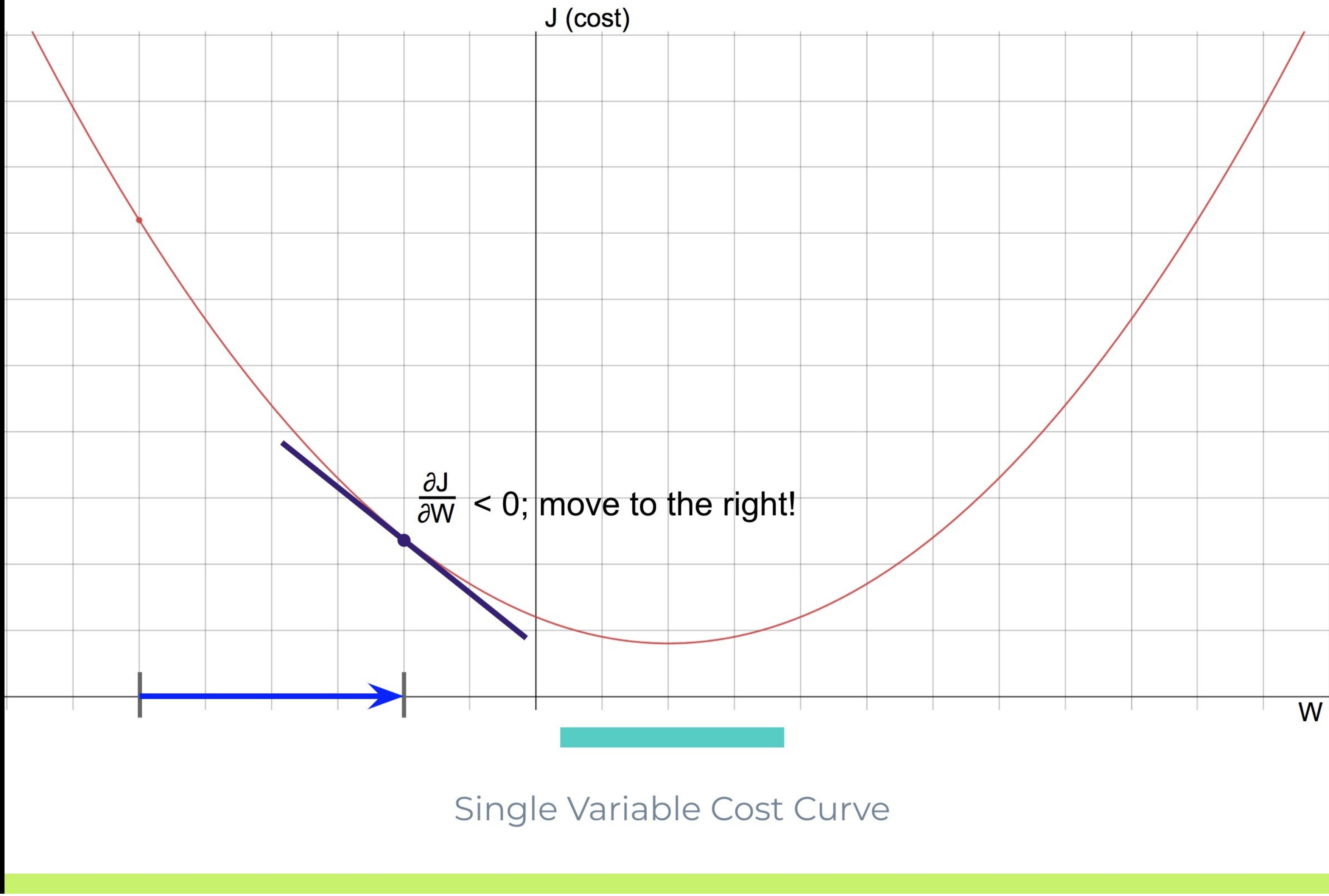
Source: Sam Abrahams (https://goo.gl/tKOvr7)

