function BACK-PROP-LEARNING(*examples*, *network*) **returns** a neural network
inputs: *examples*, a set of examples, each with input vector $\mathbf{x}$ and output vector $\mathbf{y}$
network, a multilayer network with L layers, weights $w_{i,j}$, activation function g
local variables: Δ, a vector of errors, indexed by network node

for each weight $w_{i,j}$ **in** *network* **do**
$w_{i,j} \leftarrow$ a small random number
repeat
for each example $(\mathbf{x}, \mathbf{y})$ **in** *examples* **do**
/ Propagate the inputs forward to compute the outputs */*
for each node i in the input layer **do**
$a_i \leftarrow x_i$
for $\ell = 2$ **to** L **do**
for each node j in layer ℓ **do**
$in_j \leftarrow \sum_i w_{i,j}\, a_i$
$a_j \leftarrow g(in_j)$
/ Propagate deltas backward from output layer to input layer */*
for each node j in the output layer **do**
$\Delta[j] \leftarrow y_j - a_j \quad (= -\partial Loss/\partial in_j)$
for $\ell = L - 1$ **to** 1 **do**
for each node i in layer ℓ **do**
$\Delta[i] \leftarrow g(in_i)(1 - g(in_i)) \sum_j w_{i,j} \Delta[j]$
/ Update every weight in network using deltas */*
for each weight $w_{i,j}$ **in** *network* **do**
$w_{i,j} \leftarrow w_{i,j} + \alpha \times a_i \times \Delta[j]$
until some stopping criterion is satisfied
return *network*

$Logistic(\cdot) \equiv g(\cdot)$
(pour simplifier notation)

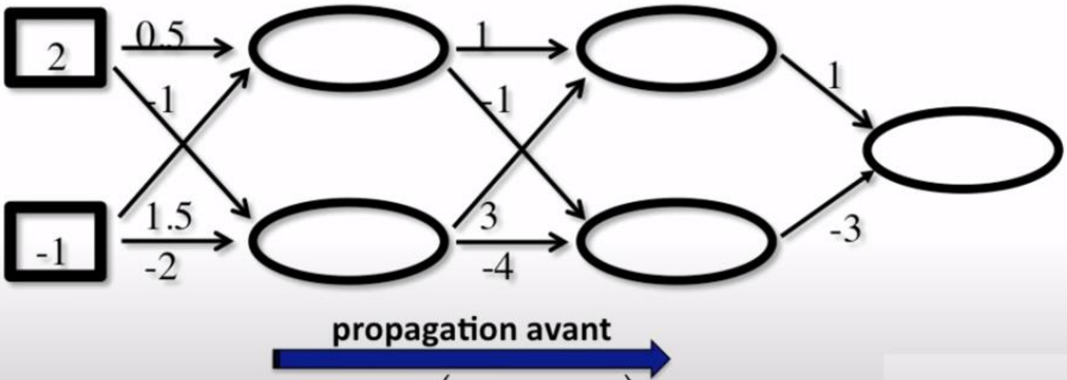
function BACK-PROP-LEARNING(*examples*, *network*) **returns** a neural network
inputs: *examples*, a set of examples, each with input vector $\mathbf{x}$ and output vector $\mathbf{y}$
network, a multilayer network with L layers, weights $w_{i,j}$, activation function g
local variables: Δ, a vector of errors, indexed by network node

for each weight $w_{i,j}$ **in** *network* **do**
$w_{i,j} \leftarrow$ a small random number
repeat
for each example $(\mathbf{x}, \mathbf{y})$ **in** *examples* **do**
/ Propagate the inputs forward to compute the outputs */*
for each node i in the input layer **do**
$a_i \leftarrow x_i$
for $\ell = 2$ **to** L **do**
for each node j in layer ℓ **do**
$in_j \leftarrow \sum_i w_{i,j}\, a_i$
$a_j \leftarrow g(in_j)$
/ Propagate deltas backward from output layer to input layer */*
for each node j in the output layer **do**
$\Delta[j] \leftarrow y_j - a_j \quad (= -\partial Loss/\partial in_j)$
for $\ell = L - 1$ **to** 1 **do**
for each node i in layer ℓ **do**
$\Delta[i] \leftarrow g(in_i)(1 - g(in_i)) \sum_j w_{i,j} \Delta[j]$
/ Update every weight in network using deltas */*
for each weight $w_{i,j}$ **in** *network* **do**
$w_{i,j} \leftarrow w_{i,j} + \alpha \times a_i \times \Delta[j]$
until some stopping criterion is satisfied
return *network*

