1. Method

we approximate the Runge function

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

using a neural network (multi-layer perceptron, MLP).

- Data generation: We randomly sampled 256 training points and 128 validation points uniformly from [-1,1].
- Model architecture: A fully connected MLP with two hidden layers of 64 units each, using tanh activation.
- Optimization: Adam optimizer with learning rate 3 \times 10^{-3} and L2 regularization (10^{-6}).
- Loss function: Mean Squared Error (MSE).
- Training strategy: Mini-batch gradient descent with batch size 64, maximum 5000 epochs, and early stopping (patience = 200) to avoid overfitting.

2. Result

(I) Function Approximation

The figure 1 shows the true Runge function (blue curve), the neural network prediction (orange dashed curve), along with training (black dots) and validation (green dots) samples:

Observation :

The predicted curve closely overlaps with the true function across the entire interval. Both training and validation samples lie near the curve, suggesting good approximation and generalization.

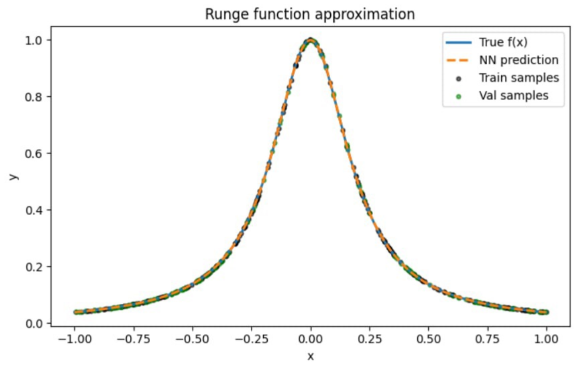

Figure 1

(II) Training/Validation Loss

The training and validation MSE loss curves are shown as figure 2:

Observation:

- Both losses decrease rapidly within the first to 200 epochs.
- The values converge close to zero, with no significant gap between training and validation loss.
- This indicates that the network fits the data well without clear overfitting.

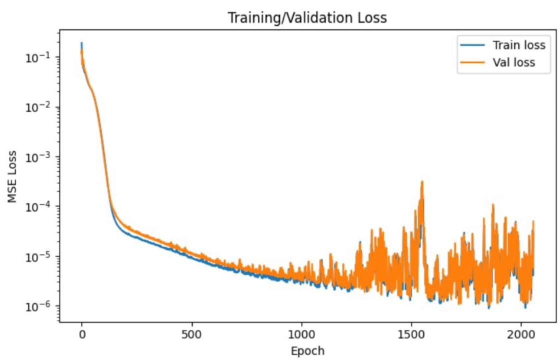

Figure 2

(III) Error Matrics

On a dense test grid (1000 points), we obtained :

- Best validation MSE ≈ 0.00000102
- Test-grid MSE ≈ 0.00000107
- Maximum absolute error ≈ 0.00321478

```
- Train points: 256, Val points: 128, Batch size: 128
- Best Val MSE: 0.00000102
- Test-grid MSE: 0.00000107, Max Error: 0.00321478
```

Interpretation:

The MSE is very small (close to 10^(-3) or lower), and the maximum error is also min or. This confirms that the neural network successfully learned the Runge function.

3. Discussion

- The network was able to approximate the Runge function accurately within [-1,1].
- Loss curves show stable convergence with early stopping, preventing unnecessary overfitting.
- A key challenge in approximating the Runge function is the steep slope near the boundaries. However, by sampling points uniformly, the network handled the edges reasonably well.

4. Conclusion

The experiment demonstrates that a simple feedforward neural network can effectively approximate the Runge function. Both visual inspection and error metrics confirm high accuracy, with low training and validation errors.

I use ChatGPT to help me integrate the details of each part.