HW2.2 函数拟合

1. 函数定义

本实验的目标是使用一个三层神经网络（两层 ReLU 网络）拟合如下的目标函数：

$$f(x) = x + 5\sin(5x)$$

此函数包含线性项和强非线性项，具有周期性波动特征，适合测试神经网络的拟合能力。

2. 数据采集

数据集的构造过程如下：

⑴ 训练集 x_train 由[-2,2]区间内均匀随机采样 1000 个数据点；

⑵ 目标输出 y_train 由目标函数计算，并加入均值为 0、标准差为 0.3 的高斯噪声，以模拟真实数据中的测量误差；

```
x_train = np.random.uniform(-2, 2, (1000, 1))
y_train = target_function(x_train) + np.random.normal(0, 0.3, (1000, 1))  # 降低噪声
```

⑶ 测试集 x_test 由 200 个等间距点组成，评估模型泛化能力；

⑷ 测试标签 y_test 由目标函数计算，无噪声。

```
x_test = np.linspace(-2, 2, 200).reshape(-1, 1)
y_test = target_function(x_test)
```

3. 模型描述

3.1 本模型是一个三层前馈神经网络，包含：

⑴ 输入层：1 个神经元，输入变量 x；

⑵ 隐藏层 1：128 个神经元，使用 ReLU 激活函数：

$$h_1 = \max(0, xW_1 + b_1)$$

⑶ 隐藏层 2：128 个神经元，使用 ReLU 激活函数：

$$h_2 = \max(0, h_1W_2 + b_2)$$

⑷ 输出层：1 个神经元，计算最终输出：

$$\hat{y} = h_2W_3 + b_3$$

```
# 模型参数初始化
input_size = 1 #输入层
hidden_size = 128 #隐藏层
output_size = 1 #输出层

# 初始化权重和偏置
W1 = np.random.randn(input_size, hidden_size) * np.sqrt(2 / input_size)
b1 = np.zeros((1, hidden_size))
W2 = np.random.randn(hidden_size, hidden_size) * np.sqrt(2 / hidden_size)
b2 = np.zeros((1, hidden_size))
W3 = np.random.randn(hidden_size, output_size) * np.sqrt(2 / hidden_size)
b3 = np.zeros((1, output_size))
```

3.2 损失函数

本模型采用均方误差 MSE 作为损失函数：

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

```
loss = np.mean((y_pred - y_train) ** 2) #MSE
```

3.3 训练与优化方法

⑴ 采用反向传播法计算梯度；

```
# 输出层梯度
dW3 = np.dot(hidden2.T, dL_dy_pred)
db3 = np.sum(dL_dy_pred, axis=0, keepdims=True)
# 反向传播到隐藏层 2
dL_dhidden2 = np.dot(dL_dy_pred, W3.T)
dhidden2_pre = dL_dhidden2 * (hidden2_pre > 0)
# 隐藏层 2 梯度
dW2 = np.dot(hidden1.T, dhidden2_pre)
db2 = np.sum(dhidden2_pre, axis=0, keepdims=True)
# 反向传播到隐藏层 1
dL_dhidden1 = np.dot(dhidden2_pre, W2.T)
dhidden1_pre = dL_dhidden1 * (hidden1_pre > 0)
# 隐藏层 1 梯度
dW1 = np.dot(x_train.T, dhidden1_pre)
db1 = np.sum(dhidden1_pre, axis=0, keepdims=True)
```

⑵ 采用梯度下降法进行参数更新；

```
W1 -= learning_rate * dW1
b1 -= learning_rate * db1
W2 -= learning_rate * dW2
b2 -= learning_rate * db2
W3 -= learning_rate * dW3
b3 -= learning_rate * db3
```

其中学习率设置为 0.01，训练 7000 轮次，每 500 轮输出一次损失。

```
learning_rate = 0.01 #学习率
num_epochs = 7000 # 7000轮次
```