$Logistic(\cdot) \equiv g(\cdot)$
(pour simplifier notation)

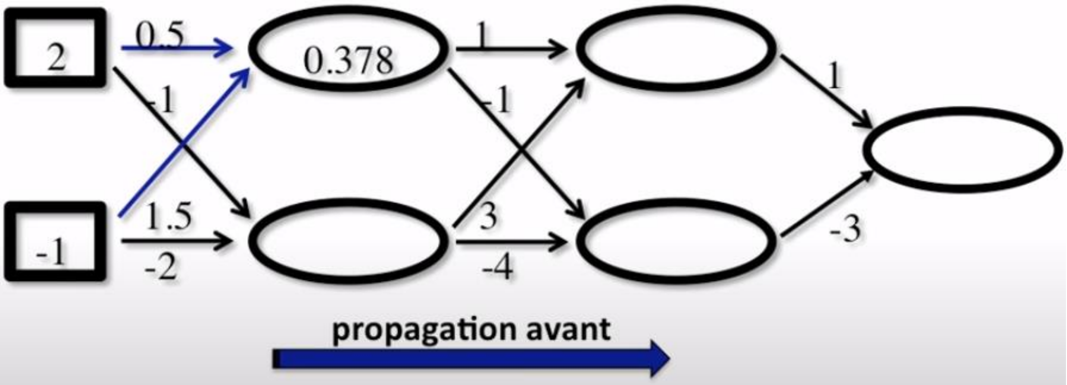
Exemple

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



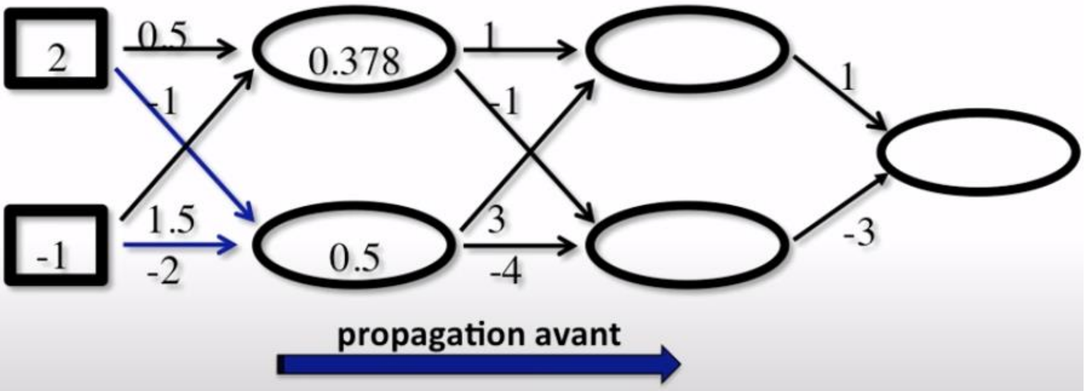
$$a_k = g\left(\sum_j w_{j,k} a_j\right)$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



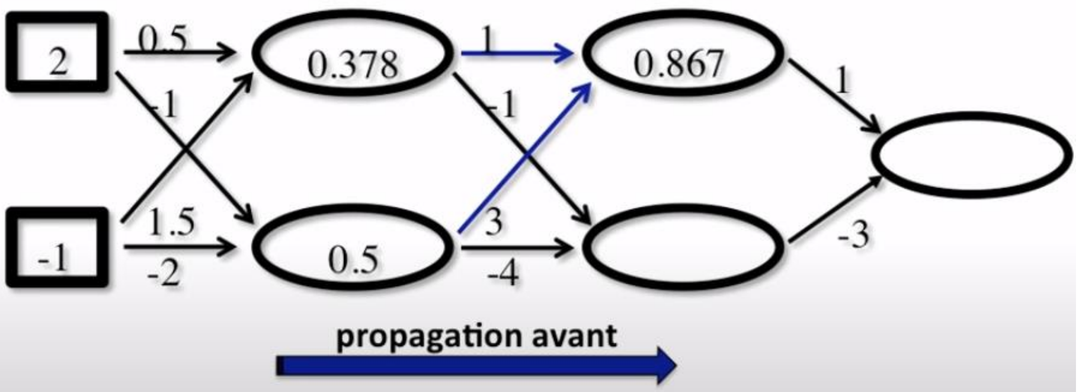
Logistic(0.5 * 2 + 1.5 * -1) = *Logistic*(-0.5) = 0.378

