Homework week 2

Problem 1 Read the paper *Deep Learning: An Introduction for Applied Mathematicians*. Consider a network as defined in (3.1) and (3.2). Assume that $n_L = 1$, find a algorithm to calculate $\nabla a^{[L]}(x)$.

Note that $a^{[1]} = x \in \mathbb{R}^{n_1}$ and

$$a^{[L]} = F(x) = \sigma\left(W^{[L]}a^{[L-1]} + b^{[L]}\right), \quad a^{[\ell]} = \sigma\left(W^{[\ell]}a^{[\ell-1]} + b^{[\ell]}\right), \quad \ell = 2, \dots, L-1, \tag{0.1}$$

where $W^{[\ell]} \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$, $a^{[\ell]} \in \mathbb{R}^{n_\ell}$ and $b^{[\ell]} \in \mathbb{R}^{n_\ell}$ for $\ell = 2, \dots, L$. Applying chain rule, we find

$$\begin{aligned}
\partial_\ell F(x) &= \sigma'\left(W^{[L]}a^{[L-1]} + b^{[L]}\right) W^{[L]} \partial_\ell a^{[L]} \\
&= \sigma'\left(W^{[L]}a^{[L-1]} + b^{[L]}\right) W^{[L]} \sigma'\left(W^{[L-1]}a^{[L-2]} + b^{[L-1]}\right) W^{[L-1]} \partial_\ell a^{[L-1]} \\
&= \dots \\
&= \sigma'\left(W^{[L]}a^{[L-1]} + b^{[L]}\right) W^{[L]} \cdots \sigma'\left(W^{[2]}x + b^{[2]}\right) \partial_\ell \left(W^{[2]}x\right).
\end{aligned} \tag{0.2}$$

Hence, we obtain

$$\begin{aligned}
\nabla F(x) &= \sigma'\left(W^{[L]}a^{[L-1]} + b^{[L]}\right) W^{[L]} \cdots \sigma'\left(W^{[2]}x + b^{[2]}\right) W^{[2]} (1, 1, \dots, 1)^T \\
&= \Pi_{\ell=2}^{L} \left(\sigma'\left(W^{[\ell]}a^{[\ell-1]} + b^{[\ell]}\right) W^{[\ell]}\right) (1, 1, \dots, 1)^T,
\end{aligned} \tag{0.3}$$

where $(1, 1, \dots, 1)^T$ is a vector with size $n_1 \times 1$.

Problem 2 Use a neural network to approximate the Runge function

$$f(x) = \frac{1}{1 + 25x^2}, \qquad x \in [-1, 1]. \tag{0.4}$$

Write a short report explaining method, results, and discussion including

- Plot the true function and the neural network prediction together.
- Show the training/validation loss curves.
- Compute and report errors

We approximate the Runge function

$$f(x) = \frac{1}{1 + 25x^2}, \qquad x \in [-1, 1], \tag{0.5}$$

using a feed-forward neural network with two hidden layers and the activation $\sigma(x) = \tanh x$. We have compared mean squared error (MSE) with the maximum absolute error (max error) and found that MSE

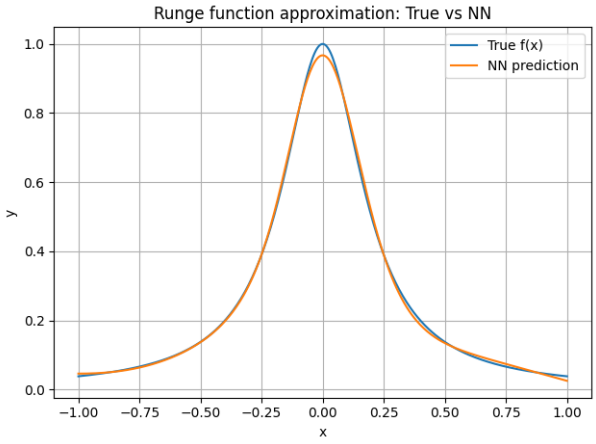

Figure 1: exact function vs. approximate function

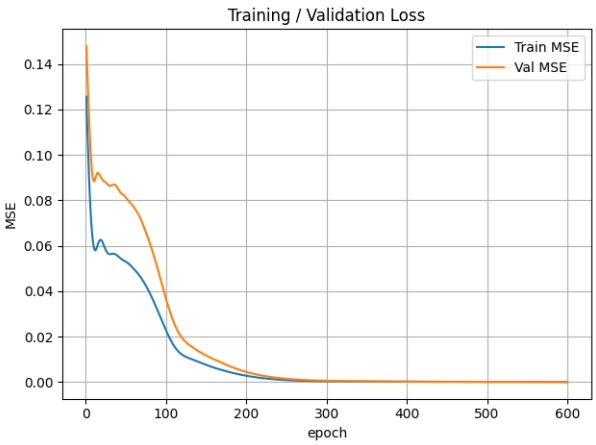

Figure 2: training loss

is the more practical criterion for model selection and early stopping in this task. Furthermore, the error report: Validation MSE $= 9.379 \times 10^{-5}$.

Problem 3

Question How can we find an approximate function that approximates an L^p function using a neural network?