Gradient Descent



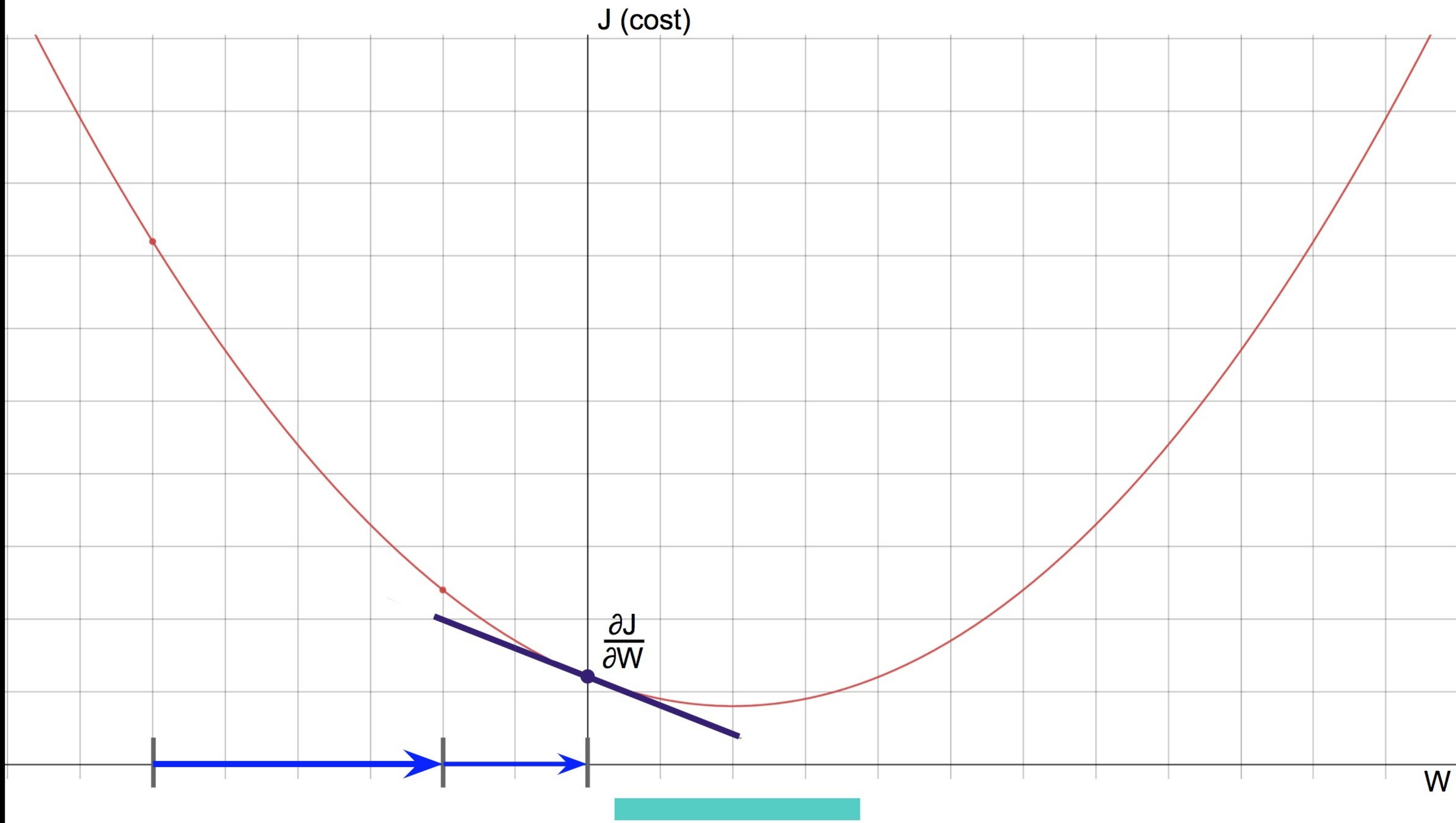
Source: Sam Abrahams (https://goo.gl/tKOvr7)

Gradient Descent



Source: Sam Abrahams (https://goo.gl/tKOvr7)

