Homework week 3

Problem 1 Reading and Explaining Lemmas 3.1 and Lemma 3.2 in the paper *Ryck et al., On the approximation of functions by tanh neural networks.*

This paper's main result establishes a constructive approximation for functions in $W^{k,\infty}$. By the Bramble–Hilbert Theorem (Lemma A.8), any $W^{k,\infty}$-function can be approximated by a polynomial of degree at most $k-1$. Hence, the key step is to construct accurate approximations to the monomials x^p for any $p \geq 1$, and to express these approximations as neural networks with activation.

Taking $\sigma = \tanh x$, the proof treats odd and even monomials separately: Lemma 3.1 handles x^{2n+1}, and Lemma 3.2 handles x^{2n} for any $n \geq 1$. In Lemma 3.1, estimate (20) eliminates the dependence on the parameter m and yields an upper bound in terms of h, which can be controlled. This immediately leads to the result (17). Lemma 3.2 proceeds analogously. Because σ is odd whereas the objective function x^{2n} is even, we should modify the previous approximate function by symmetrizing it specifically, so that the odd part cancels and the approximate function becomes even(more detail can see (27)).

Programming assignment

(1) I have reused the previous code with the following configuration. (1) The dataset consists of uniformly sampled points over the domain and is split into training, validation, and test sets; training uses only the training set, while the validation set is used to check whether the model has learned sufficiently and to monitor overfitting. (2) The hypothesis class comprises continuous functions implemented as a feed-forward neural network with the activation function $\sigma(x) = \tanh x$; the architecture is 1-32-32-1. (3) The loss function is mean squared error (MSE). After these set up, we find the hypothesis function with training point and validation point are very closed (see figure 1). Taking the test points to this model, we find the error has *9.290005e-07*(see figure 2). However, if we consider the derivative of hypothesis function, we only get the error *4.981137e-04*.

(2) Under the same code, we only modify the MSE loss as follows:

$$\text{loss function} = \sum_{i=1}^{100} \left(y^i - h_\theta(x^i)\right) + ((y^i)' - h'_\theta(x^i)). \tag{0.1}$$

Then the error can be upgraded as *2.902625e-06*(see figure 3, 4), which is better than we only consider original MSE loss.

Problem 2

Question Note that in the paper *Ryck et al., On the approximation of functions by tanh neural networks*, the authors work with function in $W^{k,\infty}$. In theoretical textbook, we all know that using

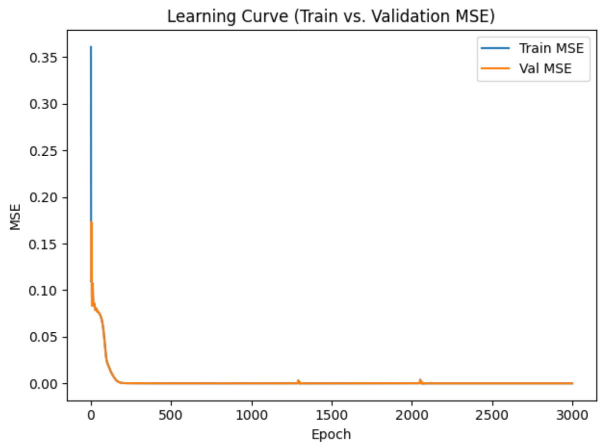

Figure 1:

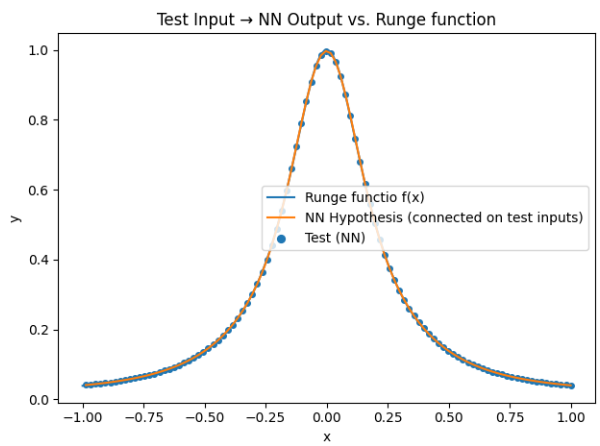

Figure 2:

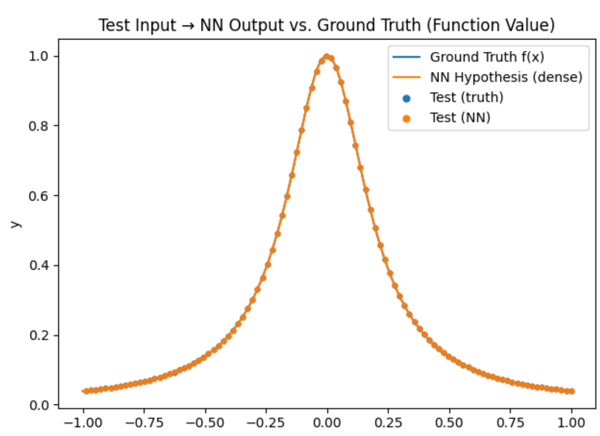

Figure 3:

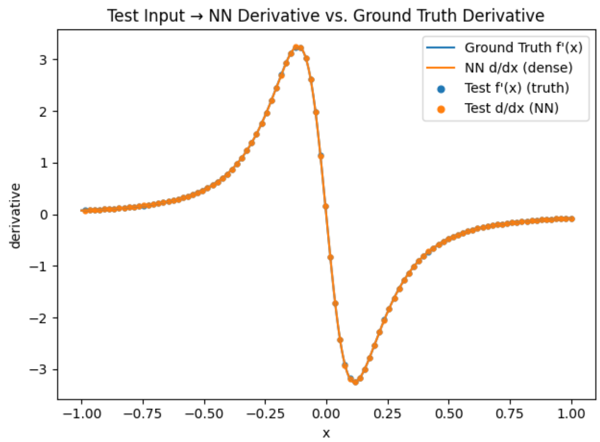

Figure 4: Caption

Morrey's estimate, we obtain the embedding $W^{k,\infty}(\Omega) \subset C^{k-1,\gamma}(\Omega)$ for $\Omega \subset \mathbb{R}^n$ and $\gamma \in (0,1)$. That is, every $W^{k,\infty}$ function is C^{k-1} function. Then, by Stone-Weierstrass Theorem, we always can find a neural network approximate function. However, I would like to ask if $W^{k,p}(\Omega)$ with $1 \leq p < n$ (i.e., we don't have Morrey's estimate), how can we construct a neural network approximate function to a $W^{k,p}(\Omega)$ function?