Neural Networks

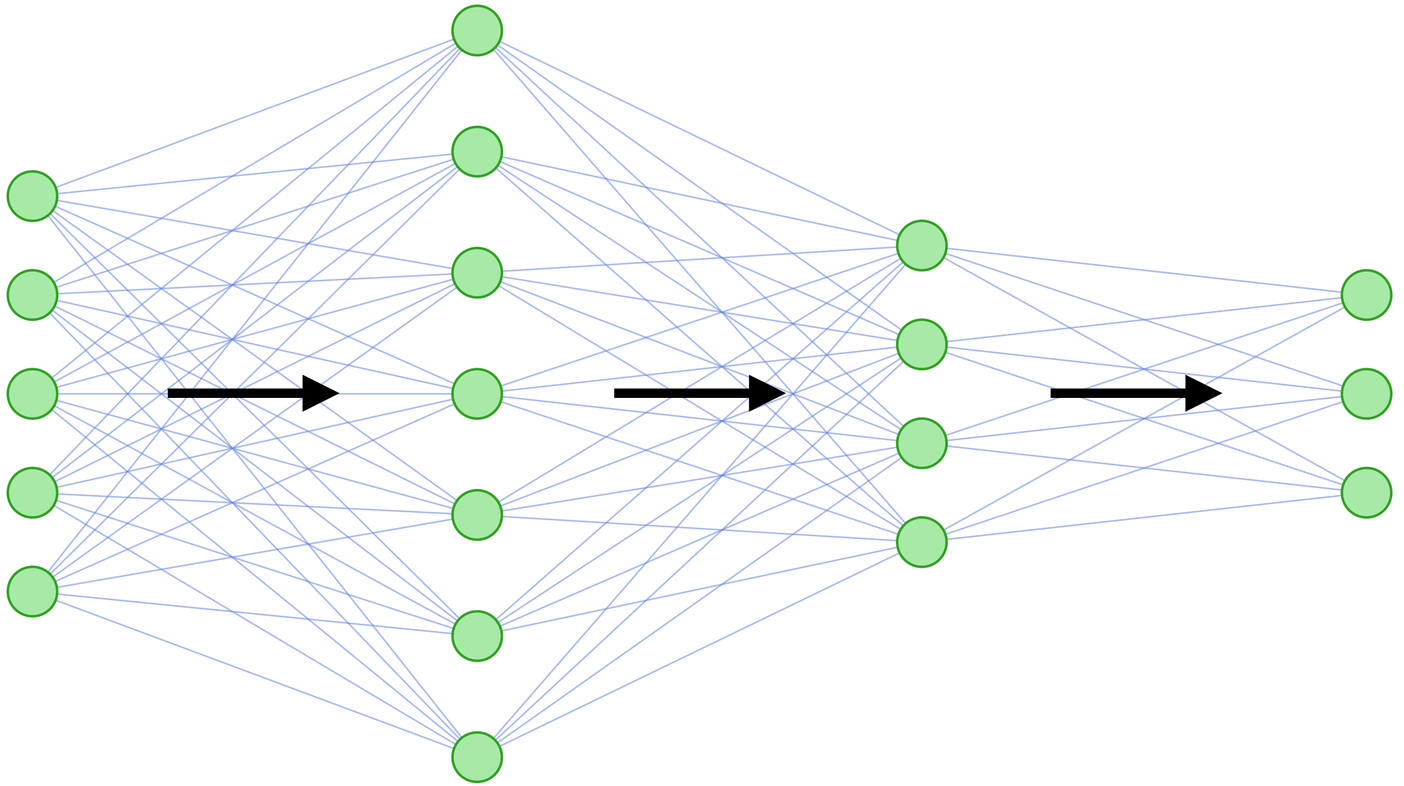
Forward/backward propagation + training

Rafał Nowak March 11, 2025

Forward propagation

$$f\colon \mathbb{R}^n \to \mathbb{R}^k$$

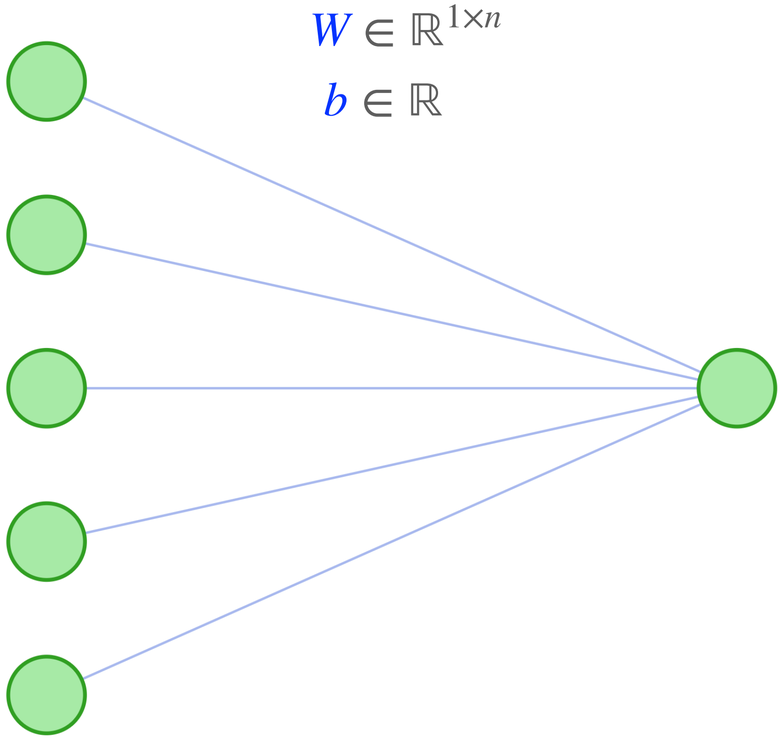
$\mathbb{R}^n \ni \boldsymbol{x}$



$\hat{\boldsymbol{y}} \in \mathbb{R}^k$

Logistic regression (recap)

$\mathbb{R}^n \ni \boldsymbol{x}$



$a \in (0,1)$

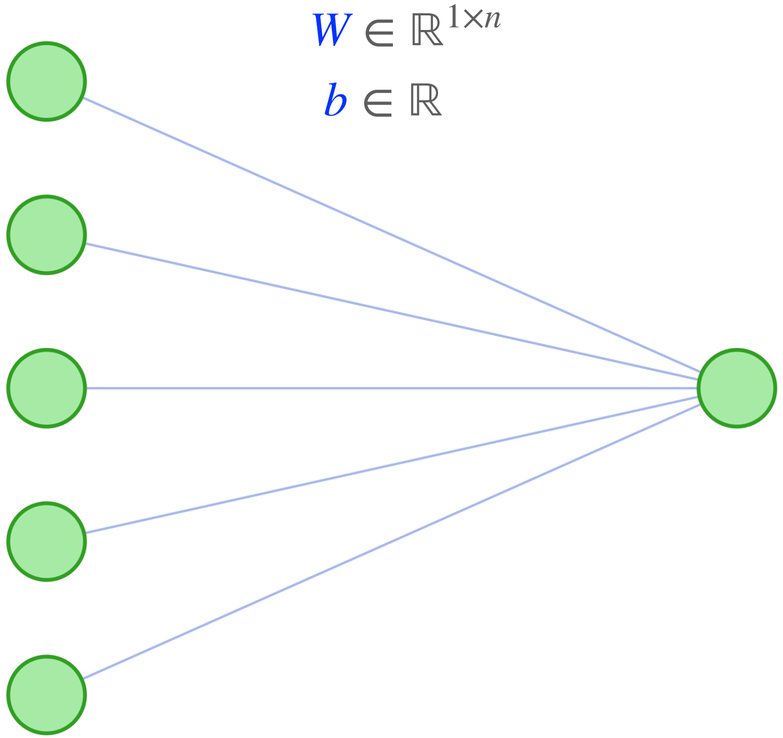
$\boldsymbol{x}^T \in \mathbb{R}^{1\times n}$

$z = \boldsymbol{x}^T W^T + b \in \mathbb{R}^{1\times 1}$

$a = \sigma(z) \in \mathbb{R}^{1\times 1}$

Logistic regression (recap)

$$\mathbb{R}^{N \times n} \ni \boldsymbol{X}$$



$$\boldsymbol{a} \in (0,1)^{N}$$

Efficient implementation

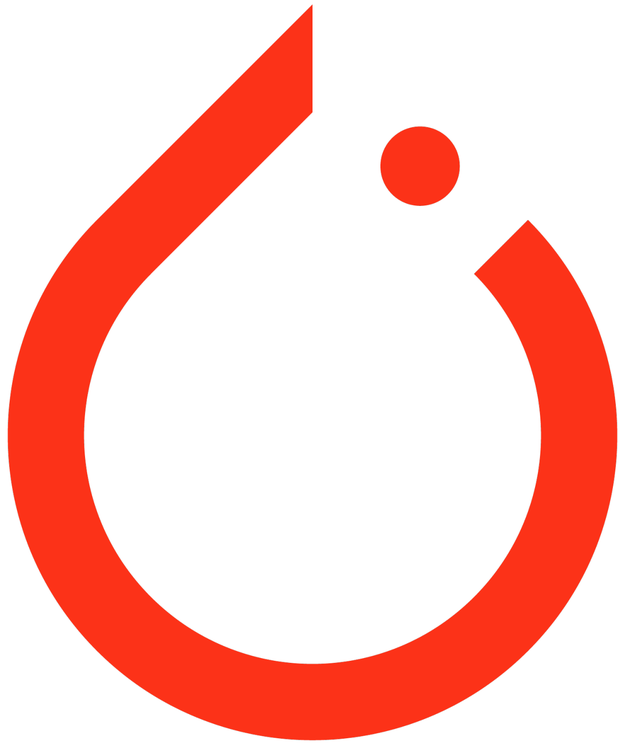
$$X \in \mathbb{R}^{N \times n}$$

$$z = X\,W^{T} \oplus b \in \mathbb{R}^{N \times 1}$$

$$\boldsymbol{a} = \sigma(z) \in \mathbb{R}^{N \times 1}$$

Forward propagation

Code demo in PyTorch



Forward propagation

In [1]:
```
import torch

torch.manual_seed(1); # for the reproducibility
```
executed in 299ms, finished 17:46:14 2023-03-11

Linear model

In [2]:
```
xT = torch.tensor([-8.0, 1.0, -3.0, 5.0, 2.0]).view(1, 5) # horizontal vector
xT
```
executed in 22ms, finished 17:46:14 2023-03-11

Out[2]:
```
tensor([[-8.,  1., -3.,  5.,  2.]])
```

In [3]:
```
xT.shape
```
executed in 19ms, finished 17:46:14 2023-03-11

Out[3]:
```
torch.Size([1, 5])
```

In [4]:
```
W = torch.rand(5).view(1, 5)  # random numbers from (0, 1)
b = torch.rand(1)
```
executed in 2ms, finished 17:46:14 2023-03-11

In [5]:
```
W
```
executed in 2ms, finished 17:46:14 2023-03-11

Out[5]:
```
tensor([[0.7576, 0.2793, 0.4031, 0.7347, 0.0293]])
```

In [6]:
```
b
```
executed in 3ms, finished 17:46:14 2023-03-11

Out[6]:
```
tensor([0.7999])
```

In [7]:
```
W.shape, b.shape
```
executed in 2ms, finished 17:46:14 2023-03-11

Out[7]:
```
(torch.Size([1, 5]), torch.Size([1]))
```

$$z = \boldsymbol{x}^T W^T + b \in \mathbb{R}^{1\times 1}$$

In [8]:
```
z = torch.mm( xT, W.T ) + b
z.shape
```
executed in 2ms, finished 17:46:14 2023-03-11

