Machine Learning

313704071 陳安定 Assignment 2

1. Consider a network as defined in (3.1) and (3.2). Assume that $n_L = 1$, find an algorithm to calculate $\nabla a^{[L]}(x)$.

(3.1) $a^{[1]} = x \in \mathbb{R}^{n_1}$

(3.2) $a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right) \in \mathbb{R}^{n_l}$ for $l = 2, 3, \dots, L$.

$$\nabla a^{[L]}(x) = \begin{bmatrix} \frac{\partial a^{[l]}}{\partial W^{[l]}} \\ \frac{\partial a^{[l]}}{\partial b^{[4]}} \\ \frac{\partial a^{[l]}}{\partial a^{[l-1]}} \end{bmatrix}$$

$$\frac{\partial a^{[l]}}{\partial W^{[l]}} = \sigma'\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)a^{[l-1]}$$

$$\frac{\partial a^{[l]}}{\partial b^{[4]}} = \sigma'\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$

$$\frac{\partial a^{[l]}}{\partial a^{[l-1]}} = \sigma'\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)W^{[l]}$$

2. Use a neural network to approximate the Runge function.

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

The $f(x)$ is an even function, in $[-1,1]$, $0 < f(x) \le 1$, maximum value $f(0) = 1$, minimum value $f(\pm 1) = \frac{1}{26}$. Moreover, $f(x)$ is decreasing when $x > 0$, and increasing when $x < 0$.

$$f'(x) = -\frac{50x}{(1+25x^2)^2}$$

***Hypothesis**:* An MLP trained with Chebyshev-like sampling and endpoint reweighting in the loss (weighted MSE); hidden layers use **tanh** and the output layer is linear. Expect to achieve $RMSE \le \varepsilon$ and $L_\infty \le \delta$, where ε, δ is the pre-set tolerance limits. ($RMSE$: root mean square error, L_∞: Maximum absolute error).

Evaluation criteria*:* The reasons why we use these two criteria as follows, RMSE focus on the overall average error, it can reflect the approximate quality of the model in most intervals, and L_∞ check the worst-case performance of the model within the interval, especially the areas where the Runge function has large curvature at the endpoints and is most prone to oscillation.

Chebyshev-like sampling*:* In the interval $[-1, 1]$, samples are distributed more densely at the endpoints to reduce the approximation error caused by the Runge function when the endpoint curvature is large.

Model*:* Two hidden layers are sufficient to represent smooth functions. 64–128 neurons are recommended per layer to ensure approximation capability. (we use 64 neurons). About the Weight setting, adopt Xavier/Glorot-uniform and follow tanh activation function to set appropriate **gain** (about 5/3). The initial value of Bias is set to 0 to facilitate symmetric training.

$$W(x) = \frac{1}{\sqrt{1 - x^2 + \varepsilon}}$$

When x reaches ± 1, $1 - x^2$ approach 0, W(x) will increase. Used to strengthen the impact of endpoints

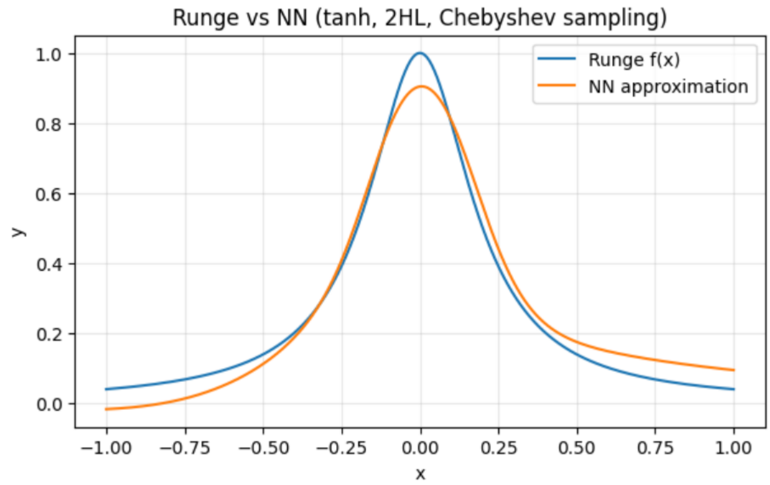
> *Note:* Xavier/Glorot-uniform initialization is a weight initialization method that aims to balance the signal variance in forward propagation and back propagation to avoid gradient explosion or disappearance.

