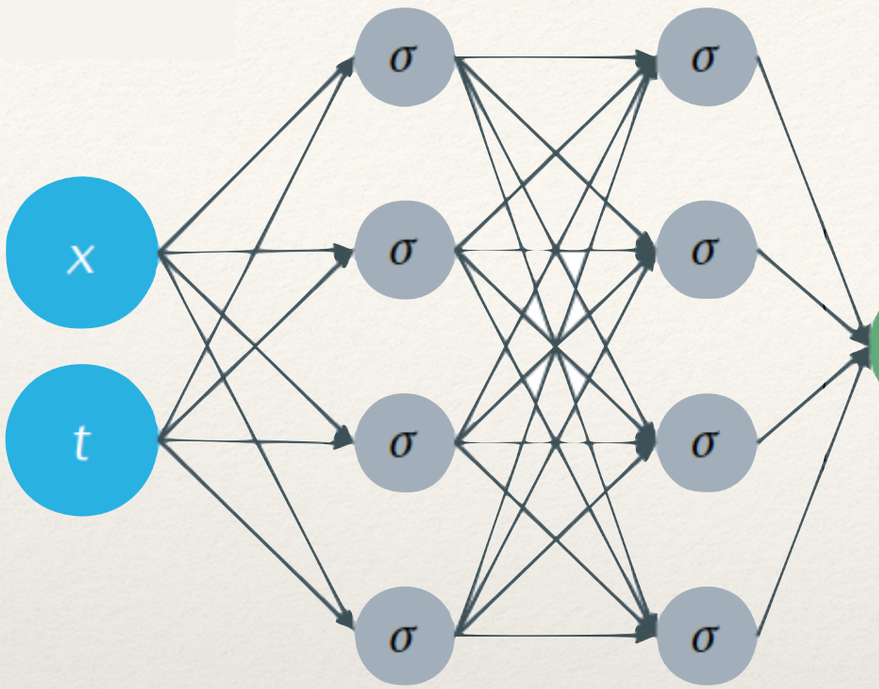
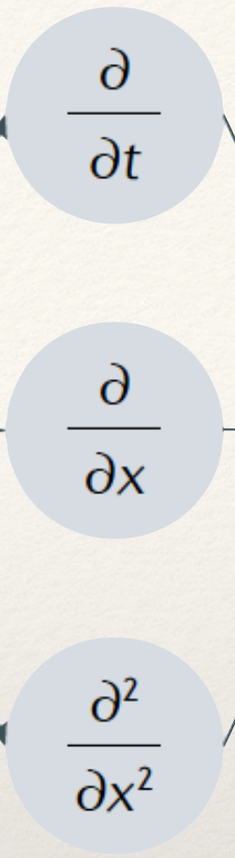
1. Neural network formation



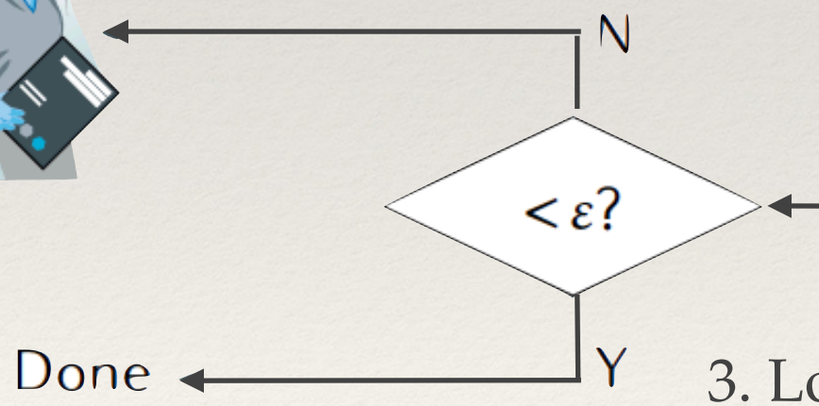
2. Physical laws (differential equation)