Out[8]:
```
torch.Size([1, 1])
```

In [9]:
```
z
```
executed in 3ms, finished 17:46:14 2023-03-11

Out[9]:
```
tensor([[-2.4591]])
```

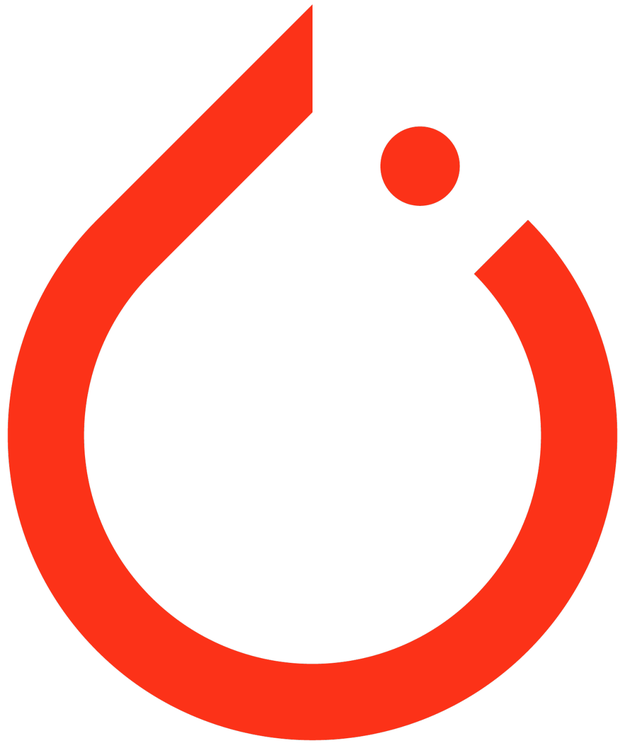
$$a = \sigma(z) \in \mathbb{R}^{1\times 1}$$

In [10]:
```
a = torch.sigmoid(z)
a
```
executed in 3ms, finished 17:46:14 2023-03-11

Out[10]:
```
tensor([[0.0788]])
```

Forward propagation

Code demo in PyTorch



Forward propagation

In [1]:
```
import torch

torch.manual_seed(1); # for the reproducibility
```
executed in 284ms, finished 17:43:29 2023-03-11

Linear model

In [2]:
```
X = torch.tensor([[-8.0, 1.0, -3.0, 5.0, 2.0], [-2.0, 2.0, -1.0, 5.0, 2.0], [7.0, 1.0, 9.0, -5.0, -2.0]])
X
```
executed in 17ms, finished 17:43:29 2023-03-11

Out[2]:
```
tensor([[-8.,  1., -3.,  5.,  2.],
        [-2.,  2., -1.,  5.,  2.],
        [ 7.,  1.,  9., -5., -2.]])
```

In [3]:
```
X.shape
```
executed in 25ms, finished 17:43:29 2023-03-11

Out[3]:
```
torch.Size([3, 5])
```

In [4]:
```
W = torch.rand(5).view(1, 5)  # random numbers from (0, 1)
b = torch.rand(1)
```
executed in 8ms, finished 17:43:29 2023-03-11

In [5]:
```
W
```
executed in 2ms, finished 17:43:29 2023-03-11

Out[5]:
```
tensor([[0.7576, 0.2793, 0.4031, 0.7347, 0.0293]])
```

In [6]:
```
b
```
executed in 3ms, finished 17:43:29 2023-03-11

Out[6]:
```
tensor([0.7999])
```

In [7]:
```
W.shape, b.shape
```
executed in 2ms, finished 17:43:29 2023-03-11

Out[7]:
```
(torch.Size([1, 5]), torch.Size([1]))
```

$$z = X\,W^T \oplus b \in \mathbb{R}^{N\times 1}$$

In [8]:
```
z = torch.mm( X, W.T ) + b
z.shape
```
executed in 2ms, finished 17:43:29 2023-03-11

Out[8]:
```
torch.Size([3, 1])
```

In [9]:
```
z
```
executed in 2ms, finished 17:43:29 2023-03-11

Out[9]:
```
tensor([[-2.4591],
        [ 3.1721],
        [ 6.2782]])
```

$$a = \sigma(z) \in \mathbb{R}^{N\times 1}$$

In [10]:
```
a = torch.sigmoid(z)
a
```
executed in 2ms, finished 17:43:29 2023-03-11

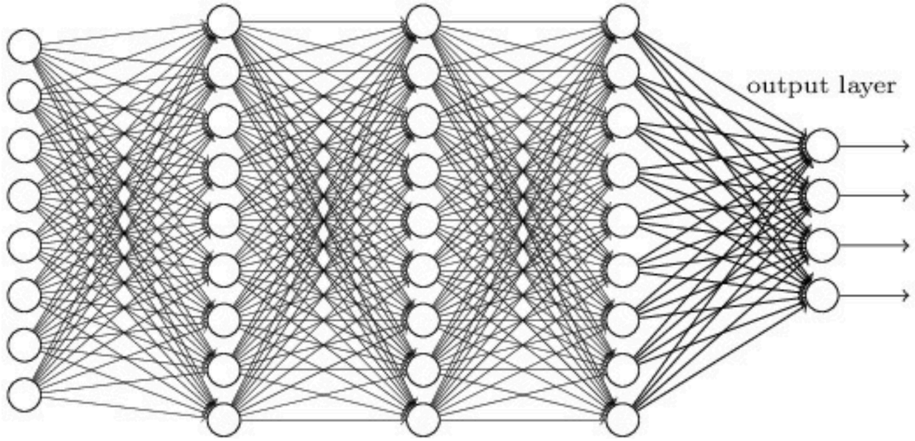
Out[10]:
```
tensor([[0.0788],
        [0.9598],
        [0.9981]])
```

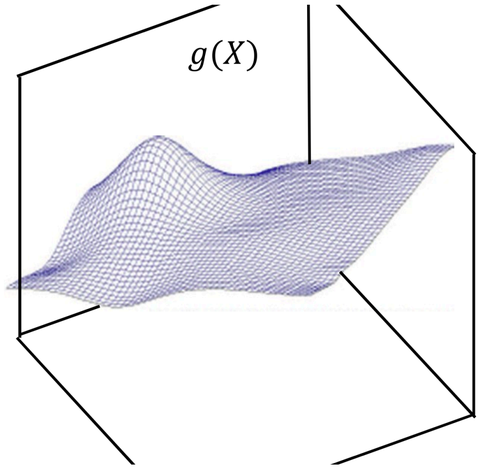
Supervised **learning**

- regression
- classification
- …

What we learn?

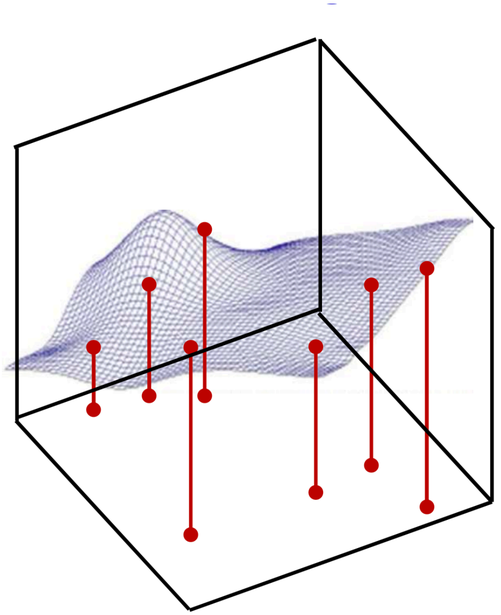
- network parameters W
- $f(X, W) \approx g(X)$

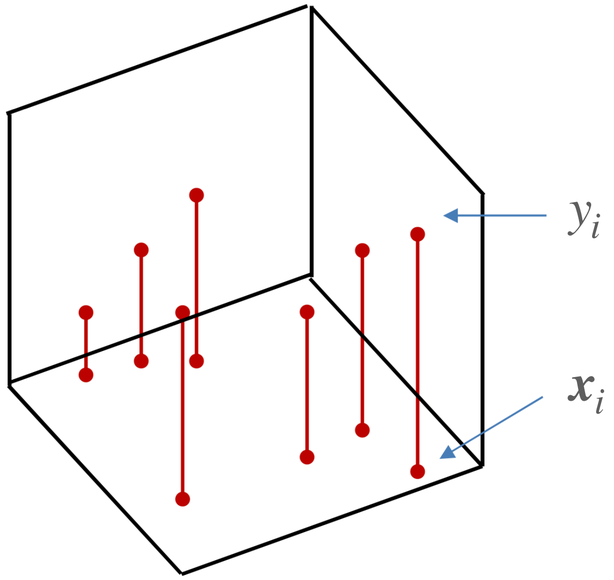




Learning by sampling

- training set $X_{\text{train}} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times n}$, $y_{\text{train}} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N]^T \in \mathbb{R}^{N \times k}$
- $f(X, W) \approx g(X)$
- Questions:
  - good sampling?
- Examples:
  - images + labels
  - speech recordings and their transcription





Training the neural network

- Find the parameters W that minimize the total average **loss**
- Total average loss:

$$\text{LOSS} = \frac{1}{N}\sum_{p=1}^{N} div(\hat{\boldsymbol{y}}_p, \boldsymbol{y}_p) \qquad \hat{\boldsymbol{y}}_p := f(\boldsymbol{x}_p; W)$$

Output Activation + Loss Function

- Output activation:
  - regression -> identity (id)
  - classification
    - binary -> sigmoid
    - multiclass -> softmax
- Loss function
  - regression —> mean squared error (MSE)
  - classification —> cross entropy (CE)

Linear regression

MSE Loss function

- (scaled) L_2 error (squared euclidean distance)

$$f(\boldsymbol{x};\, \boldsymbol{w}, b) = \hat{y}$$

$$\hat{y} = \boldsymbol{x}\boldsymbol{w}^T + b$$

$$\text{div}(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$$

Logistic regression

Cross entropy

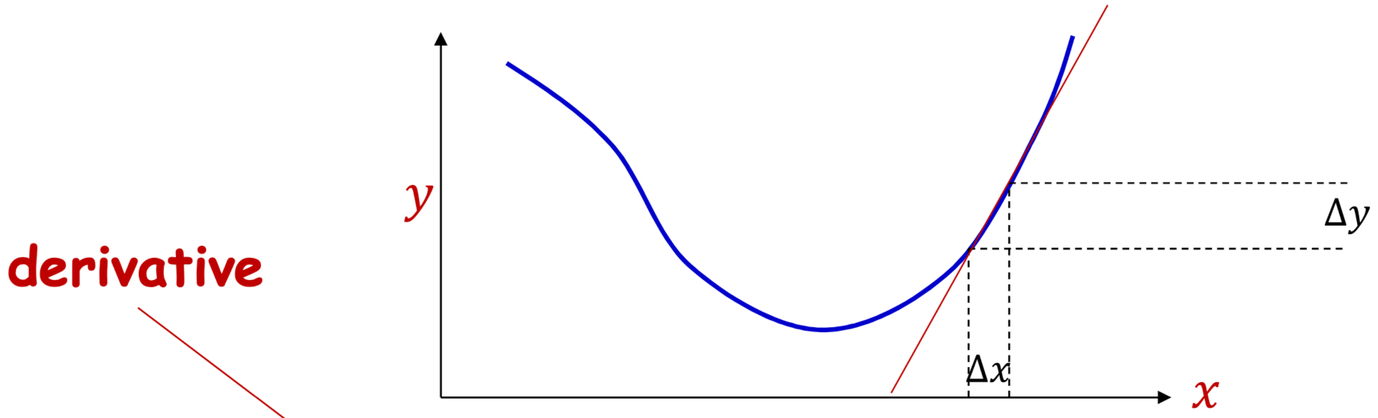
- binary cross entropy loss

$$f(\boldsymbol{x};\, \boldsymbol{w}, b) = \hat{y}$$

$$\hat{y} = \sigma(\boldsymbol{x}\boldsymbol{w}^T + b)$$

$$\text{div}(\hat{y}, y) = -\, y \log \hat{y} - (1 - y)\log(1 - \hat{y})$$

A brief note on derivatives..