Forward / Backpropagation*:* By inputting x through (1-64-64-1)

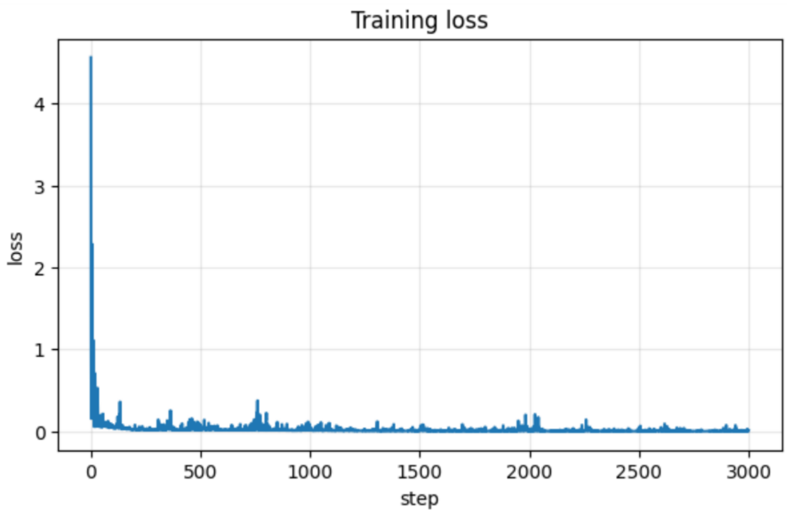
$$W^{[4]}\sigma\left(W^{[3]}\sigma\left(W^{[2]}\boldsymbol{x} + b^{[2]}\right) + b^{[3]}\right) + b^{[4]}$$

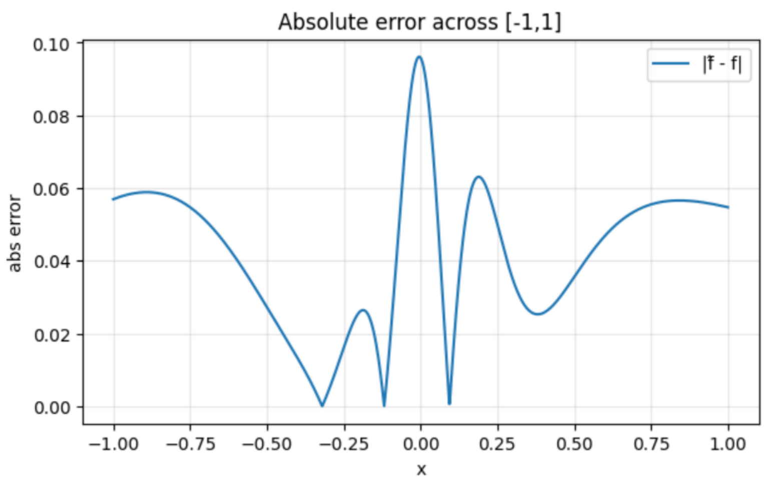
to produce approximate value $f'(x)$, then return error and calculate the gradient according to the loss function and update the weights and biases.

```
step     1 | loss=4.5633e+00 | RMSE=4.6317e-01 | L∞=1.0489e+00
step   500 | loss=8.2160e-02 | RMSE=1.7904e-01 | L∞=4.7691e-01
step  1000 | loss=1.6874e-02 | RMSE=1.4317e-01 | L∞=3.5808e-01
step  1500 | loss=2.3310e-02 | RMSE=9.2911e-02 | L∞=2.6680e-01
step  2000 | loss=2.9639e-03 | RMSE=6.1346e-02 | L∞=2.1472e-01
step  2500 | loss=2.9682e-03 | RMSE=4.2217e-02 | L∞=1.3388e-01
step  3000 | loss=1.1849e-02 | RMSE=4.6706e-02 | L∞=9.6132e-02
```



At the center point $x = 0$, NN's hotspot is slightly lower than the true value. Near the endpoints$|x| \approx 1$, the function value of NN is slightly higher than the true value.





The error for x>0 is larger than that for x<0.