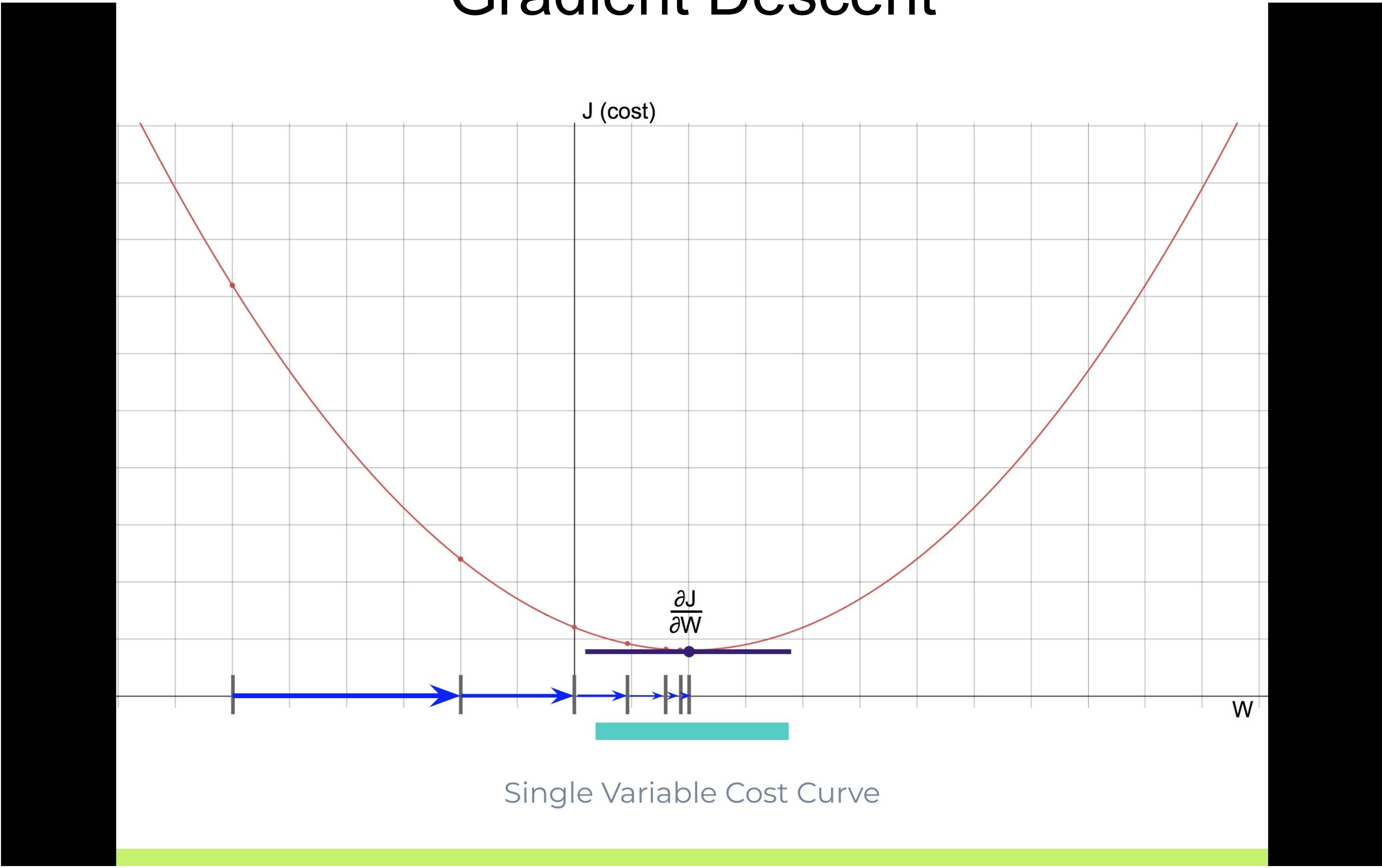
Gradient Descent



Single Variable Cost Curve

