Last time

- Backpropagation
  1. (1) represent function as graph
  2. (2) define local derivatives
  3. (3) evaluate function (forward pass)
  4. (4) use chain rule to obtain gradients (backward pass)

- Chain rule
  - univariate: $\frac{d}{dt} f(g(t)) = \frac{df}{dg} \cdot \frac{dg}{dt} = f'(g(t)) \cdot g'(t)$
  - multivariate: $\frac{d}{dt} f(g_1(t), \ldots, g_p(t)) = \sum_{i=1}^{p} \frac{\partial f}{\partial g_i} \cdot \frac{dg_i}{dt}$

- optimization demo

Today

- Backpropagation example
- neural networks

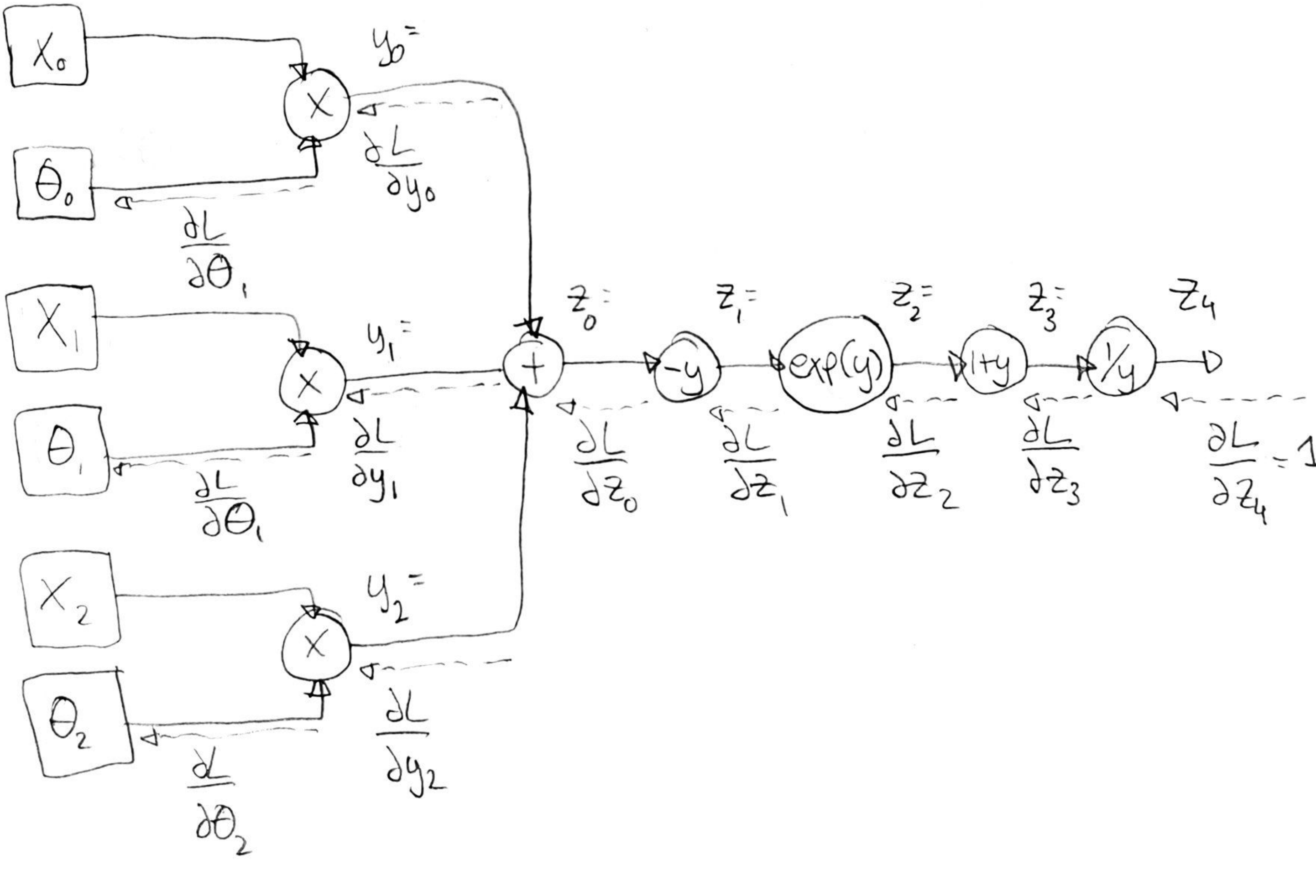
Backpropagation Example

$\rightarrow$ evaluate $\left[\nabla_\theta L(\theta, D) \right]_{\theta = \theta^{(t)}}$ at $\theta^{(t)} = [\theta_0, \theta_1, \theta_2]$

$= [-1, 4, -2]$

where $L(\theta, D) = \dfrac{1}{1 + \exp(-X^T\theta)}$ and $X = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} X_0 \\ X_1 \\ X_2 \end{bmatrix}$