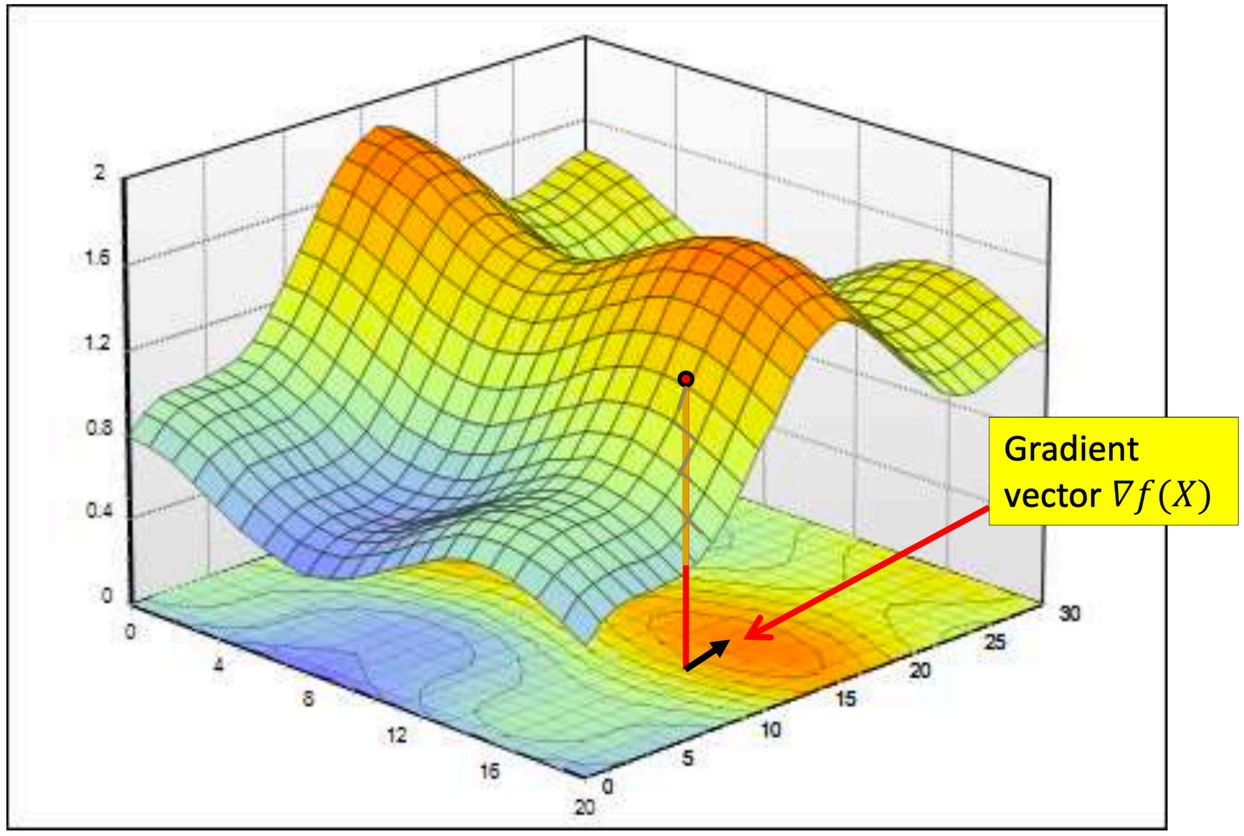


- A derivative of a function at any point tells us how much a minute increment to the *argument* of the function will increment the *value* of the function
  - For any $y = f(x)$, expressed as a multiplier α to a tiny increment Δx to obtain the increments Δy to the output

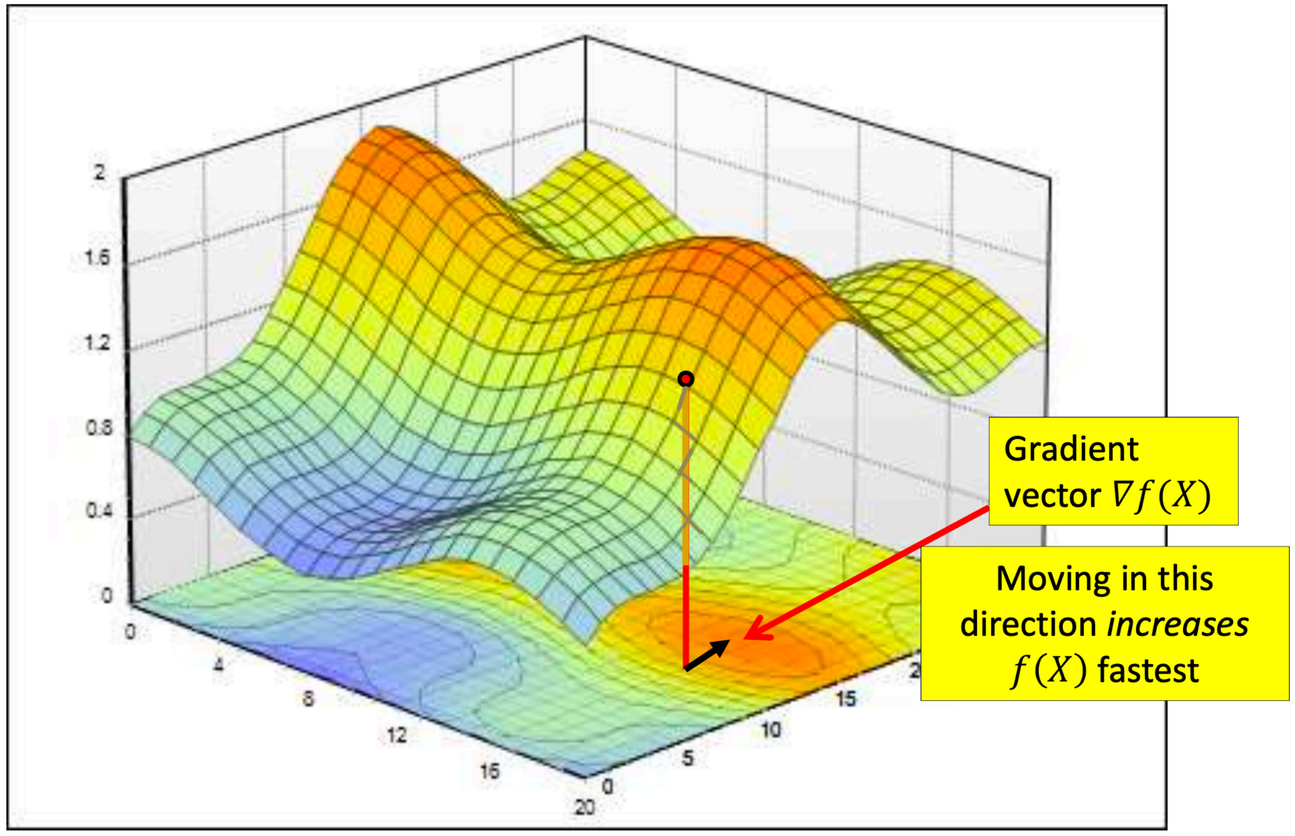
$$\Delta y = \alpha \Delta x$$

  - Based on the fact that at a fine enough resolution, any smooth, continuous function is locally linear at any point

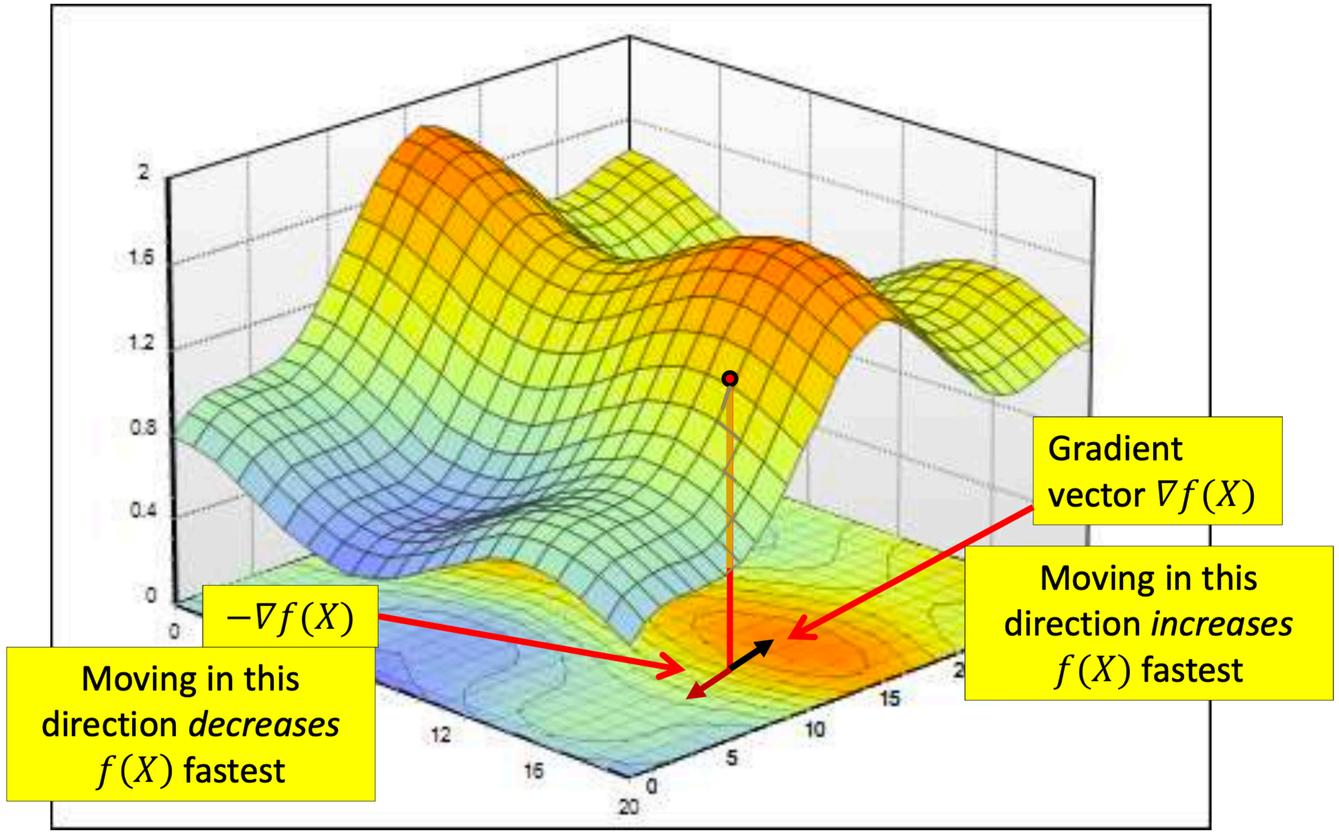
Gradient


2
1.6
1.2
0.8
0.4
0
0
4
8
12
16
20
0
5
10
15
20
25
30
Gradient vector $\nabla f(X)$