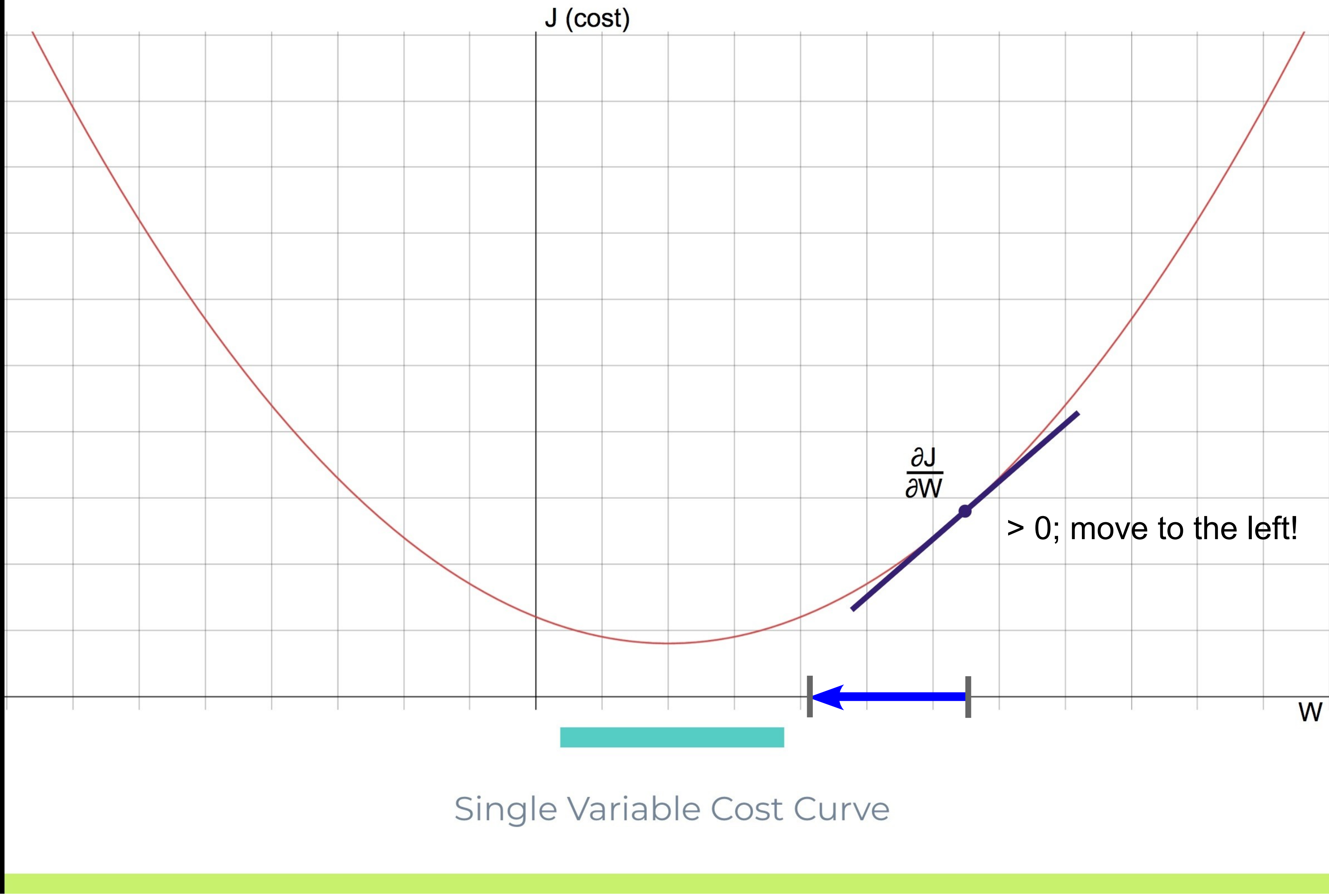
Source: Sam Abrahams (https://goo.gl/tKOvr7)

