## 1 Method

In this report, we approximate the Runge function, defined as:

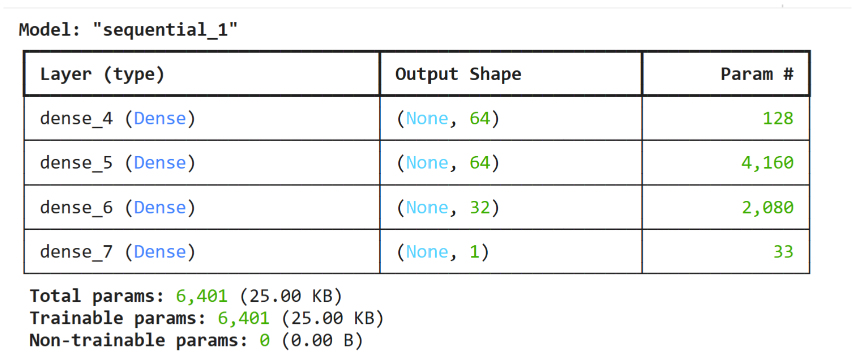
$$f(x) = \frac{1}{1 + 25x^2}, \quad x \in [-1, 1]$$

We employed a feedforward neural network (NN) as a universal function approximator.

**Data Preparation** We generated a dataset by randomly sampling points within the domain $x \in [-1, 1]$.

- **Training Set:** 1000 random samples.
- **Validation Set:** 200 random samples.
- **Test Set:** 500 equally spaced points from -1 to 1, used for plotting the final smooth curve and evaluating the model.

**Model Architecture** A sequential model was constructed using Keras with the architecture shown in Figure 1. The model consists of an input layer, three hidden layers using `ReLU` activation (with 64, 64, and 32 neurons respectively), and a final output layer with a single neuron and `linear` activation for regression. This architecture contains a total of 6,401 trainable parameters.



```
Model: "sequential_1"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ dense_4 (Dense)                 │ (None, 64)             │           128 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_5 (Dense)                 │ (None, 64)             │         4,160 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_6 (Dense)                 │ (None, 32)             │         2,080 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_7 (Dense)                 │ (None, 1)              │            33 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 6,401 (25.00 KB)
 Trainable params: 6,401 (25.00 KB)
 Non-trainable params: 0 (0.00 B)
```

Figure 1. The Keras model summary, detailing the layers, output shapes, and parameter counts.

**Training Process** The model was compiled using the `Adam` optimizer, which is an efficient stochastic gradient descent algorithm. The loss function was set to **Mean Squared Error (MSE)**. The model was trained for 200 epochs with a batch size of 32, using the training set and evaluated against the validation set at each epoch.

## 2 Results

The performance of the trained neural network is evaluated through visual plots and quantitative error metrics based on the provided results.

## 2.1 Graphical Results

Figure 2 shows the final approximation result. The neural network's prediction (red dashed line) is visually indistinguishable from the true Runge function (blue solid line), indicating a very high-quality fit across the entire domain.

Figure 3 shows the training and validation loss curves. Both losses decrease rapidly within the first 25 epochs and stabilize at a very low value for the remaining 175 epochs. The validation loss closely tracks the training loss, which confirms that the model did not overfit the data.