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



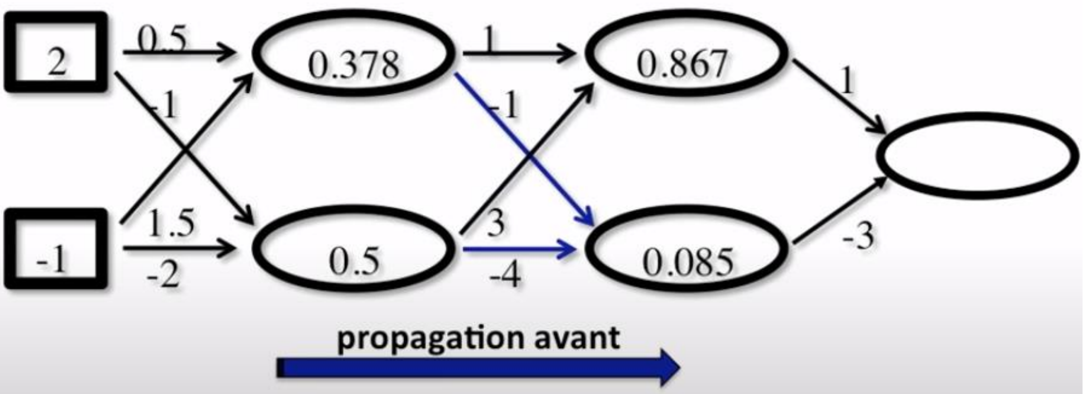
Logistic(-1 * 2 + -2 * -1) = *Logistic*(0) = 0.5

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



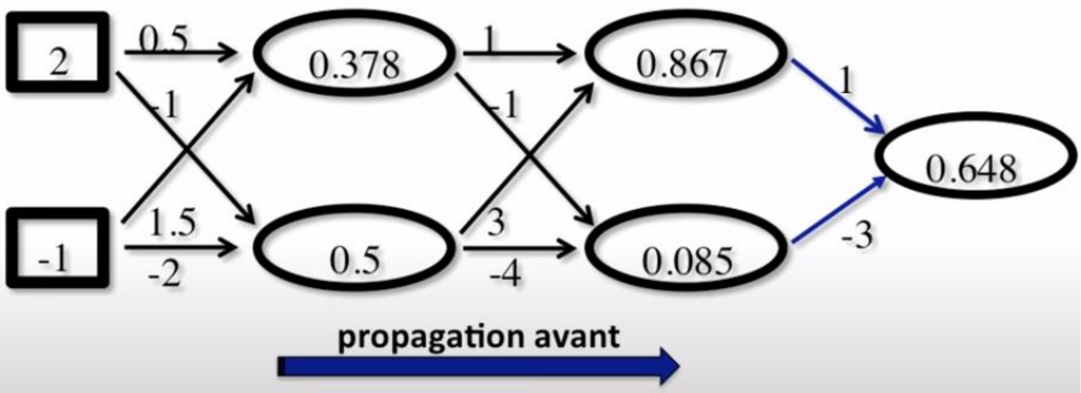
$$Logistic(1 * 0.378 + 3 * 0.5) = Logistic(1.878) = 0.867$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