Gradient Descent



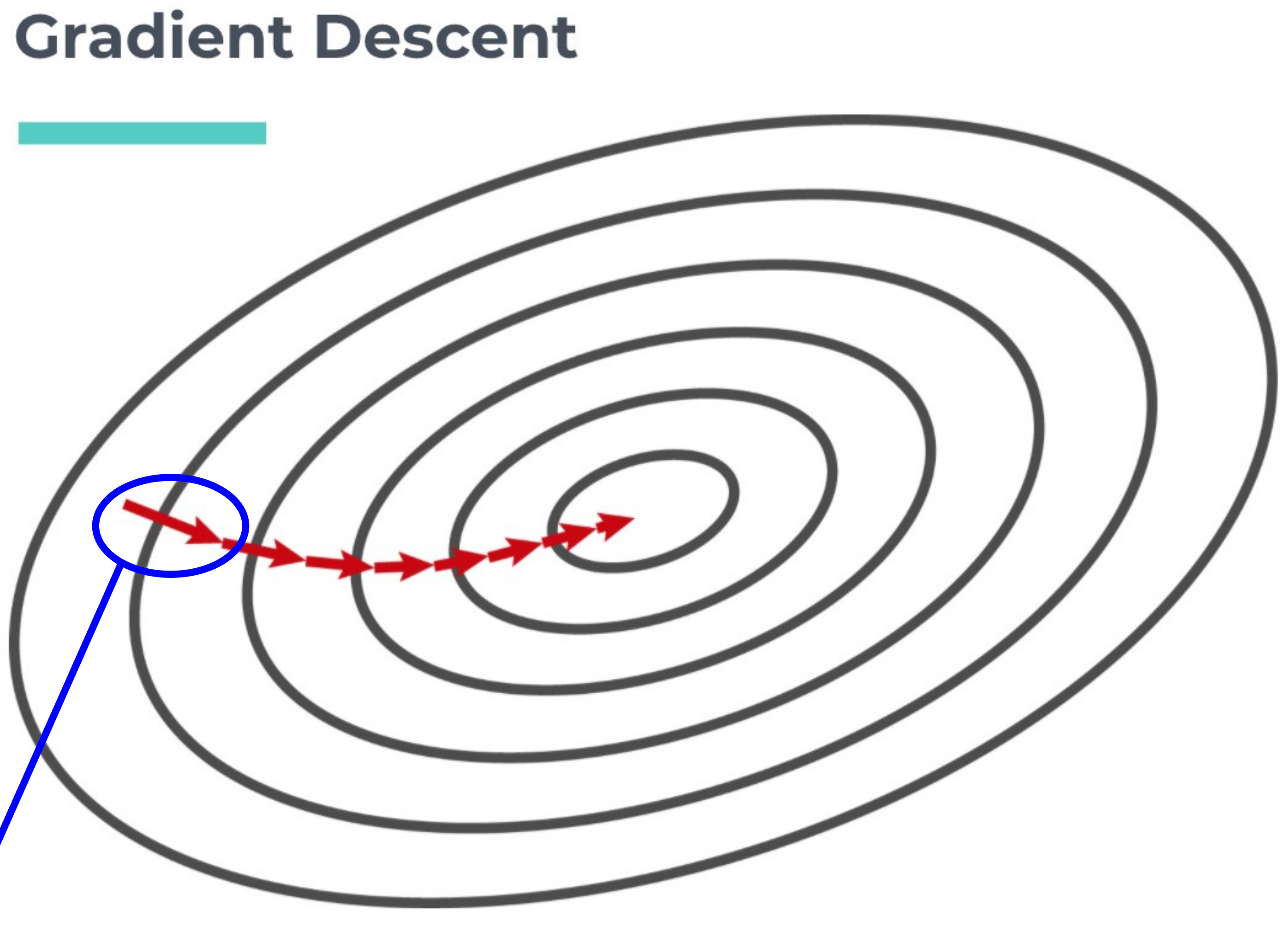
Source: Sam Abrahams (https://goo.gl/tKOvr7)

Gradient Descent



Source: Sam Abrahams (https://goo.gl/tKOvr7)