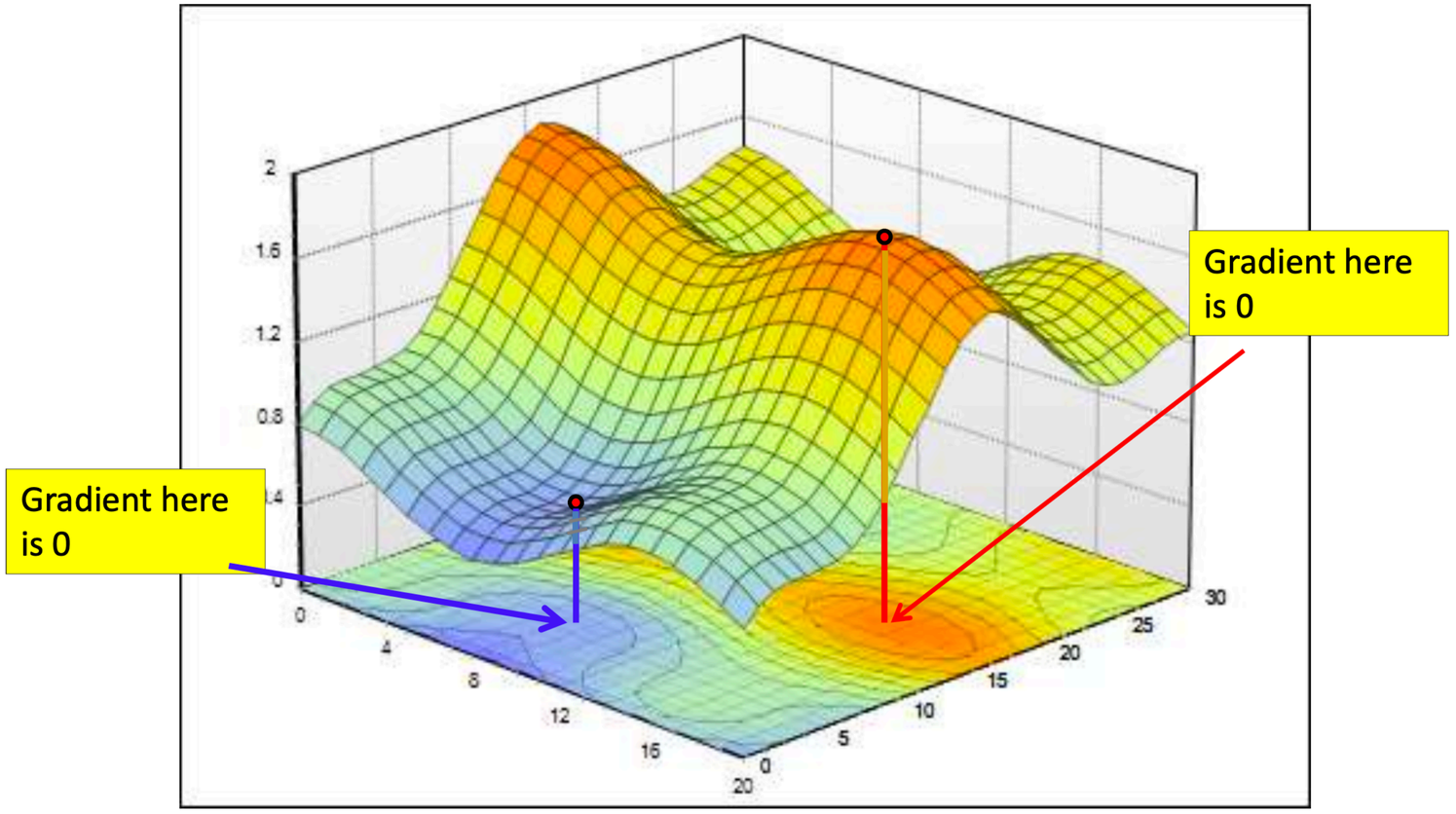
Gradient

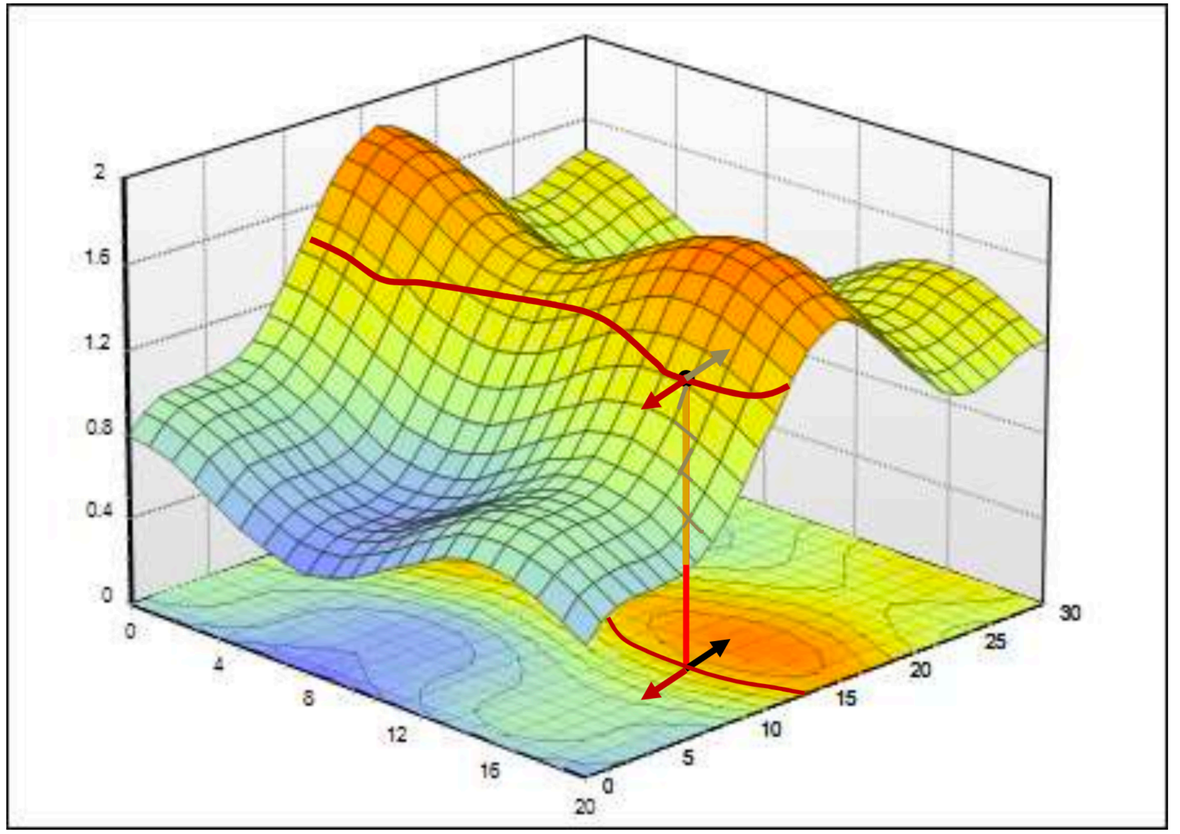

2
1.6
1.2
0.8
0.4
0
0
4
8
12
16
20
0
5
10
15
Gradient vector $\nabla f(X)$
Moving in this direction *increases* $f(X)$ fastest

Gradient


2
1.6
1.2
0.8
0.4
0
0
12
16
20
0
5
10
15
Gradient vector $\nabla f(X)$
Moving in this direction *increases* $f(X)$ fastest
$-\nabla f(X)$
Moving in this direction *decreases* $f(X)$ fastest

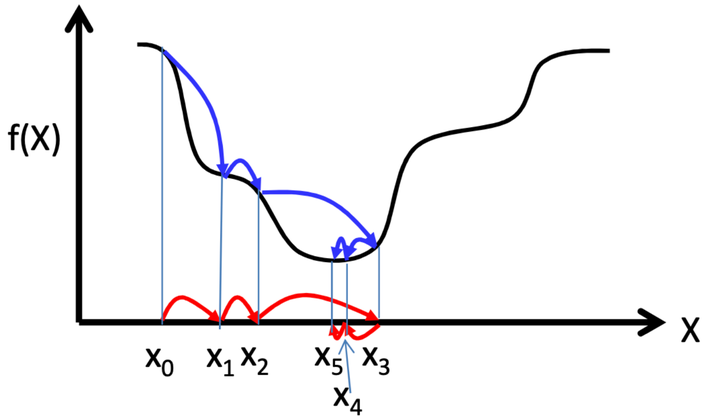
Gradient

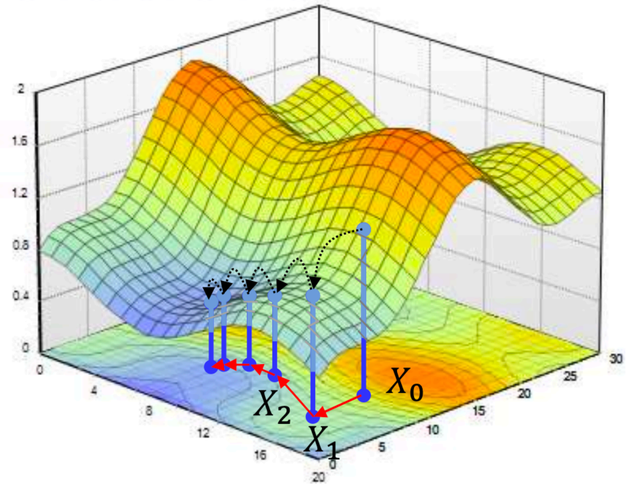

Gradient here is 0
Gradient here is 0
2
1.6
1.2
0.8
0
4
8
12
16
20
0
5
10
15
20
25
30



- The gradient vector $\nabla f(X)$ is perpendicular to the level curve

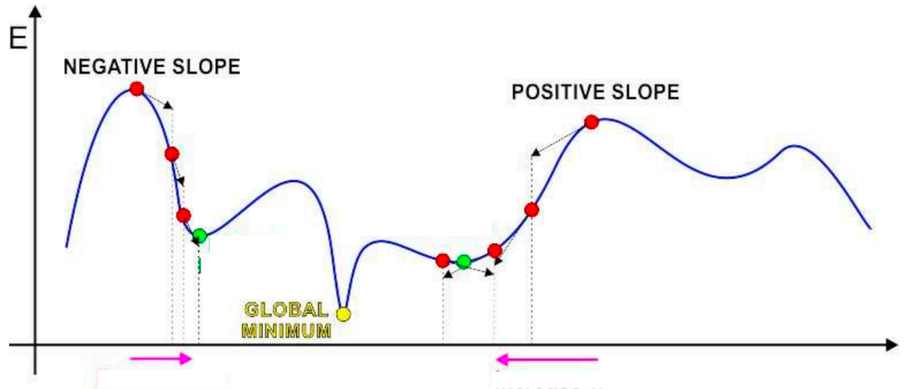
Iterative solutions





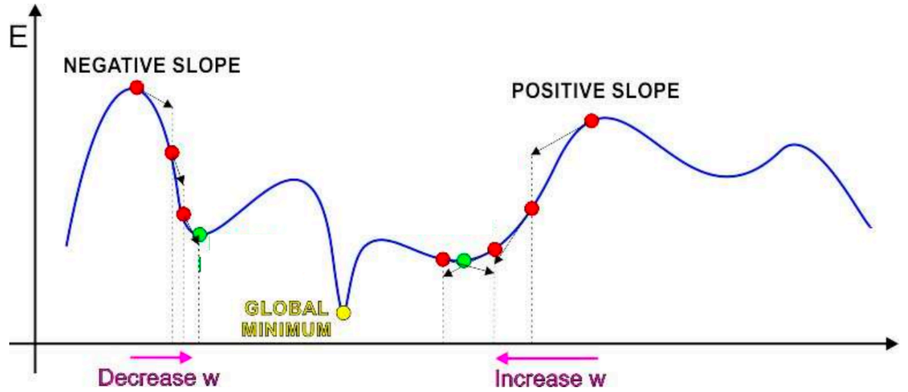
- Iterative solutions
  - Start from an initial guess X_0 for the optimal X
  - Update the guess towards a (hopefully) "better" value of $\mathrm{f}(X)$
  - Stop when $\mathrm{f}(X)$ no longer decreases
- Problems:
  - Which direction to step in
  - How big must the steps be

