Approximating the Runge Function and Its Derivative with a Neural Network

Ou Chia Yu
Department of Information Management and Finance, NYCU

September 21, 2025

1 Introduction

In the previous assignment, we trained a neural network to approximate the Runge function

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

In this assignment, we extend the task by also approximating its derivative

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

The goal is to evaluate how well the model can fit both the function values and their derivatives simultaneously.

2 Method

We follow the same neural network setup as in the previous assignment, where a feed-forward model was used to approximate the Runge function. The dataset was split into 70% for training, 15% for validation, and 15% for testing.

The loss function consists of two components:

$$\mathcal{L} = \mathcal{L}_{\text{func}} + \lambda \, \mathcal{L}_{\text{deriv}},$$

where

$$\mathcal{L}_{\text{func}} = \frac{1}{N} \sum_{i=1}^{N} \big(\hat{f}(x_i) - f(x_i)\big)^2, \quad \mathcal{L}_{\text{deriv}} = \frac{1}{N} \sum_{i=1}^{N} \big(\widehat{f'}(x_i) - f'(x_i)\big)^2.$$

Here $\hat{f}(x)$ is the neural network prediction, $\widehat{f'}(x)$ is the derivative of the network output obtained via automatic differentiation, and λ is a weighting factor set to 1 in this experiment.

3 Results

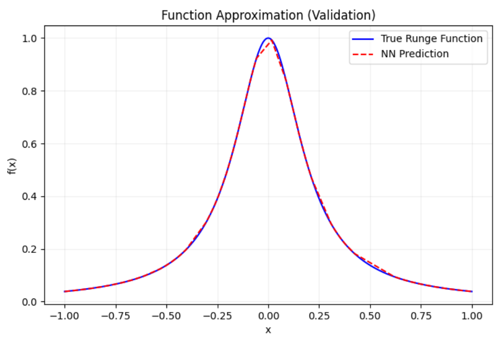
The neural network was trained to approximate both the Runge function $f(x)$ and its derivative $f'(x)$. Unlike the previous assignment, this work employed a 70%/15%/15% split for training, validation, and testing. The introduction of a separate test set provides a more rigorous evaluation of generalization, but also leads to small numerical differences compared with earlier results.

As shown in Figure 1a, the prediction for $f(x)$ almost perfectly overlaps the true curve, and Figure 1b shows that the model also learned $f'(x)$ with high accuracy. The training and validation losses in Figure 2 converge to nearly zero with only minor oscillations caused by the derivative component.

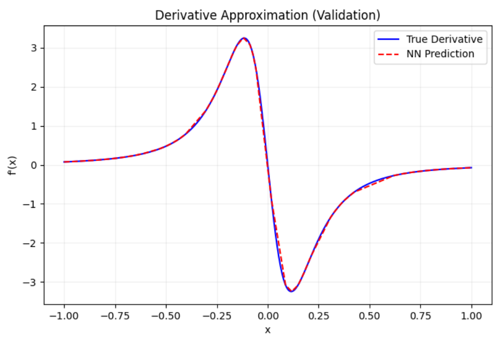
On the held-out test set, the results were:

$$\mathrm{MSE}(f) \approx 4.81 \times 10^{-7}, \quad \text{Max Error}(f) \approx 1.49 \times 10^{-3},$$

$$\mathrm{MSE}(f') \approx 6.12 \times 10^{-6}, \quad \text{Max Error}(f') \approx 1.41 \times 10^{-2}.$$



(a) Function approximation.



(b) Derivative approximation.

Figure 1: Comparison of neural network predictions with the true Runge function and its derivative.

4 Discussion

The results show that the neural network achieved very low errors when approximating the Runge function $f(x)$, with predictions nearly indistinguishable from the true curve. The derivative $f'(x)$ was also learned accurately, though with slightly larger errors, which is expected since derivatives are more sensitive to small variations in the network output.

The training and validation losses both converged to nearly zero (Figure 2), with only minor fluctuations throughout training. Such oscillations are common when incorporating derivative loss, as it amplifies small changes in the parameters. These could be further reduced by adjusting the weighting factor or applying a learning rate schedule, though the current results are already satisfactory.

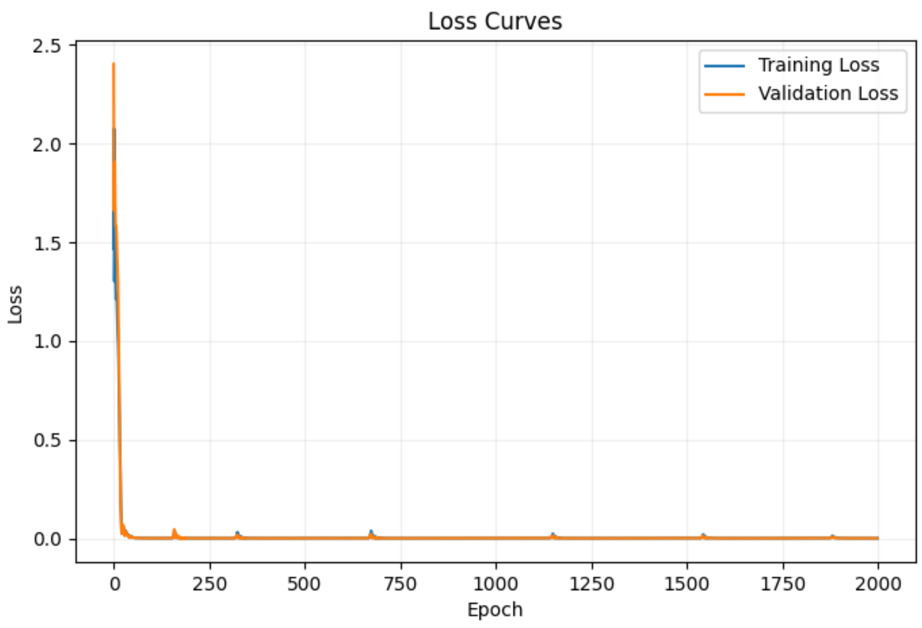


Figure 2: Training and validation loss curves.