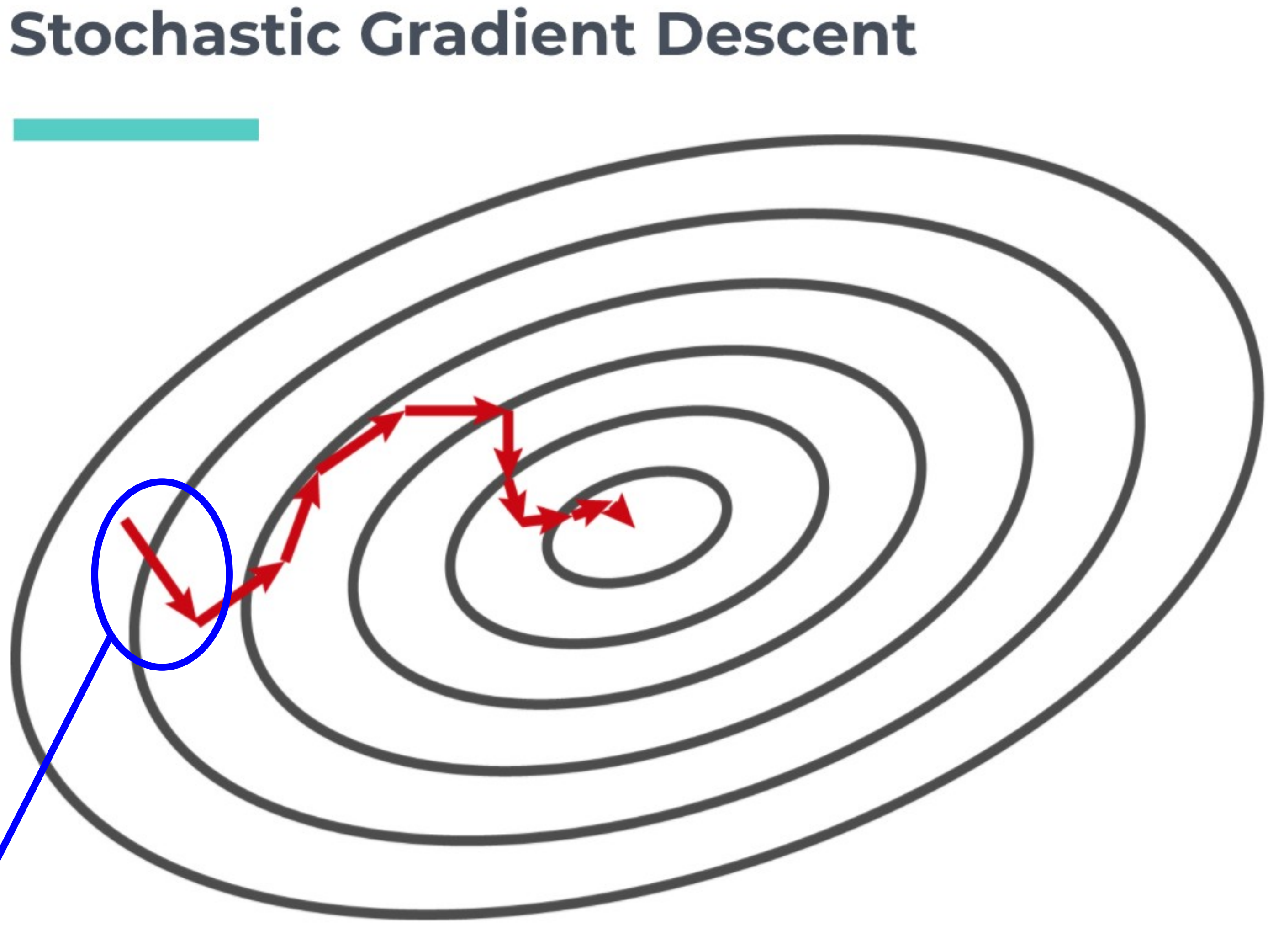
Weight Space Trajectory



Each update is based on **all *N* patterns** in the dataset

Source: Sam Abrahams (https://goo.gl/tKOvr7)

