# 114-1 Machine Learning

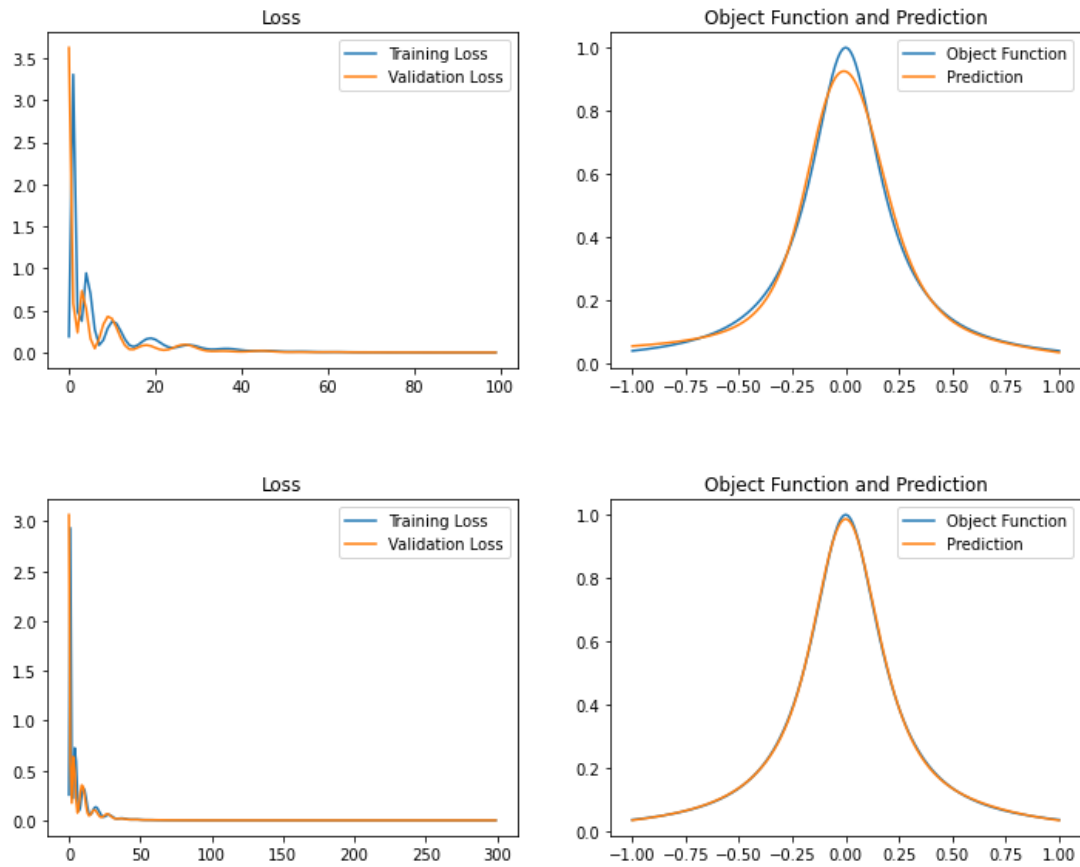# Week 2 Assignment

Ming Hsun Wu

September 15, 2025

PROBLEM 1.

Use a neural network to approximate the Runge function

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

SOLUTION.

NOTE OF PROBLEM 1.

The programming language I used is Python, and I mainly relied on the PyTorch library to carry out the simulations. Based on the trained model, the mean squared error (MSE) of the prediction results, evaluated on 500 points within the interval$[1, 1]$, is 0.000520.

Figure1 shows the results of the preliminary test. After tuning more parameters (such as the number of epochs), better results can be obtained, as shown in Figure2, with the MSE reduced to 0.000014.

```python
1  # Algorithm Code
2  # (Define object function f, grand_f and some parameters)
3  'create network'
4  class Curve_Fitting(torch.nn.Module):
5      def __init__(self):
6          super(Curve_Fitting, self).__init__()
7          self.Linear_1 = torch.nn.Linear(1, 128)
8          self.Linear_2 = torch.nn.Linear(128, 64)
9          self.Linear_3 = torch.nn.Linear(64, 1)
10         self.Tanh = torch.nn.Tanh()
11
12     def forward(self, x):
13         x = self.Linear_1(x)
14         x = self.Linear_2(x)
15         x = self.Tanh(x)
16         x = self.Linear_3(x)
17
18         return x
19
20 # compile model
21 model = Curve_Fitting()
22 MSE = torch.nn.MSELoss()
23 optimizer = optim.Adam(model.parameters(), lr=0.01)
24 'train model and predict'
25 epochs = 100
26 train_losses = []
```

```python
27  val_losses = []
28  for epoch in range(epochs):
29      # train
30      model.train()
31      optimizer.zero_grad()
32      outputs = model(x_train)
33      loss = MSE(outputs, y_train)
34      loss.backward()
35      optimizer.step()
36      train_losses.append(loss.item())
37      # val
38      model.eval()
39      with torch.no_grad():
40          val_outputs = model(x_val)
41          val_loss = MSE(val_outputs, y_val).item()
42          val_losses.append(val_loss)
43
44
45  # estimation
46  x_test = torch.tensor(np.linspace(-1, 1, 500).reshape(-1,
          1).astype(np.float32))
47  with torch.no_grad():
48      y_pred = model(x_test).numpy()
49
50  y_true = object_function(x_test.numpy())
```