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2}$$



$$loss_d = [u(0) - u_0]^2$$

$$loss_e = \sum_j^N f^2(x_j, u(x_j))$$



3. Loss function: $loss = (1-\lambda)loss_d + \lambda loss_e$

4. Optimization $w^{(i+1)} = w^{(i)} - \eta^{(i)}\nabla_w J(\mathbf{x}, w^{(i)})$

Physics-informed Neural network

1. Automatic differentiation

2. Weight initialization and normalization

$$x_j^{(n)} = h\left(\sum_{i=1} w_{ji}^{(n-1)} x_i + b_{j0}^{(n-1)}\right)$$

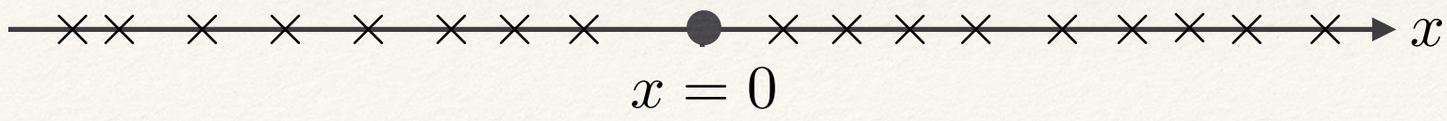
This lecture

For **all** neural network

3. Weight of equation loss

$$loss = (1-\lambda) loss_d + \lambda loss_e$$

4. Collocation point distribution



Next lecture

Particular for physics-informed neural network

Different approaches of differentiation

$$l_1 = x$$
$$l_{n+1} = 4l_n(1 - l_n)$$

$$f(x) = l_4 = 64x(1-x)(1-2x)^2(1-8x+8x^2)^2$$

Manual or symbolic differentiation

$$f'(x) = 128x(1-x)(-8+16x)(1-2x)^2(1-8x+8x^2)+64(1-x)(1-2x)^2(1-8x+8x^2)^2-64x(1-2x)^2(1-8x+8x^2)^2-256x(1-x)(1-2x)(1-8x+8x^2)^2$$