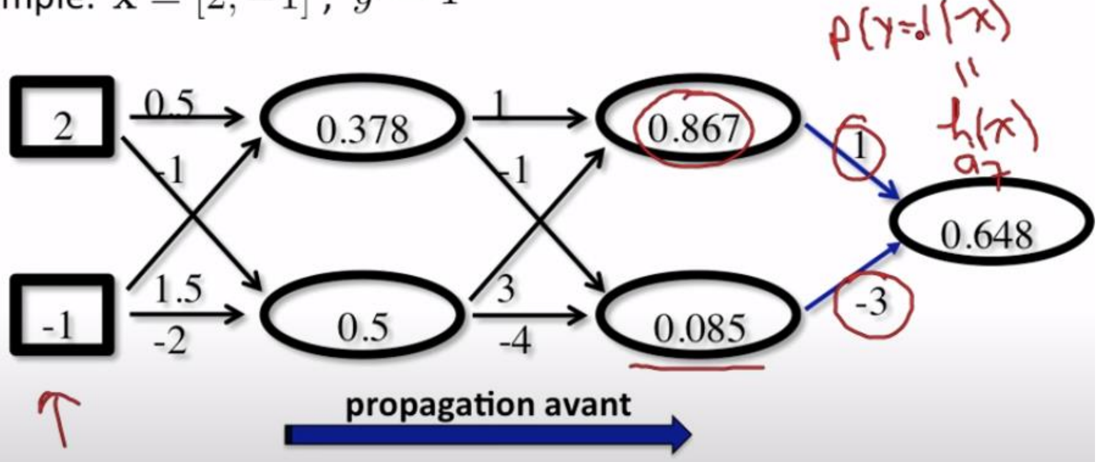
$$Logistic(-1 * 0.378 + -4 * 0.5) = Logistic(-2.378) = 0.085$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



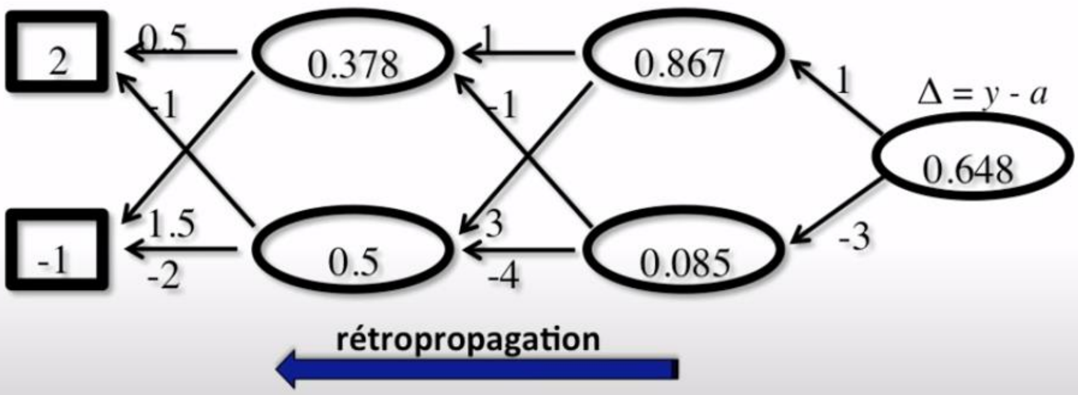
$$Logistic(1 * 0.867 + -3 * 0.085) = Logistic(0.612) = 0.648$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



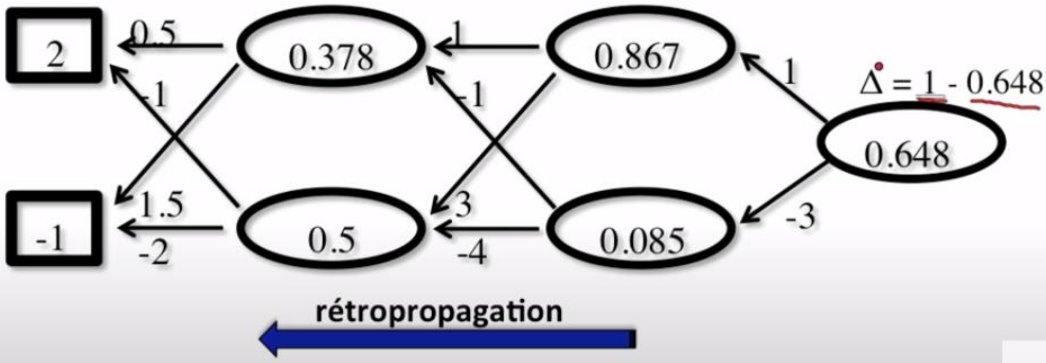
$$Logistic(1 * 0.867 + -3 * 0.085) = Logistic(0.612) = 0.648$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