The Approach of Gradient Descent



- Iterative solution:
  - Start at some point
  - Find direction in which to shift this point to decrease error
    - This can be found from the derivative of the function
      - A positive derivative → moving left decreases error
      - A negative derivative → moving right decreases error
  - Shift point in this direction

The Approach of Gradient Descent



- Iterative solution: Trivial algorithm
  - Initialize x^0
  - While $f'(x^k) \neq 0$
    - If $sign\left(f'(x^k)\right)$ is positive:

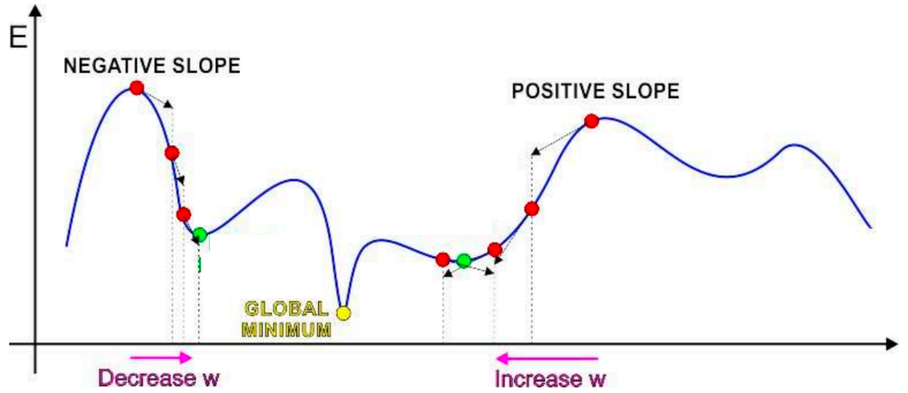
$$x^{k+1} = x^k - step$$

    - Else

$$x^{k+1} = x^k + step$$

  - What must step be to ensure we actually get to the optimum?

The Approach of Gradient Descent

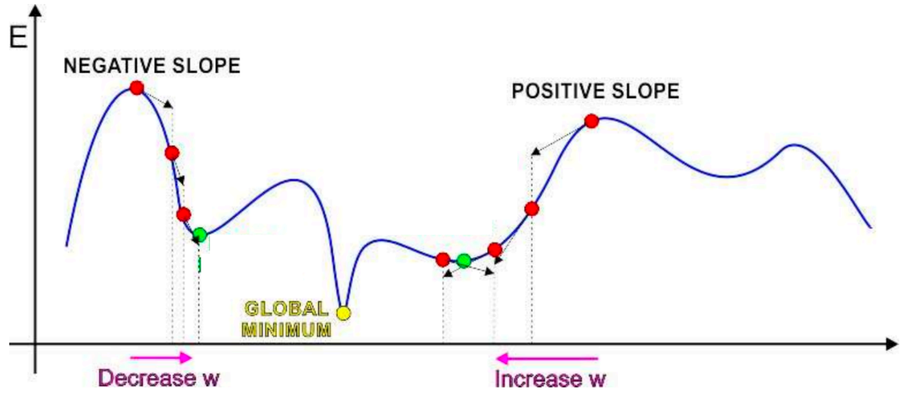


- Iterative solution: Trivial algorithm
  - Initialize x^0
  - While $f'(x^k) \neq 0$

$$x^{k+1} = x^k - sign\left(f'(x^k)\right).step$$

- Identical to previous algorithm

The Approach of Gradient Descent



- Iterative solution: Trivial algorithm
  - Initialize x^0
  - While $f'(x^k) \neq 0$

$$x^{k+1} = x^k - \eta^k f'(x^k)$$

- η^k is the "step size"

Gradient descent/ascent (multivariate)

- The gradient descent/ascent method to find the minimum or maximum of a function f iteratively
  - To find a *maximum* move *in the direction of the gradient*

$$x^{k+1} = x^k + \eta^k \nabla f\left(x^k\right)^T$$

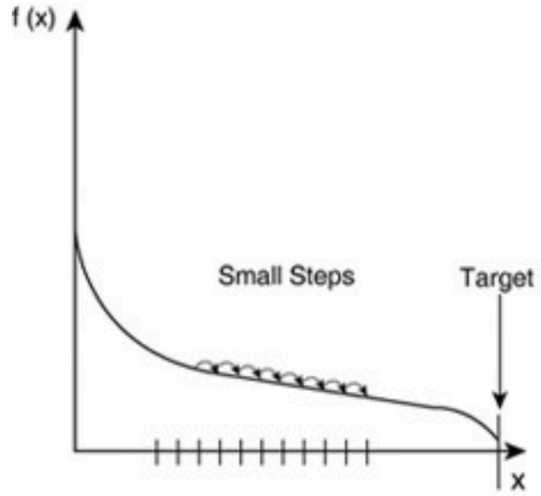
  - To find a *minimum* move *exactly opposite the direction of the gradient*

$$x^{k+1} = x^k - \eta^k \nabla f\left(x^k\right)^T$$

- Many solutions to choosing step size η^k

1. Fixed step size

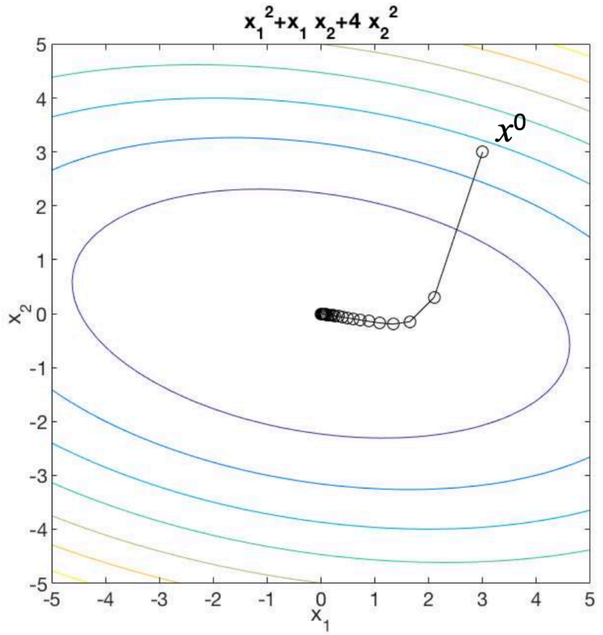
- Fixed step size
  - Use fixed value for η^k



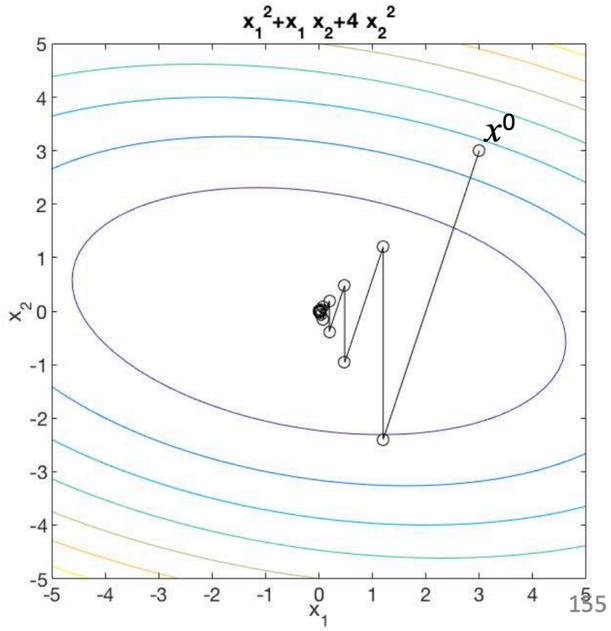
Influence of step size example (constant step size)

$$f(x_1, x_2) = (x_1)^2 + x_1 x_2 + 4(x_2)^2 \qquad x^{initial} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$\eta = 0.1$



$\eta = 0.2$

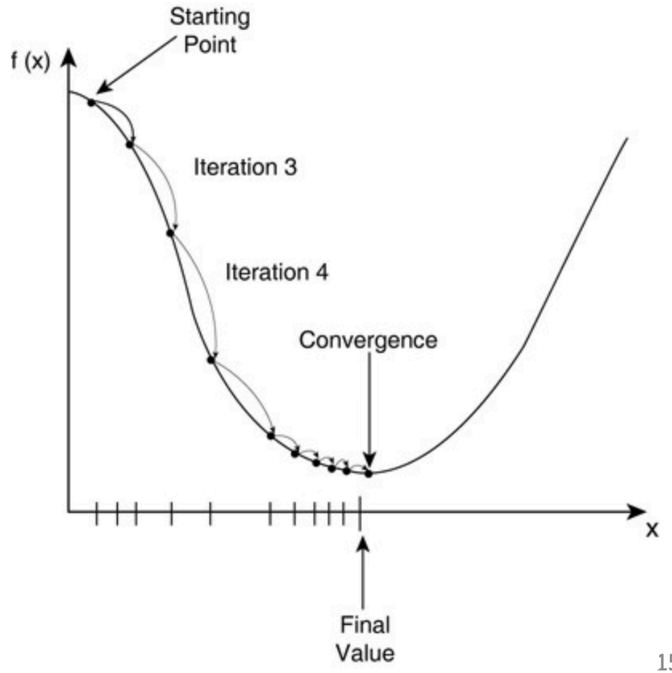


Gradient descent convergence criteria

- The gradient descent algorithm converges when one of the following criteria is satisfied

$$\left| f(x^{k+1}) - f(x^k) \right| < \varepsilon_1$$

- Or $\left\| \nabla f(x^k) \right\| < \varepsilon_2$



Recap: Gradient Descent Algorithm

- In order to minimize any function $f(x)$ w.r.t. x
- Initialize:
  - x^0
  - $k = 0$
- While $\left|f\left(x^{k+1}\right) - f\left(x^{k}\right)\right| > \varepsilon$
  - For every component i
    - $x_i^{k+1} = x_i^k - \eta^k \frac{df}{dx_i}$