4. 拟合效果

⑴ 训练过程中，损失函数逐步下降，经过 7000 轮次的训练，最终损失为 0.1094，数值较小，表明模型成功学习到了目标函数的映射关系；

```
Epoch [500/7000], Loss: 11.6169
Epoch [1000/7000], Loss: 7.7328
Epoch [1500/7000], Loss: 4.0532
Epoch [2000/7000], Loss: 3.4452
Epoch [2500/7000], Loss: 2.6921
Epoch [3000/7000], Loss: 1.7588
Epoch [3500/7000], Loss: 1.0258
Epoch [4000/7000], Loss: 0.6094
Epoch [4500/7000], Loss: 0.3821
Epoch [5000/7000], Loss: 0.2741
Epoch [5500/7000], Loss: 0.2060
Epoch [6000/7000], Loss: 0.1351
Epoch [6500/7000], Loss: 0.1127
Epoch [7000/7000], Loss: 0.1094
```

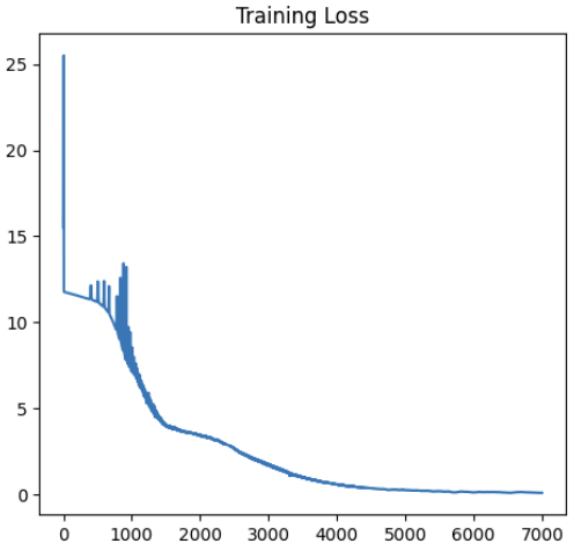
⑵ 在测试集上，模型的 MSE 为 0.0451，进一步验证了其泛化能力；

```
Test MSE: 0.0451
```

⑶ 可视化结果

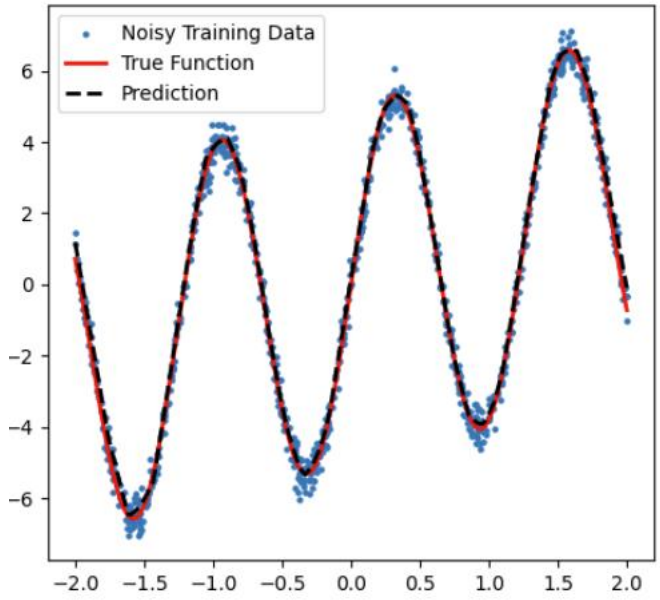
① 训练损失曲线

从损失下降趋势可看出，模型收敛良好，训练轮次至 5000 次左右后损失趋于稳定；



② 函数拟合曲线

在[-2,2]上，神经网络拟合出的曲线（黑色虚线）与目标函数（红色曲线）较为吻合；即使训练数据（蓝色点）带有一定噪声，模型仍能较好地拟合整体趋势，体现了其良好的抗噪能力与泛化能力。


Noisy Training Data
True Function
Prediction
6
4
2
0
−2
−4
−6
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0