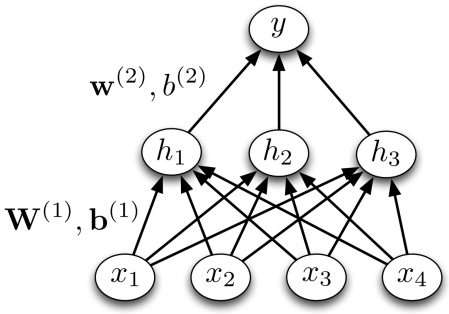
# MATH 629 Homework 1

Due Date: October 25 2024, 2:00 PM

## Problem 1

Find a set of weights and biases for a multilayer perceptron which determines if a list of length 4 is in sorted order. More specifically, you receive four inputs where $x_1, x_2, x_3$, and $x_4$, where $x_i \in \mathcal{R}$. The network should output 1 if $x_1 < x_2 < x_3 < x_4$, and 0 otherwise. Use the following architecture:




All of the hidden units and the output use a hard threshold activation function:

$$\phi(z) = \begin{cases} 1 & z \geq 0 \\ 0 & z \leq 0 \end{cases}$$

Please give a set of weights and biases for the network which correctly implements this function (including cases where some of the inputs are equal). Your answer should include:

- A weight matrix $\mathbf{W}^{[1]}$ for the hidden layer (dimension is $3 \times 4$).
- A bias vector $\mathbf{b}^{[1]}$ for hidden layer (dimension is 3).
- A 3-dimensional weight vector $\mathbf{W}^{[2]}$ for the output layer.
- A scalar bias $b^{[2]}$ for the output layer.

## Problem 2

Suppose we have a prediction problem where the target t corresponds to an angle, measured in radians. A reasonable loss function we might use is:

$$\mathcal{L}(y, t) = 1 - \cos(y - t)$$

As usual, the cost is the average loss over the training set:

$$\mathcal{E} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i, t_i)$$

Derive a sequence of vectorized mathematical expressions for the gradients of the cost with respect to $w$ and $b$. As usual, the inputs are organized into a design matrix $X$ with one row per training example. The

expressions should be something you can translate into a Python code without requiring a for-loop. Your answer should look like:

$$
\begin{aligned}
Y &= ... \\
\frac{\partial \mathcal{E}}{\partial Y} &= ... \\
\frac{\partial \mathcal{E}}{\partial W} &= ... \\
\frac{\partial \mathcal{E}}{\partial B} &= ...
\end{aligned}
$$

## Problem 3

Show that we can rewrite regression with *absolute* loss function $|y_n - \mathbf{w^T x_n}|$ as a re-weighted least squares objective where the squared loss term for each example $(\mathbf{X}_n, y_n)$ is multiplied by an importance weight $s_n > 0$.

- Write down the expression for $s_n$ and briefly explain why this expression makes intuitive sense.
- Given N examples $\{\mathbf{X}_n, y_n\}_{n=1}^{n=N}$ briefly outline the steps of an optimization algorithm that estimates the unknowns ($\mathbf{w}$ and the importance weights $\{s_n\}_{n=1}^{n=N}$) for this re-weighted least square problem.

## Problem 4

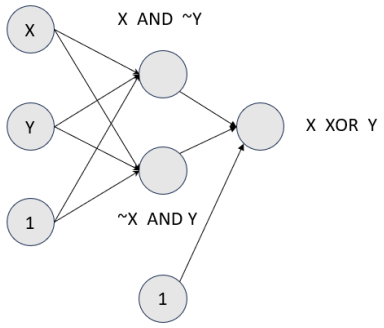
We know that a 2-layer neural network cannot implement the $XOR$ function. Using the network architecture below and the sigmoid activation function:

$$F = \frac{1}{1 + \exp^{(w_0 + w_1 x + w_2 y)}}$$

Find the connection weights only from the set {10,-10,100,-100} such that the $XOR$ function $x\ XOR\ y$ can be implemented. Use:

$$x\ XOR\ y = \begin{cases} > \frac{1}{2} & x \neq y \\ < \frac{1}{2} & otherwise \end{cases}$$

The meaninings of the hidden nodes are also given. The bias node gives a constant output of 1.
In your answer draw the neural network and write the weights on the connections.



## Problem 5

You will be filling in code to implement a perceptron from scratch. See the attached notebook.

## Problem 6

You will be filling in code to implement a multi-layer perceptron using PyTorch. See the attached notebook.