    Explicitly stating it by component
  - $k = k + 1$

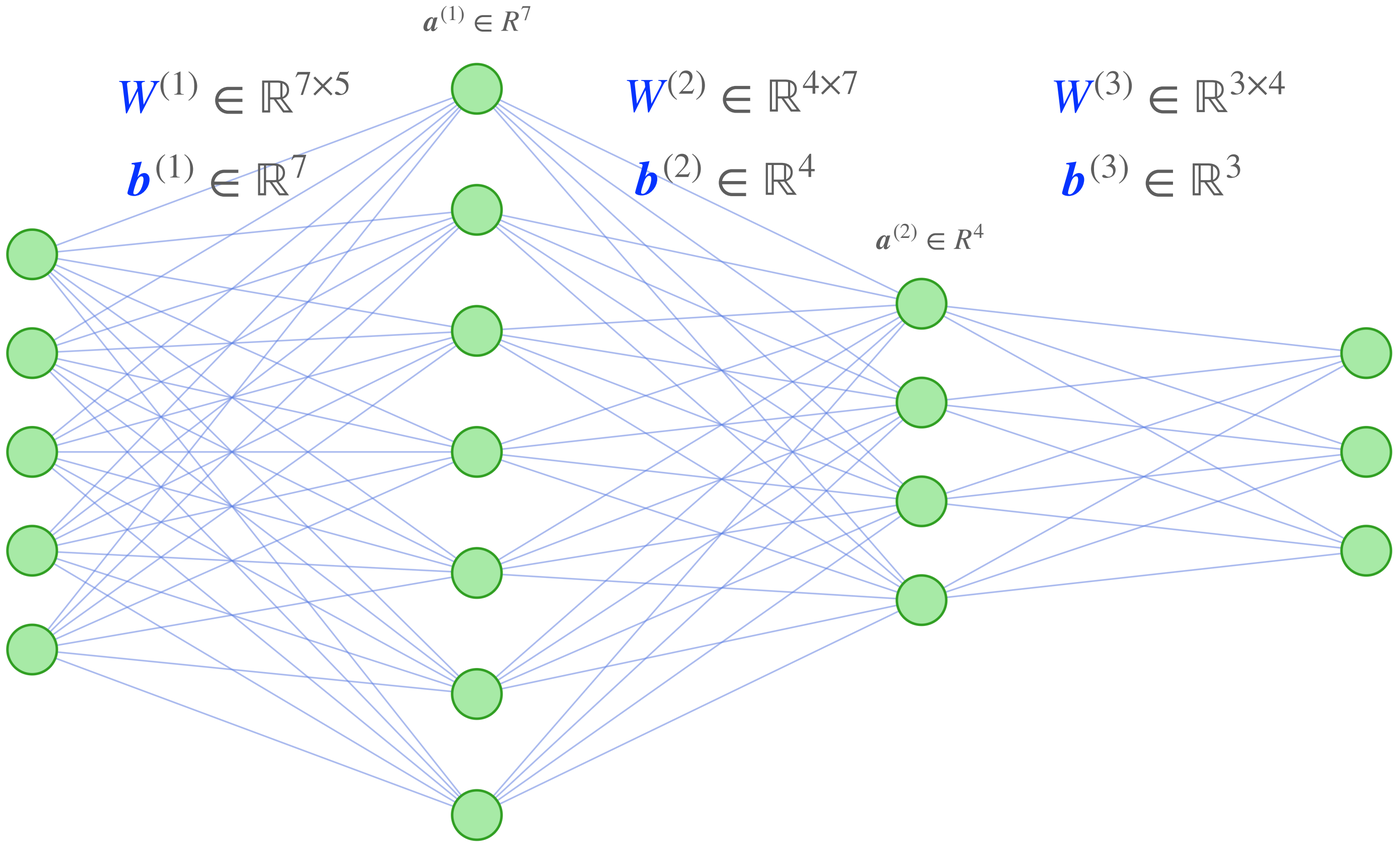
How to calculate gradients?

- Solution = **forward** + **backward** propagation
- Forward propagation -> calculate neuron values (from left to right)
- Backward propagation -> calculate partial derivates of **loss function** w.r.t to neuron values -> calculate partial derivatives w.r.t to **model parameters**.

Forward propagation

MLP

$\mathbb{R}^5 \ni \boldsymbol{x}$



$\boldsymbol{a}^{(3)} \in \mathbb{R}^3$

Efficient implementation

$\boldsymbol{x}^T \in \mathbb{R}^{1\times5}$

$z^{(1)} = \boldsymbol{x}^T W^{(1)^T} + \boldsymbol{b}^{(1)} \in \mathbb{R}^{1\times7}$

$\boldsymbol{a}^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times7}$

$z^{(2)} = \boldsymbol{a}^{(1)} W^{(2)^T} + \boldsymbol{b}^{(2)} \in \mathbb{R}^{1\times4}$

$\boldsymbol{a}^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times4}$

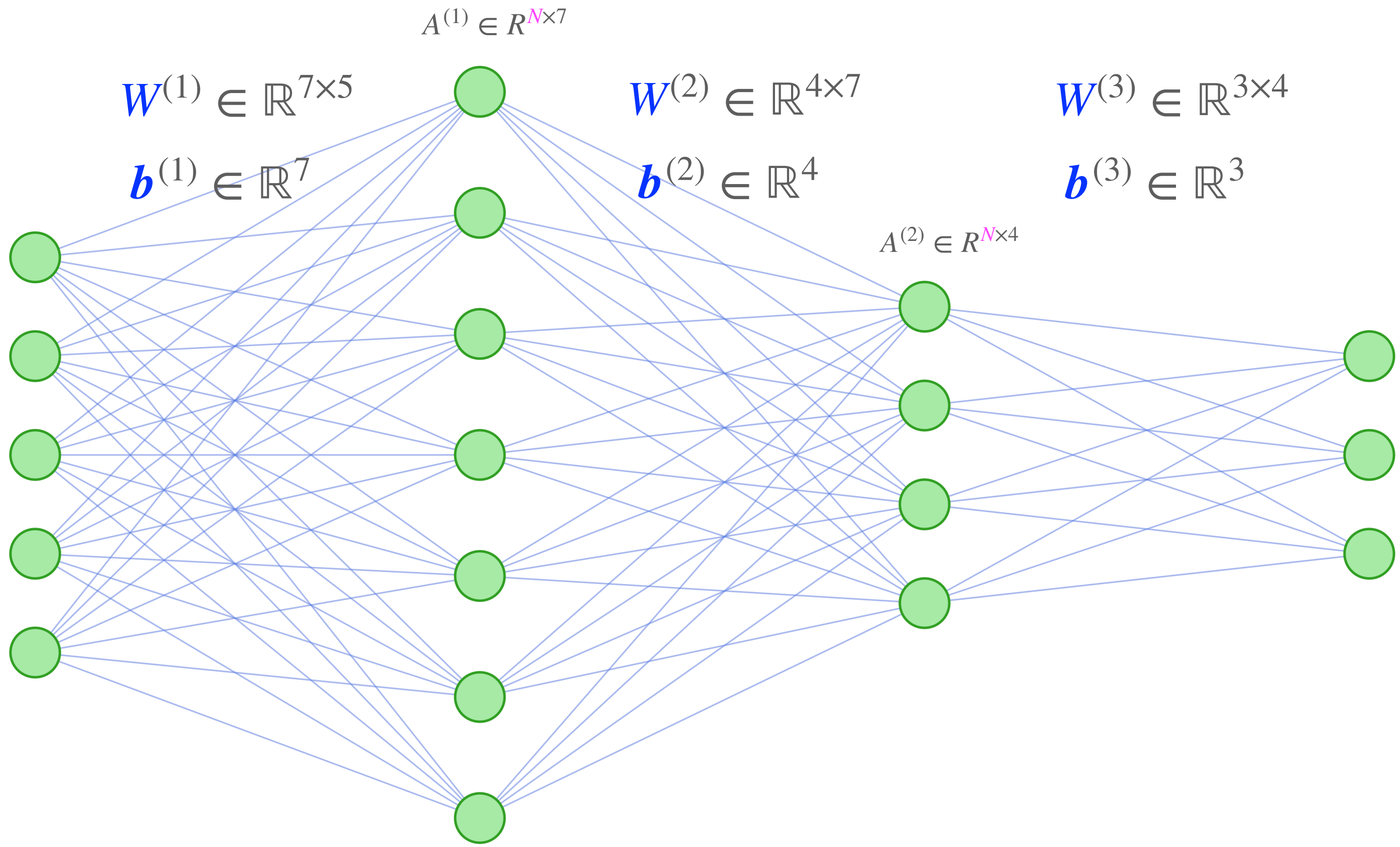
$z^{(3)} = \boldsymbol{a}^{(2)} W^{(3)^T} + \boldsymbol{b}^{(3)} \in \mathbb{R}^{1\times3}$

$\boldsymbol{a}^{(3)} = g_3(z^{(3)}) \in \mathbb{R}^{1\times3}$

Forward propagation

MLP

$\mathbb{R}^{N\times 5} \ni \boldsymbol{X}$



$A^{(3)} \in \mathbb{R}^{N\times 3}$

Efficient implementation

$X \in \mathbb{R}^{N\times 5}$

$Z^{(1)} = X\, W^{(1)^T} + \boldsymbol{b}^{(1)} \in \mathbb{R}^{N\times 7}$

$A^{(1)} = g_1(Z^{(1)}) \in \mathbb{R}^{N\times 7}$

$Z^{(2)} = A^{(1)}\, W^{(2)^T} + \boldsymbol{b}^{(2)} \in \mathbb{R}^{N\times 4}$

$A^{(2)} = g_2(Z^{(2)}) \in \mathbb{R}^{N\times 4}$

$Z^{(3)} = A^{(2)}\, W^{(3)^T} + \boldsymbol{b}^{(3)} \in \mathbb{R}^{N\times 3}$

$A^{(3)} = g_3(Z^{(3)}) \in \mathbb{R}^{N\times 3}$

Let's calculate the derivatives

$\frac{\partial L}{\partial A^{(p)}} = ?$

$$L = \frac{1}{N}\sum_{p=1}^{N} div(\hat{y}_p, y_p)$$

- Observe that the total loss is the average of the losses of independent samples
- One can do calculations row-wise, i.e.

$$\frac{\partial L}{\partial A^{(3)}} = -\frac{1}{N}\begin{bmatrix} \frac{\partial L}{\partial a_1^{(3)}} \\ \frac{\partial L}{\partial a_2^{(3)}} \\ \vdots \\ \frac{\partial L}{\partial a_N^{(3)}} \end{bmatrix}, \quad \frac{\partial L}{\partial Z^{(3)}} = -\frac{1}{N}\begin{bmatrix} \frac{\partial L}{\partial z_1^{(3)}} \\ \frac{\partial L}{\partial z_2^{(3)}} \\ \vdots \\ \frac{\partial L}{\partial z_N^{(3)}} \end{bmatrix}, \quad \bullet\bullet\bullet$$

Let's calculate the derivatives

Single sample (row)

$$L = \frac{1}{N}\sum_{p=1}^{N} div(\hat{y}_p, y_p) = \frac{1}{N}\sum_{p=1}^{N} L_p$$

$$y^T := Y[1, :]$$

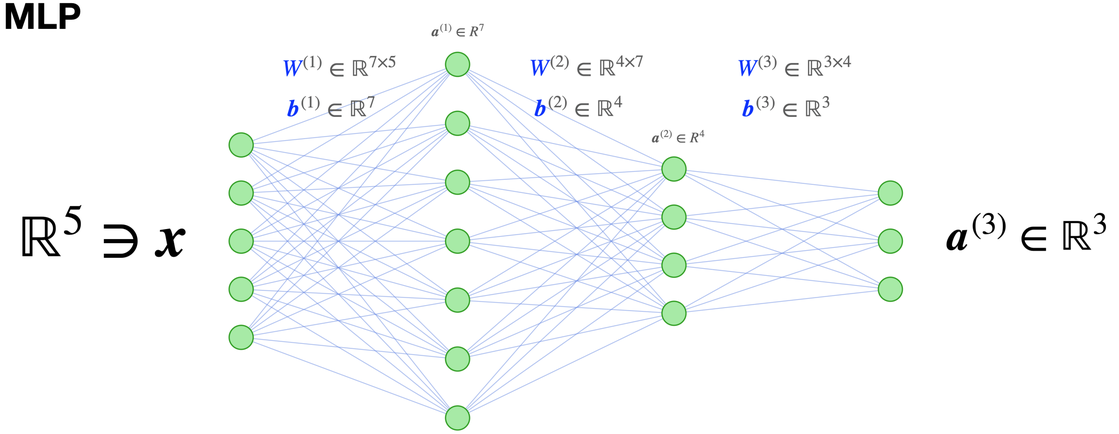
$$a^{(3)^T} := A^{(3)}[1, :] = a_1^{(3)}$$

$$z^{(3)^T} := Z^{(3)}[1, :]$$

$$\frac{\partial L}{\partial a^{(3)}} = ?, \frac{\partial L}{\partial z^{(3)}} = ?, \frac{\partial L}{\partial a^{(2)}} = ?, \frac{\partial L}{\partial z^{(2)}} = ?, \frac{\partial L}{\partial a^{(1)}} = ?, \frac{\partial L}{\partial z^{(1)}} = ?,$$