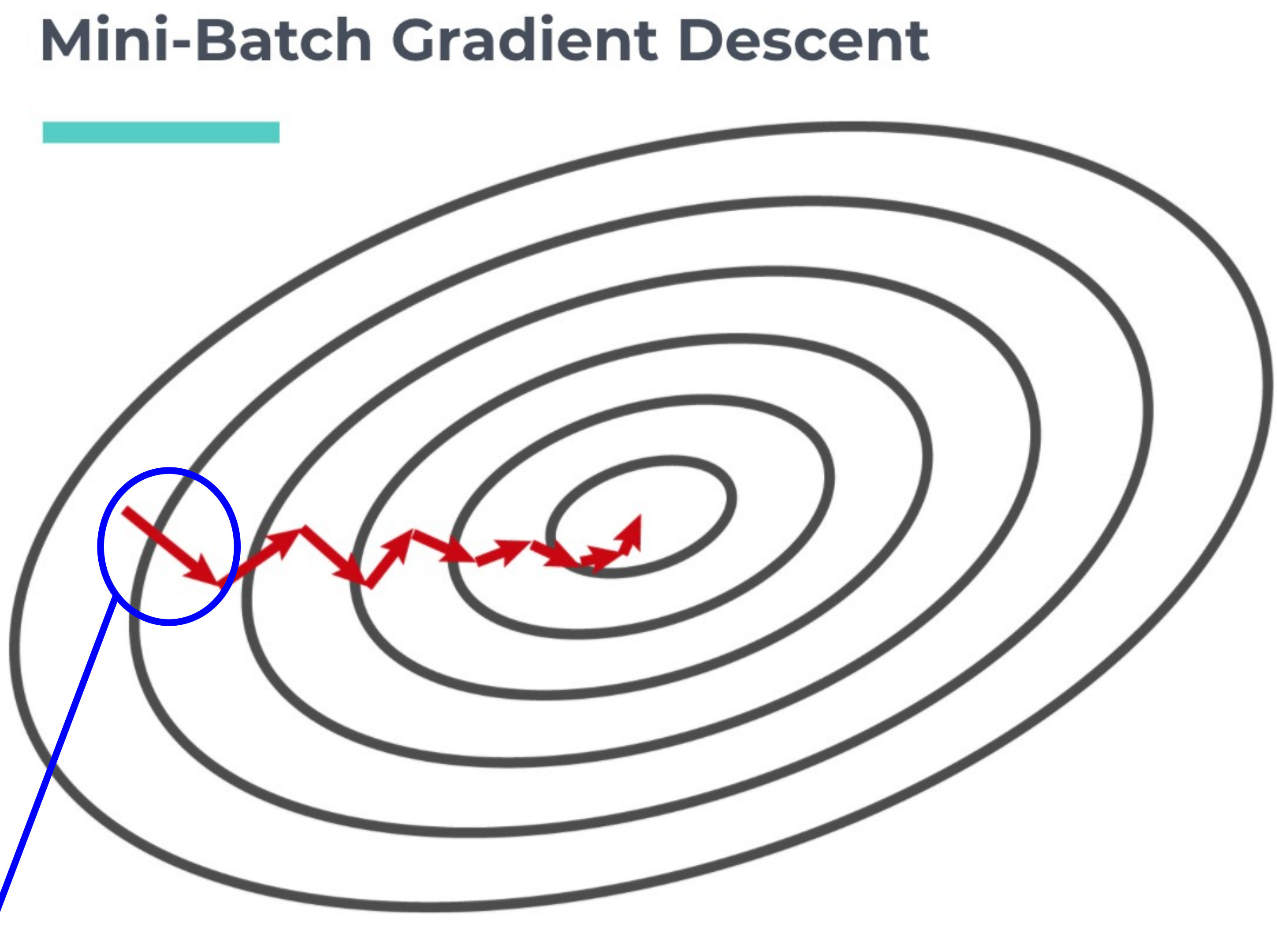
Weight Space Trajectory



Each update is based on **just 1 pattern** in the dataset

Source: Sam Abrahams (https://goo.gl/tKOvr7)