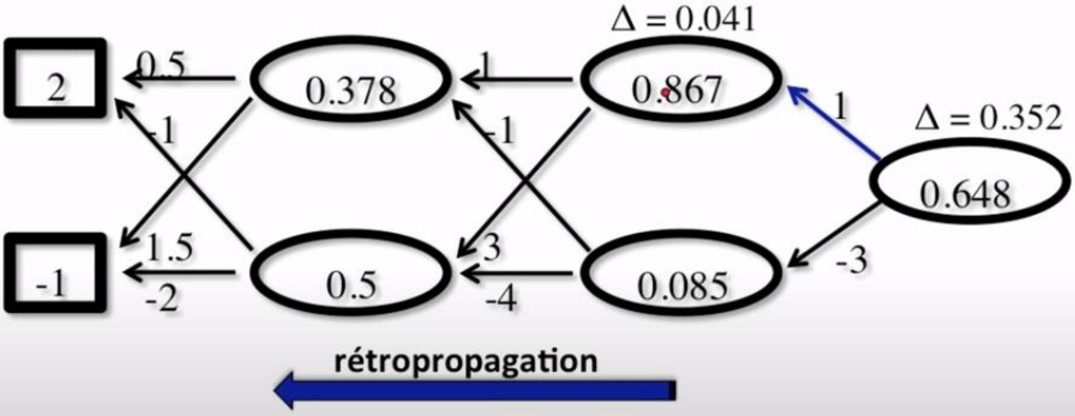
$$\Delta[j] = g(in_j)(1 - g(in_j)) \sum_k w_{j,k} \Delta[k]$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



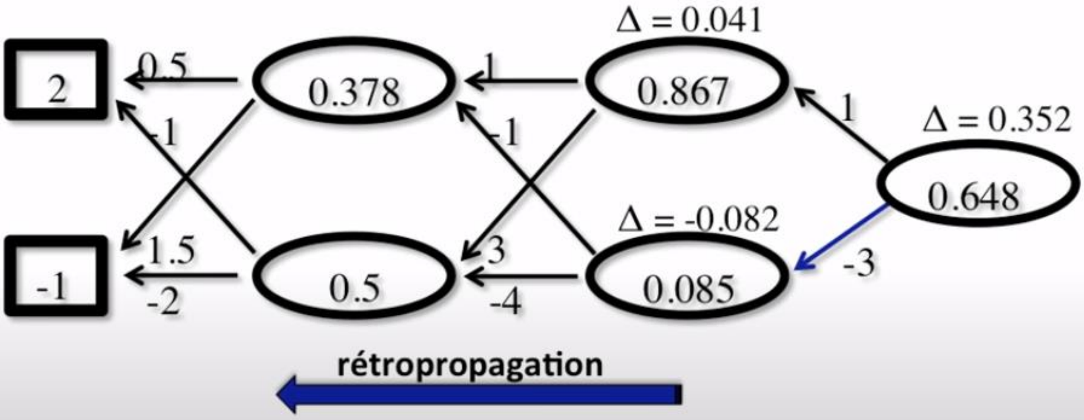
$$\Delta[j] = g(in_j)(1 - g(in_j)) \sum_k w_{j,k} \Delta[k]$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



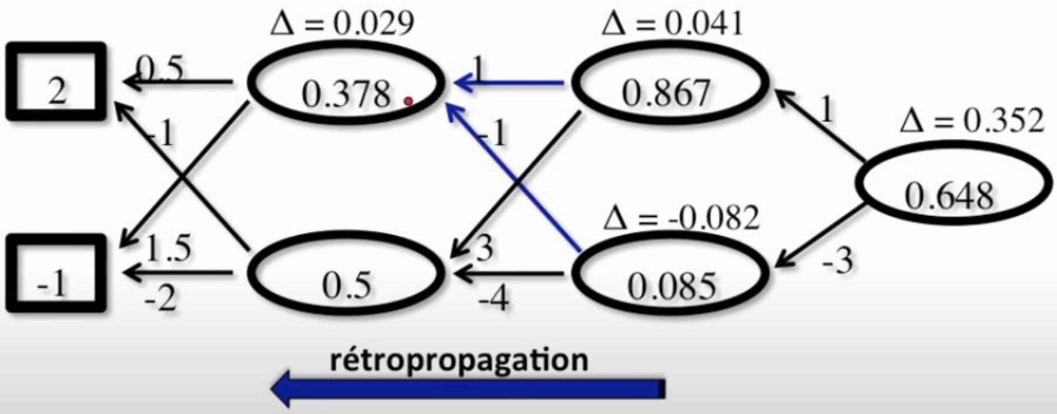
$$\Delta = 0.867 * (1-0.867) * 1 * 0.352 = 0.041$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