Forward propagation

MLP



Efficient implementation

$x^T \in \mathbb{R}^{1\times5}$

$z^{(1)} = x^T W^{(1)^T} + b^{(1)} \in \mathbb{R}^{1\times7}$

$a^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times7}$

$z^{(2)} = a^{(1)} W^{(2)^T} + b^{(2)} \in \mathbb{R}^{1\times4}$

$a^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times4}$

$z^{(3)} = a^{(2)} W^{(3)^T} + b^{(3)} \in \mathbb{R}^{1\times3}$

$a^{(3)} = \text{softmax}(z^{(3)}) \in \mathbb{R}^{1\times3}$

Let’s calculate the derivatives

Case: MSE loss + “id” activation function

$$\boldsymbol{x}^T \in \mathbb{R}^{1\times 5}$$

$$z^{(1)} = \boldsymbol{x}^T W^{(1)^T} + \boldsymbol{b}^{(1)} \in \mathbb{R}^{1\times 7} \qquad \boldsymbol{a}^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times 7}$$

$$z^{(2)} = \boldsymbol{a}^{(1)} W^{(2)^T} + \boldsymbol{b}^{(2)} \in \mathbb{R}^{1\times 4} \qquad \boldsymbol{a}^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times 4}$$

$$z^{(3)} = \boldsymbol{a}^{(2)} W^{(3)^T} + \boldsymbol{b}^{(3)} \in \mathbb{R}^{1\times 3} \qquad \boldsymbol{a}^{(3)} = \mathrm{id}(z^{(3)}) \in \mathbb{R}^{1\times 3}$$

$$\boldsymbol{y}^T := Y[1, :]$$

$$\boldsymbol{a}^{(3)^T} := A^{(3)}[1, :]$$

$$L_1 := \frac{1}{2} \|(Y[1, :] - A^{(3)}[1, :]\|_2^2$$

- $$\frac{\partial L_1}{\partial \boldsymbol{a}^{(3)}} = ?$$

Let's calculate the derivatives

Case: MSE loss + "id" activation function

$$\boldsymbol{x}^T \in \mathbb{R}^{1\times 5}$$

$$z^{(1)} = \boldsymbol{x}^T W^{(1)^T} + \boldsymbol{b}^{(1)} \in \mathbb{R}^{1\times 7} \qquad \boldsymbol{a}^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times 7}$$

$$z^{(2)} = \boldsymbol{a}^{(1)} W^{(2)^T} + \boldsymbol{b}^{(2)} \in \mathbb{R}^{1\times 4} \qquad \boldsymbol{a}^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times 4}$$

$$z^{(3)} = \boldsymbol{a}^{(2)} W^{(3)^T} + \boldsymbol{b}^{(3)} \in \mathbb{R}^{1\times 3} \qquad \boldsymbol{a}^{(3)} = \mathrm{id}(z^{(3)}) \in \mathbb{R}^{1\times 3}$$

$$\boldsymbol{y}^T := Y[1, :]$$

$$\boldsymbol{a}^{(3)^T} := A^{(3)}[1, :]$$

$$L_1 := \frac{1}{2} \|(Y[1, :] - A^{(3)}[1, :]\|_2^2$$

- $$\frac{\partial L_1}{\partial \boldsymbol{a}^{(3)}} = \boldsymbol{a}^{(3)^T} - \boldsymbol{y}^T$$

$$\frac{\partial L_1}{\partial z^{(3)}} = \frac{\partial L_1}{\partial \boldsymbol{a}^{(3)}} \cdot \frac{\partial \boldsymbol{a}^{(3)}}{\partial z^{(3)}} =$$

Let's calculate the derivatives

Case: MSE loss + "id" activation function

$$x^T \in \mathbb{R}^{1\times 5}$$

$$z^{(1)} = x^T W^{(1)^T} + b^{(1)} \in \mathbb{R}^{1\times 7} \qquad a^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times 7}$$

$$z^{(2)} = a^{(1)} W^{(2)^T} + b^{(2)} \in \mathbb{R}^{1\times 4} \qquad a^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times 4}$$

$$z^{(3)} = a^{(2)} W^{(3)^T} + b^{(3)} \in \mathbb{R}^{1\times 3} \qquad a^{(3)} = \mathrm{id}(z^{(3)}) \in \mathbb{R}^{1\times 3}$$

$$y^T := Y[1, :]$$

$$a^{(3)^T} := A^{(3)}[1, :]$$

$$L_1 := \frac{1}{2} \|(Y[1, :] - A^{(3)}[1, :]\|_2^2$$

- $$\frac{\partial L_1}{\partial a^{(3)}} = a^{(3)^T} - y^T$$

$$\frac{\partial L_1}{\partial z^{(3)}} = \frac{\partial L_1}{\partial a^{(3)}} \cdot \frac{\partial a^{(3)}}{\partial z^{(3)}} = a^{(3)^T} - y^T$$

Let's calculate the derivatives

Case: MSE loss + "id" activation function

$$x^T \in \mathbb{R}^{1\times 5}$$

$$z^{(1)} = x^T W^{(1)^T} + b^{(1)} \in \mathbb{R}^{1\times 7} \qquad a^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times 7}$$

$$z^{(2)} = a^{(1)} W^{(2)^T} + b^{(2)} \in \mathbb{R}^{1\times 4} \qquad a^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times 4}$$

$$z^{(3)} = a^{(2)} W^{(3)^T} + b^{(3)} \in \mathbb{R}^{1\times 3} \qquad a^{(3)} = \mathrm{id}(z^{(3)}) \in \mathbb{R}^{1\times 3}$$

$$y^T := Y[1, :]$$

$$a^{(3)^T} := A^{(3)}[1, :]$$

$$L_1 := \frac{1}{2} \|(Y[1, :] - A^{(3)}[1, :]\|_2^2$$

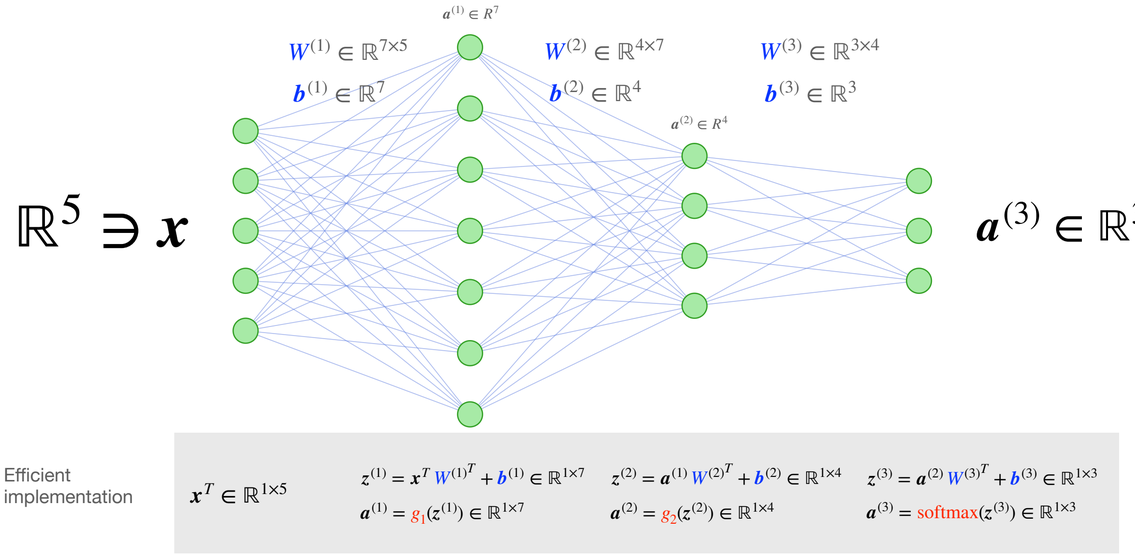
- $$\frac{\partial L_1}{\partial a^{(3)}} = a^{(3)^T} - y^T \qquad \frac{\partial L_1}{\partial z^{(3)}} = \frac{\partial L_1}{\partial a^{(3)}} \cdot \frac{\partial a^{(3)}}{\partial z^{(3)}} = a^{(3)^T} - y^T$$

Matrix case -> $$\frac{\partial L}{\partial Z^{(3)}} = \frac{1}{N}(A^{(3)} - Y)$$

Backward propagation

MLP

$$L_1 := \frac{1}{2} \|(Y[1, :] - A^{(3)}[1, :]\|_2^2$$



Efficient implementation

$\boldsymbol{x}^T \in \mathbb{R}^{1\times5}$

$z^{(1)} = \boldsymbol{x}^T W^{(1)^T} + \boldsymbol{b}^{(1)} \in \mathbb{R}^{1\times7}$
$\boldsymbol{a}^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times7}$

$z^{(2)} = \boldsymbol{a}^{(1)} W^{(2)^T} + \boldsymbol{b}^{(2)} \in \mathbb{R}^{1\times4}$
$\boldsymbol{a}^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times4}$

$z^{(3)} = \boldsymbol{a}^{(2)} W^{(3)^T} + \boldsymbol{b}^{(3)} \in \mathbb{R}^{1\times3}$
$\boldsymbol{a}^{(3)} = \text{softmax}(z^{(3)}) \in \mathbb{R}^{1\times3}$

$$\frac{\partial L_1}{\partial \boldsymbol{a}^{(3)}} = ???$$

$$\frac{\partial L_1}{\partial z^{(3)}} = \boldsymbol{a}^{(3)^T} - \boldsymbol{y}^T \in \mathbb{R}^{1\times3}$$

$$\frac{\partial L_1}{\partial W^{(3)}} = \frac{\partial L_1}{\partial z^{(3)}}^T \cdot \frac{\partial z^{(3)^T}}{\partial W^{(3)}} = \frac{\partial L_1}{\partial z^{(3)}}^T \cdot \boldsymbol{a}^{(2)^T}$$

$$\frac{\partial L_1}{\partial \boldsymbol{a}^{(2)}} = \frac{\partial L_1}{\partial z^{(3)}} \cdot \frac{\partial z^{(3)}}{\partial \boldsymbol{a}^{(2)}} = \frac{\partial L_1}{\partial z^{(3)}} \cdot W^{(3)}$$

$$\frac{\partial L_1}{\partial z^{(2)}} = \frac{\partial L_1}{\partial \boldsymbol{a}^{(2)}} \cdot \frac{\partial \boldsymbol{a}^{(2)}}{\partial z^{(2)}} = \frac{\partial L_1}{\partial \boldsymbol{a}^{(2)}} \otimes \frac{\partial g_2}{\partial z^{(2)}}$$

$$\frac{\partial L_1}{\partial W^{(2)}} = \frac{\partial L_1}{\partial z^{(2)}}^T \cdot \frac{\partial z^{(2)}}{\partial W^{(2)}} = \frac{\partial L_1}{\partial z^{(2)}}^T \cdot \boldsymbol{a}^{(1)^T}$$

$$\frac{\partial L_1}{\partial \boldsymbol{a}^{(1)}} = \frac{\partial L_1}{\partial z^{(2)}} \cdot \frac{\partial z^{(2)}}{\partial \boldsymbol{a}^{(1)}} = \frac{\partial L_1}{\partial z^{(2)}} \cdot W^{(2)}$$