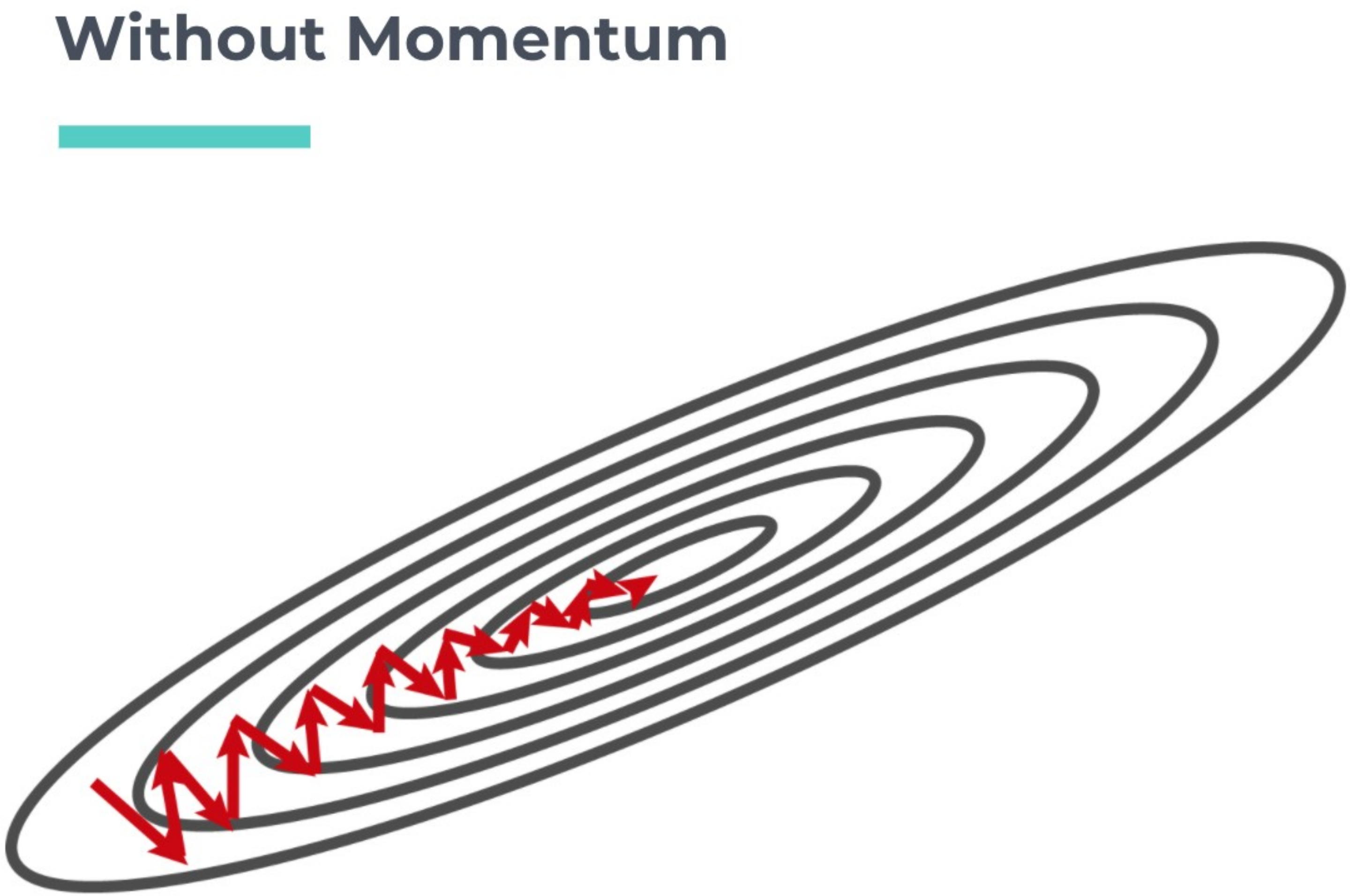
Weight Space Trajectory



Each update is based on a **mini-batch** of *m* patterns

Source: Sam Abrahams (https://goo.gl/tKOvr7)

Weight Space Trajectory



Source: Sam Abrahams (https://goo.gl/tKOvr7)

Weight Space Trajectory

Source: Sam Abrahams (https://goo.gl/tKOvr7)

Adding Momentum

Momentum parameter $0 \leq \alpha \leq 1$ controls how much the **previous weight change** at time $t - 1$ contributes to the **current** amount of weight change at time t

$$\underbrace{\Delta w_{ij}(t)}_{\text{weight change on the current time step}} = \underbrace{-\eta\, \delta_i\, a_j}_{\text{weight change calculated from the current cost gradient}} + \alpha \underbrace{\Delta w_{ij}(t-1)}_{\text{amount the weight was changed on the previous time step}}$$

hidden → output weights

Adding Momentum

Momentum parameter $0 \leq \alpha \leq 1$ controls how much the **previous weight change** at time $t - 1$ contributes to the **current** amount of weight change at time t

$$\Delta w_{ij}(t) \;=\; -\eta\, \delta_i\, a_j \;+\; \alpha\, \Delta w_{ij}(t-1)$$ hidden → output weights

$$\Delta w_{jk}(t) \;=\; -\eta\, \delta_j\, x_k \;+\; \alpha\, \Delta w_{jk}(t-1)$$ input → hidden weights

$$\Delta b_i(t) \;=\; -\eta\, \delta_i \;+\; \alpha\, \Delta b_i(t-1)$$ output unit biases

$$\Delta b_j(t) \;=\; -\eta\, \delta_j \;+\; \alpha\, \Delta b_j(t-1)$$ hidden unit biases

Many Variations on Gradient Descent

- **SGD** (Stochastic Gradient Descent) with Momentum
- **Adagrad** (Adaptive Gradient Descent)
- **RMSprop** (Root Mean Square Propagation)
- **Adam** (Adaptive Moment Estimation)
- **Nesterov** Accelerated Gradient
- **Nadam** (Nesterov-accelerated Adaptive Moment Estimation)

More info: https://ruder.io/optimizing-gradient-descent