Accurate but time-consuming

Different approaches of differentiation

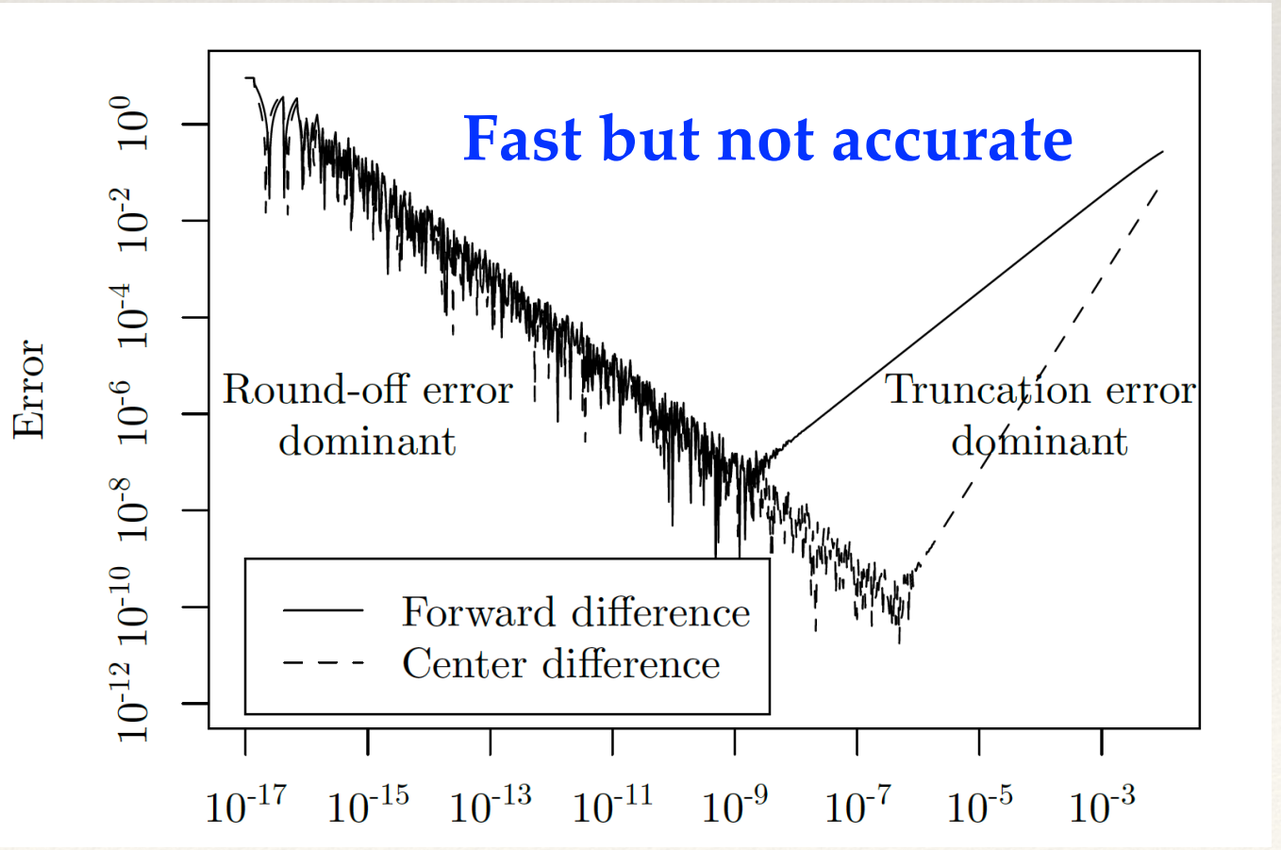
$$l_1 = x$$
$$l_{n+1} = 4l_n(1 - l_n)$$

$$f(x) = l_4 = 64x(1-x)(1-2x)^2(1-8x+8x^2)^2$$

Numerical differentiation

$$\frac{\partial f(\mathbf{x})}{\partial x_i} \approx \frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h}$$

```
f'(x):
    h = 0.000001
    return (f(x + h) - f(x)) / h
```



Automatic differentiation

$$y = f(x) = \sin(\ln(1 + x^2))$$

compute dy/dx at $x = 1$

Forward mode $\qquad \dot{v}_i = \dfrac{dv_i}{dx}$

$$\begin{aligned}
v_0 &= x & &= 1 & \dot{v}_0 &= 1 & \\
v_1 &= 1 + v_0^2 & &= 2 & \dot{v}_1 &= 2v_0\dot{v}_0 & &= 2 \\
v_2 &= \ln(v_1) & &= \ln 2 & \dot{v}_2 &= \frac{1}{v_1}\dot{v}_1 & &= 1 \\
y = v_3 &= \sin(v_2) & &= \sin(\ln 2) & \dot{y} = \dot{v}_3 &= \cos(v_2)\dot{v}_2 & &= \cos(\ln 2)
\end{aligned}$$

Automatic differentiation

$$y = f(x) = \sin(\ln(1 + x^2))$$

compute dy/dx at $x = 1$

Backward mode $\quad \hat{v_i} = \dfrac{dy}{dv_i}$

$$\begin{aligned} v_0 &= x & &= 1 \\ v_1 &= 1 + v_0^2 & &= 2 \\ v_2 &= \ln(v_1) & &= \ln 2 \\ y = v_3 &= \sin(v_2) & &= \sin(\ln 2) \end{aligned}$$

$$\begin{aligned} \hat{v_0} &= \frac{dv_1}{dv_0}\hat{v_1} & &= 2v_0\frac{\cos v_2}{v_1} \\ \hat{v_1} &= \frac{dy}{dv_1} = \frac{dv_2}{dv_1}\hat{v_2} & &= \frac{\cos v_2}{v_1} \\ \hat{v_2} &= \frac{dy}{dv_2} = \frac{dv_3}{dv_2}\hat{v_3} & &= \cos(v_2) \\ \hat{v_3} &= \frac{dy}{dv_3} = 1 \end{aligned}$$