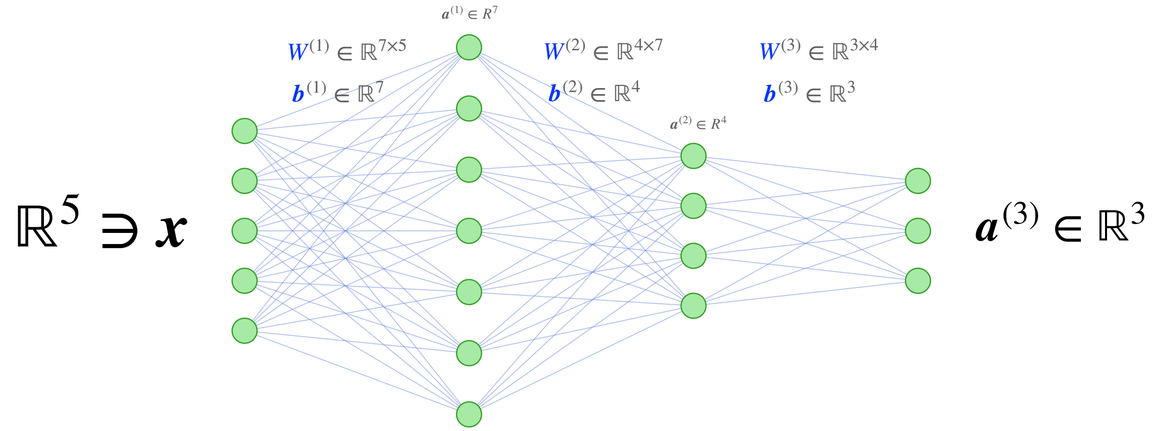
$$\frac{\partial L_1}{\partial z^{(1)}} = \frac{\partial L_1}{\partial \boldsymbol{a}^{(1)}} \cdot \frac{\partial \boldsymbol{a}^{(1)}}{\partial z^{(1)}} = \frac{\partial L_1}{\partial \boldsymbol{a}^{(1)}} \otimes \frac{\partial g_1}{\partial z^{(1)}}$$

$$\frac{\partial L_1}{\partial W^{(1)}} = \frac{\partial L_1}{\partial z^{(1)}}^T \cdot \frac{\partial z^{(1)}}{\partial W^{(1)}} = \frac{\partial L_1}{\partial z^{(1)}}^T \cdot \boldsymbol{x}^T$$

Backward propagation

MLP

$$L_1 := \frac{1}{2} \|(Y[1, :] - A^{(3)}[1, :]\|_2^2$$



Efficient implementation

$x^T \in \mathbb{R}^{1\times5}$

$z^{(1)} = x^T W^{(1)^T} + b^{(1)} \in \mathbb{R}^{1\times7}$
$a^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times7}$

$z^{(2)} = a^{(1)} W^{(2)^T} + b^{(2)} \in \mathbb{R}^{1\times4}$
$a^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times4}$

$z^{(3)} = a^{(2)} W^{(3)^T} + b^{(3)} \in \mathbb{R}^{1\times3}$
$a^{(3)} = \text{softmax}(z^{(3)}) \in \mathbb{R}^{1\times3}$

$$\frac{\partial L_1}{\partial W^{(3)}} = \frac{\partial L_1}{\partial z^{(3)}}^T \cdot \frac{\partial z^{(3)^T}}{\partial W^{(3)}} = \frac{\partial L_1}{\partial z^{(3)}}^T \cdot a^{(2)^T}$$

$$\frac{\partial L_1}{\partial W^{(2)}} = \frac{\partial L_1}{\partial z^{(2)}}^T \cdot \frac{\partial z^{(2)}}{\partial W^{(2)}} = \frac{\partial L_1}{\partial z^{(2)}}^T \cdot a^{(1)^T}$$

$$\frac{\partial L_1}{\partial W^{(1)}} = \frac{\partial L_1}{\partial z^{(1)}}^T \cdot \frac{\partial z^{(1)}}{\partial W^{(1)}} = \frac{\partial L_1}{\partial z^{(1)}}^T \cdot x^T$$

$$\frac{\partial L_1}{\partial b^{(3)}} = \frac{\partial L_1}{\partial z^{(3)}} \cdot \frac{\partial z^{(3)}}{\partial b^{(3)}} = \frac{\partial L_1}{\partial z^{(3)}}$$

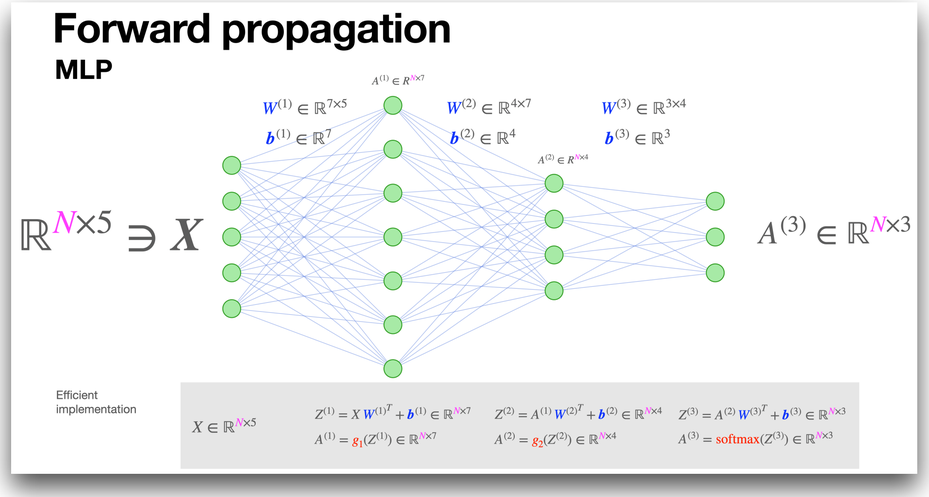
$$\frac{\partial L_1}{\partial b^{(2)}} = \frac{\partial L_1}{\partial z^{(2)}} \cdot \frac{\partial z^{(2)}}{\partial b^{(2)}} = \frac{\partial L_1}{\partial z^{(2)}}$$

$$\frac{\partial L_1}{\partial b^{(1)}} = \frac{\partial L_1}{\partial z^{(1)}} \cdot \frac{\partial z^{(1)}}{\partial b^{(1)}} = \frac{\partial L_1}{\partial z^{(1)}}$$

Backward propagation

MLP

$$L = -\frac{1}{N}\sum_{p=1}^{N} L_p$$



$$\frac{\partial L}{\partial W^{(3)}} = \frac{\partial L}{\partial Z^{(3)}}^{T} \cdot \frac{\partial Z^{(3)}}{\partial W^{(3)}} = \frac{\partial L}{\partial Z^{(3)}}^{T} \cdot Z^{(3)} \cdot \frac{\partial}{\partial W^{(3)}} = \frac{\partial L}{\partial Z^{(3)}}^{T} \cdot A^{(2)}$$

$$\frac{\partial L_1}{\partial \boldsymbol{b}^{(3)}} = \frac{\partial L_1}{\partial Z^{(3)}}^{T} \cdot \frac{\partial Z^{(3)}}{\partial \boldsymbol{b}^{(3)}} = \frac{\partial L_1}{\partial Z^{(3)}}^{T} \cdot \mathbf{1}^N$$

$$\frac{\partial L}{\partial W^{(2)}} = \frac{\partial L}{\partial Z^{(2)}}^{T} \cdot \frac{\partial Z^{(2)}}{\partial W^{(2)}} = \frac{\partial L}{\partial Z^{(2)}}^{T} \cdot Z^{(2)} \cdot \frac{\partial}{\partial W^{(2)}} = \frac{\partial L}{\partial Z^{(2)}}^{T} \cdot A^{(1)}$$

$$\frac{\partial L_1}{\partial \boldsymbol{b}^{(2)}} = \frac{\partial L_1}{\partial Z^{(2)}}^{T} \cdot \frac{\partial Z^{(2)}}{\partial \boldsymbol{b}^{(2)}} = \frac{\partial L_1}{\partial Z^{(2)}}^{T} \cdot \mathbf{1}^N$$

$$\frac{\partial L}{\partial W^{(1)}} = \frac{\partial L}{\partial Z^{(1)}}^{T} \cdot \frac{\partial Z^{(1)}}{\partial W^{(1)}} = \frac{\partial L}{\partial Z^{(1)}}^{T} \cdot Z^{(1)} \cdot \frac{\partial}{\partial W^{(1)}} = \frac{\partial L}{\partial Z^{(1)}}^{T} \cdot X$$

$$\frac{\partial L_1}{\partial \boldsymbol{b}^{(1)}} = \frac{\partial L_1}{\partial Z^{(1)}}^{T} \cdot \frac{\partial Z^{(1)}}{\partial \boldsymbol{b}^{(1)}} = \frac{\partial L_1}{\partial Z^{(1)}}^{T} \cdot \mathbf{1}^N$$

Let’s calculate the derivatives

Case: cross entropy loss / “sigmoid” activation function

$$\boldsymbol{x}^T \in \mathbb{R}^{1\times 5}$$

$$z^{(1)} = \boldsymbol{x}^T W^{(1)^T} + \boldsymbol{b}^{(1)} \in \mathbb{R}^{1\times 7}$$

$$\boldsymbol{a}^{(1)} = g_1(z^{(1)}) \in \mathbb{R}^{1\times 7}$$

$$z^{(2)} = \boldsymbol{a}^{(1)} W^{(2)^T} + \boldsymbol{b}^{(2)} \in \mathbb{R}^{1\times 4}$$

$$\boldsymbol{a}^{(2)} = g_2(z^{(2)}) \in \mathbb{R}^{1\times 4}$$

$$z^{(3)} = \boldsymbol{a}^{(2)} W^{(3)^T} + \boldsymbol{b}^{(3)} \in \mathbb{R}^{1\times 3}$$

$$\boldsymbol{a}^{(3)} = \sigma(z^{(3)}) \in \mathbb{R}^{1\times 3}$$

Logistic regression

Derivative of loss function

$$z = \boldsymbol{x}\boldsymbol{w}^T + b, \qquad \hat{y} = \sigma(z), \qquad \frac{\partial \hat{y}}{\partial z} = \sigma(z)(1 - \sigma'(z))$$

$$L(\boldsymbol{w}, b) = -\, y \log \hat{y} - (1 - y)\log(1 - \hat{y})$$

$$\frac{\partial L}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{1 - y}{1 - \hat{y}}\,, \quad \frac{\partial L}{\partial z} = \hat{y} - y\,, \quad \frac{\partial L}{\partial \boldsymbol{w}} = (\hat{y} - y)\,\boldsymbol{x}^T\,, \quad \frac{\partial L}{\partial b} = \hat{y} - y$$