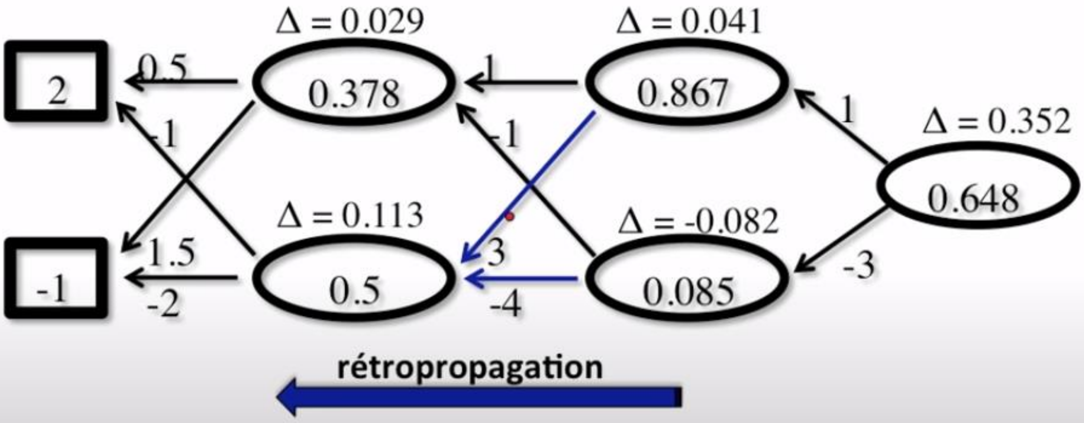
$$\Delta = 0.085 * (1-0.085) * -3 * 0.352 = -0.082$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



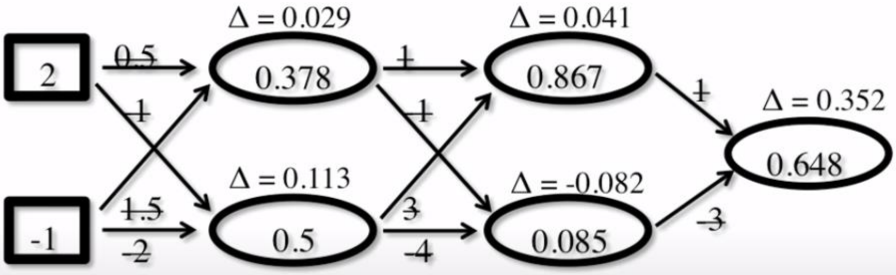
$$\Delta = 0.378 * (1-0.378) * (1 * 0.041 + -1 * -0.082) = 0.029$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



$$\Delta = 0.5 * (1\text{-}0.5) * (3 * 0.041 + \text{-}4 * \text{-}0.082) = 0.113$$

- Exemple: $\mathbf{x} = [2, -1]$, $y = 1$



mise à jour (α=0.1)

$w_{1,3} \leftarrow 0.5 + 0.1 * 2 * 0.029 = 0.506$

$w_{1,4} \leftarrow -1 + 0.1 * 2 * 0.113 = -0.977$

$w_{2,3} \leftarrow 1.5 + 0.1 * -1 * 0.029 = 1.497$

$w_{2,4} \leftarrow -2 + 0.1 * -1 * 0.113 = -2.011$

$w_{3,5} \leftarrow 1 + 0.1 * 0.378 * 0.041 = 1.002$

$w_{3,6} \leftarrow -1 + 0.1 * 0.378 * -0.082 = -1.003$

$w_{4,5} \leftarrow 3 + 0.1 * 0.5 * 0.041 = 3.002$

$w_{4,6} \leftarrow -4 + 0.1 * 0.5 * -0.082 = -4.004$

$w_{5,7} \leftarrow 1 + 0.1 * 0.867 * 0.352 = 1.031$

$w_{6,7} \leftarrow -3 + 0.1 * 0.085 * 0.352 = -2.997$