Consider the simple neural network that takes a scalar real input, has 1 hidden layer with \( k \) units in it and a sigmoid nonlinearity for those units, and an output linear (affine) layer to predict a scalar output. We can algebraically write any function that it represents as

\[
y_{\text{pred}} = C\sigma(Ax + b) + d
\]

The \( \sigma(.) \) represents an arbitrary nonlinearity, with derivative \( \sigma'(.) \). Where \( x \in \mathbb{R}, A \in \mathbb{R}^{k \times 1}, b \in \mathbb{R}^{k \times 1},\ C \in \mathbb{R}^{1 \times k}, d \in \mathbb{R}, \) and \( y_{\text{pred}} \in \mathbb{R} \). We can write it as \( y_{\text{pred}} = C\sigma(z) + d \), where \( z = Ax + b \) and the nonlinearity is applied element-wise. We have the true label \( y_{\text{true}} \) for each \( x \), and we use the L2 Loss \( L(y_{\text{true}}, y_{\text{pred}}) = (y_{\text{true}} - y_{\text{pred}})^2 \).

1. (a) Consider the sigmoid nonlinearity function \( \sigma(z) = \frac{1}{1 + e^{-z}} \). Show that \( \frac{d}{dz} \sigma(z) = \sigma(z)(1 - \sigma(z)) \)

(b) Calculate \( \frac{\partial L}{\partial d} \)

(c) Calculate \( \frac{\partial L}{\partial C} \)

(d) Calculate \( \frac{\partial L}{\partial b} \)

(e) Calculate \( \frac{\partial L}{\partial A} \)

(f) Write the gradient-descent update rule for \( b_{t+1} \) with learning rate \( \alpha \).
2. Given the Regularized Objective function:

\[ \arg\min_x \|Ax - b\|^2 + \lambda \|x\|^2 \]

Use vector calculus to find the closed form solution for \( x \). Interpret what this means in terms of the singular values.

3. Consider a simple neural network that spits out 1-dim values after a nonlinearity. These values for a batch are \{1, 7, 7, 9\}. What is the output of running batch norm with this data and \( \gamma = 1 \) and \( \beta = 0 \). In other words, standardize the data to have mean 0 and variance 1.

4. Consider a simplified batch norm layer where we don’t actually divide by standard deviation, instead we just de-mean our data before scaling it by \( \gamma \), then passing it to the next layer. That is, we calculate our mini-batch mean \( \mu \), then simply let \( \hat{x}_i = x_i - \mu \), and \( y_i = \gamma \hat{x}_i \) is passed onto the next layer. Assume batchsize of \( m \). If our final loss function is \( L \), Calculate \( \frac{\partial L}{\partial x_i} \) in terms of \( \frac{\partial L}{\partial y_j} \) for \( j = 1, \ldots, m, \gamma, \) and \( m \).