Automatic differentiation

$$y = f(x) = v_5(v_4(v_3(v_2(v_1(v_0(x))))))$$

Chain rule:

$$\frac{dy}{dx} = \frac{dy}{dv_5} \cdot \frac{dv_5}{dv_4} \cdots \frac{dv_0}{dx}$$

Forward mode

$$\frac{dy}{dx} = \frac{dy}{dv_5} \cdots \frac{dv_1}{dv_0} \cdot \frac{dv_0}{dx}$$

Backward mode

$$\frac{dy}{dx} = \frac{dy}{dv_5} \cdot \frac{dv_5}{dv_4} \cdots \frac{dv_0}{dx}$$

Different approaches of differentiation

$$l_1 = x$$
$$l_{n+1} = 4l_n(1 - l_n)$$

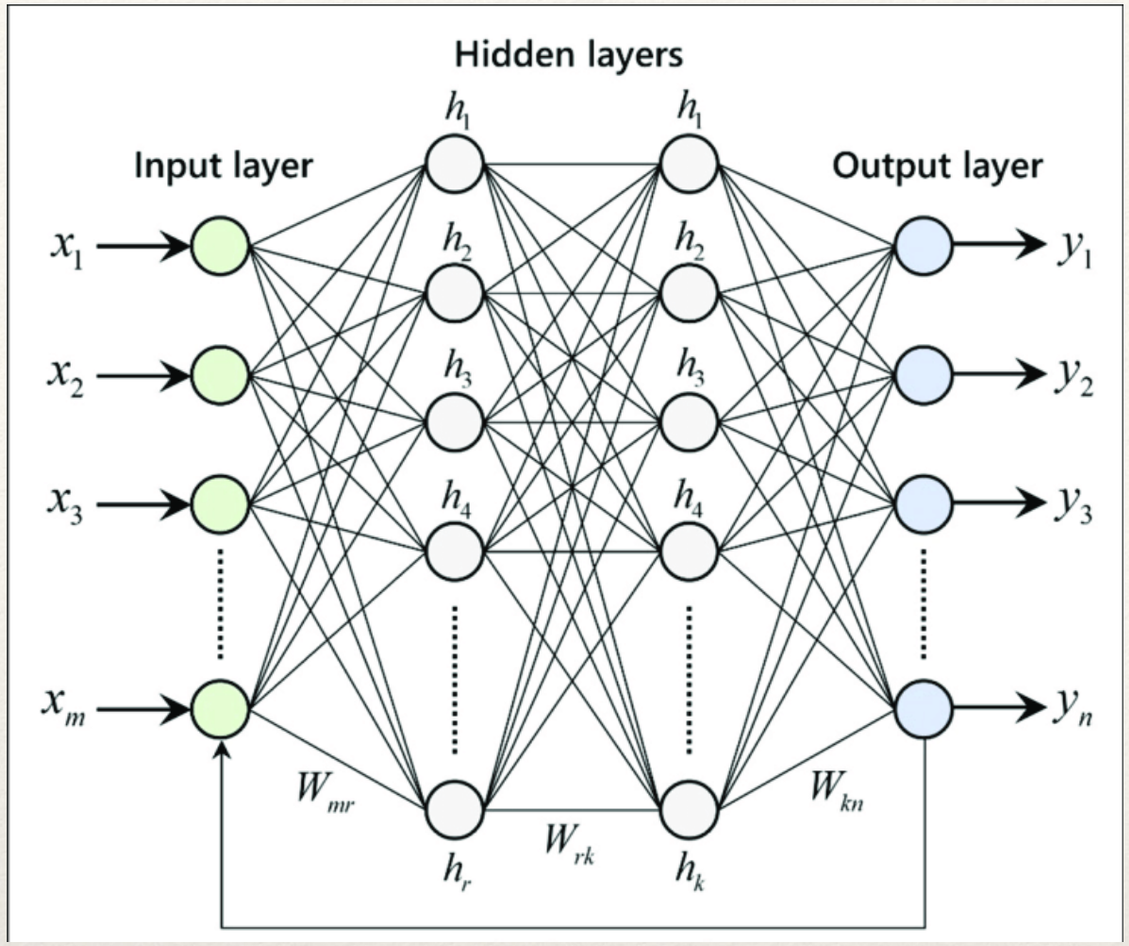
$$f(x) = l_4 = 64x(1-x)(1-2x)^2(1-8x+8x^2)^2$$