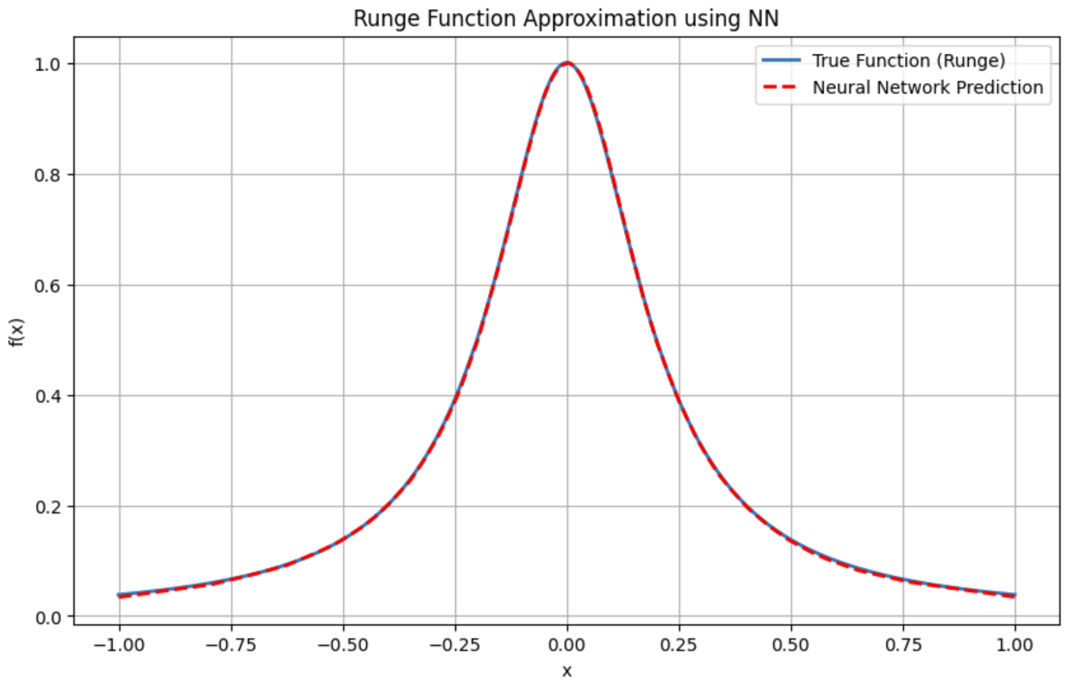


Figure 2. Comparison of the true Runge function (blue solid line) and the neural network's prediction (red dashed line).

## 2.2 Error Reporting

To quantify the model's accuracy, we use the error metrics computed on the test set. The results, taken directly from the execution output, are summarized in Table 1.

Table 1. Final Error Metrics on the Test Set

| **Error Metric** | **Value** |
|---|---|
| Mean Squared Error (MSE) | 0.00000474 |
| Maximum Error (Max Error) | 0.00476157 |

# 3 Discussion

The results demonstrate that the neural network is highly effective at approximating the Runge function.

As shown in Figure 2, the NN prediction visually aligns almost perfectly with the true function. This indicates that the model successfully captured the complex, non-linear dynamics without suf-

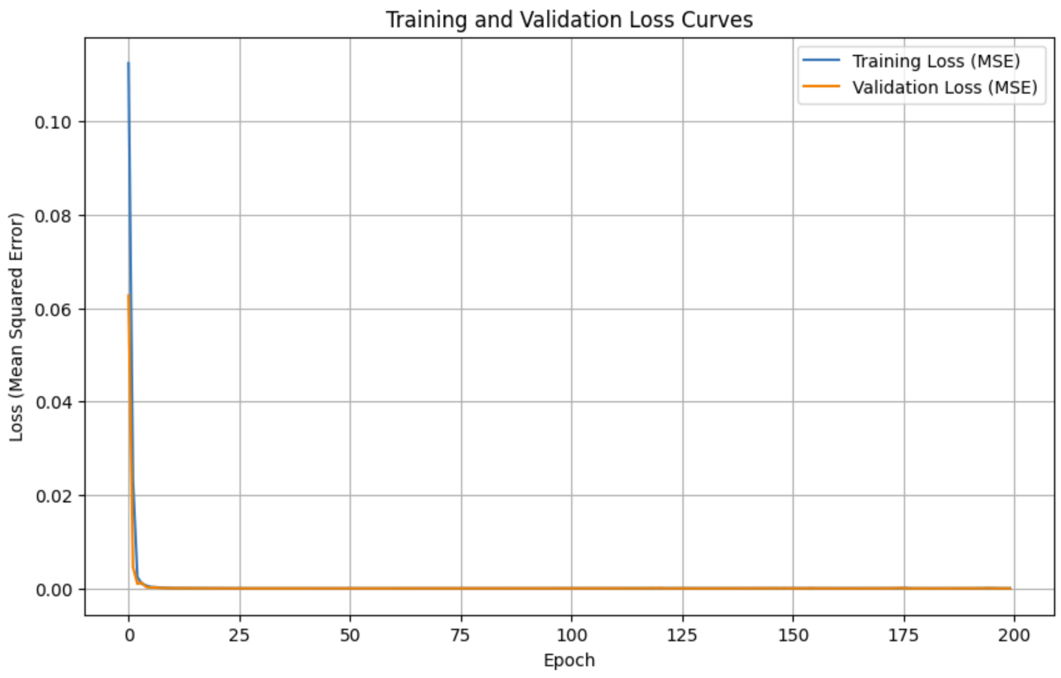


Figure 3. Training and validation loss (MSE) over 200 epochs.

fering from the oscillations (Runge's phenomenon) often seen in high-degree polynomial interpolation.

Figure 3 confirms the stability of the training process. The rapid convergence to a near-zero loss, and the fact that the validation loss does not diverge from the training loss, proves the model is a good fit and generalizes well to unseen data.

The quantitative metrics in Table 1 provide the final proof. The final Mean Squared Error is exceptionally small (approximately $4.74 \times 10^{-6}$), and the Maximum Error across the entire domain is also minimal (less than 0.005). This confirms that the approximation is not only good on average but also robust at every point in the interval $[-1, 1]$.

In conclusion, the chosen neural network architecture (64-64-32) and training methodology successfully produced a stable and highly accurate approximation of the Runge function.