Step I: construct graph



Step 2: Define local derivatives

	node	local gradients		evaluated (4a)	
A	$f(x_0, \theta_0) = x_0\theta_0 = y_0$	$\frac{\partial y_0}{\partial x_0} = \theta_0$	$\frac{\partial y_0}{\partial \theta_0} = x_0$	$\frac{\partial y_0}{\partial x_0} = -1$	$\frac{\partial y_0}{\partial \theta_0} = 1$
B	$f(x_1, \theta_1) = x_1\theta_1 = y_1$	$\frac{\partial y_1}{\partial x_1} = \theta_1$	$\frac{\partial y_1}{\partial \theta_1} = x_1$	$\frac{\partial y_1}{\partial x_1} = 4$	$\frac{\partial y_1}{\partial \theta_1} = 2$
C	$f(x_2, \theta_2) = x_2\theta_2 = y_2$	$\frac{\partial y_2}{\partial x_2} = \theta_2$	$\frac{\partial y_2}{\partial \theta_2} = x_2$	$\frac{\partial y_2}{\partial x_2} = -2$	$\frac{\partial y_2}{\partial \theta_2} = 3$
D	$f(y_0, y_1, y_2) = \sum_{i=0}^{2} y_i = z_0$	$\frac{\partial z_0}{\partial y_i} = 1$		$\frac{\partial z_0}{\partial y_i} = 1$	
E	$f(z_0) = -z_0 = z_1$	$\frac{dz_1}{dz_0} = -1$		$\frac{dz_1}{dz_0} = -1$	
F	$f(z_1) = \exp(z_1) = z_2$	$\frac{dz_2}{dz_1} = \exp(z_1)$		$\frac{dz_2}{dz_1} = \exp(-1) = \frac{1}{e} = 0.3678$	
G	$f(z_2) = 1 + z_2 = z_3$	$\frac{dz_3}{dz_2} = 1$		$\frac{dz_3}{dz_2} = 1$	
H	$f(z_3) = 1/z_3 = z_4$	$\frac{dz_4}{dz_3} = \frac{-1}{z_3^2}$		$\frac{dz_4}{dz_3} = \frac{-1}{(1+\frac{1}{e})^2} = -0.53445$	

Step 3: Evaluate function

node	forward pass	
A	$1(-1) = -1$	y_0
B	$2(4) = 8$	y_1
C	$3(-2) = -6$	y_2
D	$-1 + 8 - 6 = 1$	z_0
E	-1	z_1
F	$\exp(-1) = \frac{1}{e} = 0.36787944$	z_2
G	$1 + \frac{1}{e} = 1.36787944$	z_3
H	$\frac{1}{1+(\frac{1}{e})} = 0.7310586$	z_4

Step 4

(A) Evaluate local gradients

(B) Backpropagate gradients using the chain rule

(4b)

$$\frac{\partial L}{\partial z_4} = 1$$

$$\frac{\partial L}{\partial z_3} = \frac{\partial L}{\partial z_4} \cdot \frac{dz_4}{dz_3} = (1)(-0.53445) = -0.53445$$

$$\frac{\partial L}{\partial z_2} = \frac{\partial L}{\partial z_3} \cdot \frac{dz_3}{dz_2} = (-.53445)(1) = -0.53445$$

$$\frac{\partial L}{\partial z_1} = \frac{\partial L}{\partial z_2} \cdot \frac{dz_2}{dz_1} = (-.53445)(0.3678) = -0.196612$$

$$\frac{\partial L}{\partial z_0} = \frac{\partial L}{\partial z_1} \cdot \frac{dz_1}{dz_0} = (-0.196612)(-1) = 0.196612$$

$$\frac{\partial L}{\partial y_0} = \frac{\partial L}{\partial z_0} \cdot \frac{\partial z_0}{\partial y_0} = (0.196612)(1) = 0.196612$$

$$\frac{\partial L}{\partial y_1} = \frac{\partial L}{\partial z_0} \cdot \frac{\partial z_0}{\partial y_1} = (0.196612)(1) = 0.196612$$

$$\frac{\partial L}{\partial y_2} = \frac{\partial L}{\partial z_0} \cdot \frac{\partial z_0}{\partial y_2} = (0.196612)(1) = 0.196612$$

$$\frac{\partial L}{\partial \theta_0} = \frac{\partial L}{\partial y_0} \cdot \frac{\partial y_0}{\partial \theta_0} = (0.196612)(1) = 0.196612$$

$$\frac{\partial L}{\partial \theta_1} = \frac{\partial L}{\partial y_1} \cdot \frac{\partial y_1}{\partial \theta_1} = (0.196612)(2) = 0.39322$$

$$\frac{\partial L}{\partial \theta_2} = \frac{\partial L}{\partial y_2} \cdot \frac{\partial y_2}{\partial \theta_2} = (0.196612)(3) = 0.58984$$

so, $$\left[\nabla_\theta L(\theta, D)\right]_{\theta=\theta^{(t)}} = \begin{bmatrix} 0.196612 \\ 0.39322 \\ 0.58983 \end{bmatrix}$$

Core Problem:

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}}\ L(f, D) \qquad (1)$$

$$\Longleftrightarrow \quad \theta^* = \underset{\theta \in \Theta}{\operatorname{argmin}}\ L(\theta, D) \qquad (2)$$

$$\approx \quad \hat{\theta} = \underset{\theta \in \Theta}{\operatorname{opt}}\ L(\theta, D) \qquad (3)$$

we have all the tools now to solve (3) for arbitrarily complex $\mathcal{F}$ (given $f \in \mathcal{F}$ are differentiable).

Note: if f not differentiable? reinforcemet learning

Next, we will consider neural networks, one such (large) class $\mathcal{F}$.

Neural Networks

- a family of functions $\mathcal{F}$
- biologically inspired
- $f \in \mathcal{F}$ are characterized by their modular lego-like form
  - each module is called a <u>layer</u> (L_j) (for now)
  - each layer has two components:
    (1) linear piece
    (2) non-linear piece (called the <u>activation function</u>)
- the full function f is a composition of several of these layers. The number of layers making up the neural network is called its <u>depth</u>.

$$\text{e.g. } f(X) = L_2(X, L_1(L_0(X)))$$

- next we will see several important layers