Questions

Using python to write a function using for-loop and the idea of automatic differentiation that allow to compute the derivative of f(x) at any x

Try both ***forward*** and ***backward*** mode

```
def fdot(x):
    f = x
    df = 1
    for i = 1 to 3:
        f = # to do#
        df = # to do #
    return df
```



Fully-connected neural network

$$x_j^{(n)} = h\left(\sum_{i=1} w_{ji}^{(n-1)} x_i + b_{j0}^{(n-1)}\right)$$

Loss function

$$J(\mathbf{y}(\mathbf{x}, w)) = \frac{1}{N}\sum_{j=1}^{N} J(y_j(x_j, w))$$

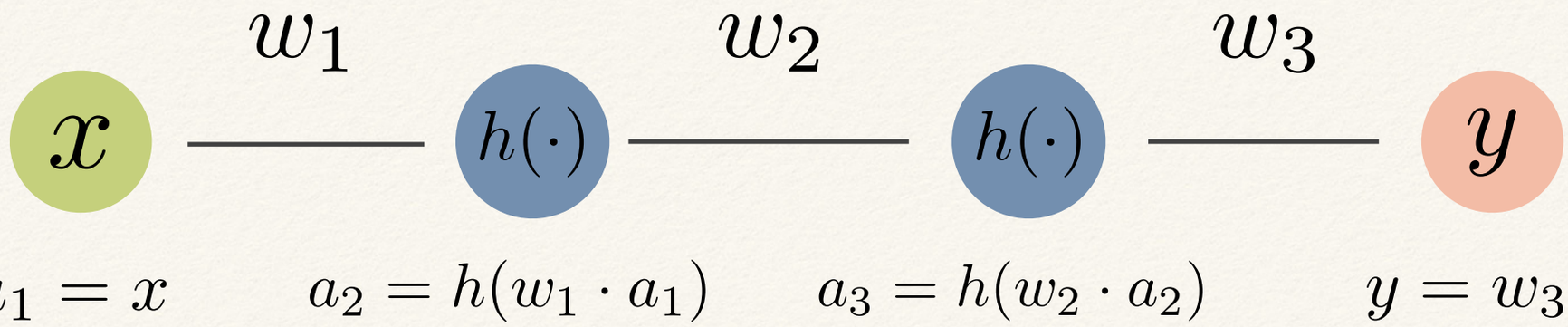
Optimization:

$$w^{(i+1)} = w^{(i)} - \eta^{(i)} \nabla_w J(\mathbf{x}, w^{(i)})$$

Calculate the derivative of J(x,w) with respect to each w

Automatic differentiation

Forward mode ? Backward mode



Entire neural network

$$y = w_3 \cdot h[w_2 \cdot h(w_1 \cdot x)]$$

Loss function: mean squared error

$$J(y(x, w)) = (y(x, w) - y_0)^2$$

$$\frac{dJ}{dw_3} = 2(y - y_0)\frac{dy}{dw_3} = 2(y - y_0)a_3$$

Computational efficient

$$\frac{dJ}{dw_2} = 2(y - y_0)\frac{dy}{dw_2} = 2(y - y_0)w_3\frac{da_3}{dw_2} = 2(y - y_0) \cdot w_3 h'(w_2 a_2) \cdot a_2$$

$$\frac{dJ}{dw_1} = 2(y - y_0)\frac{dy}{dw_1} = 2(y - y_0)w_3\frac{da_3}{dw_1} = 2(y - y_0) \cdot w_3 h'(w_2 a_2) \cdot w_2\frac{da_2}{dw_1}$$

$$h'(w_1 a_1)a_1$$