Laboratorium 10 - Równania różniczkowe - spectral bias

Mateusz Podmokły - II rok Informatyka WI

28 maj 2024

1 Treść zadania

Zadanie 1. Dane jest równanie różniczkowe zwyczajne

$$\frac{du(x)}{dx} = cos(\omega x), \quad x \in \Omega$$

gdzie:
x, ω, $u \in \mathbb{R}$,
x to położenie,
Ω to dziedzina, na której rozwiązujemy równanie, $\Omega = \{x| - 2\pi \leqslant x \leqslant 2\pi\}$,
$u(x)$ to funkcja, której postaci szukamy.

Warunek początkowy zdefiniowany jest następująco:

$$u(0) = 0.$$

Analityczna postać rozwiązania równania z warunkiem początkowym jest następująca:

$$u(x) = \frac{sin(\omega x)}{\omega}$$

Rozwiąż powyższe zagadnienie początkowe. Do rozwiązania użyj sieci neuronowych typu PINN (ang. Physics-informed Neural Network).
Koszt rezydualny zdefiniowany jest następująco:

$$\mathcal{L}_r(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left|\left|\frac{d\hat{u}(x)}{dx} - cos(\omega x_i)\right|\right|^2,$$

gdzie N jest liczbą punktów kolokacyjnych.
Koszt związany z warunkiem początkowym przyjmuje postać:

$$\mathcal{L}_{IC}(\theta) = ||\hat{u}(0) - 0||^2.$$

Funkcja kosztu zdefiniowana jest następująco:

$$\mathcal{L}(\theta) = \mathcal{L}_r(\theta) + \mathcal{L}_{IC}(\theta).$$

Warstwa wejściowa sieci posiada 1 neuron, reprezentujący zmienną x. Warstwa wyjściowa także posiada 1 neuron, reprezentujący zmienną $\hat{u}(x)$. Uczenie trwa przez 50 000 kroków algorytmem Adam ze stałą uczenia równą 0.001. Jako funkcję aktywacji przyjmij tangens hiperboliczny, $tanh(x)$.

Rozważ następujące przypadki:

1. Przypadek $\omega = 1$.
   Ustal następujące wartości:
   - 2 warstwy ukryte, 16 neuronów w każdej warstwie
   - liczba punktów treningowych: 200
   - liczba punktów testowych: 1000
2. Przypadek $\omega = 15$.
   Ustal następujące wartości:
   - liczba punktów treningowych: $200 \cdot 15 = 3000$
   - liczba punktów testowych: 5000

   Eksperymenty przeprowadź z trzema architekturami sieci:
   - 2 warstwy ukryte, 16 neuronów w każdej warstwie
   - 4 warstwy ukryte, 64 neurony w każdej warstwie
   - 5 warstw ukrytych, 128 neuronów w każdej warstwie
3. Dla wybranej przez siebie sieci porównaj wynik z rozwiązaniem, w którym przyjęto, że szukane rozwiązanie (*ansatz*) ma postać:

   $$\hat{u}(x;\theta) = tanh(\omega x) \cdot NN(x;\theta).$$

   Taka postać rozwiązania gwarantuje spełnienie warunku $\hat{u} = 0$ bez wprowadzania składnika $\mathcal{L}_{IC}$ do funkcji kosztu.
4. Porównaj pierwotny wynik z rozwiązaniem, w którym pierwszą warstwę ukrytą zainicjalizowano cechami Fouriera:

   $$\gamma(x) = [sin(2^0\pi x), cos(2^0\pi x), \ldots, sin(2^{L-1}\pi x), cos(2^{L-1}\pi x)]$$

   Dobierz L tak, aby nie zmieniać szerokości warstwy ukrytej.

Dla każdego z powyższych przypadków stwórz następujące wykresy:

– Wykres funkcji $u(x)$, tj. dokładnego rozwiązania oraz wykres funkcji $\hat{u}(x)$, tj. rozwiązania znalezionego przez sieć neuronową

– Wykres funkcji błędu.

Stwórz także wykres funkcji kosztu w zależności od liczby epok.

2 Specyfikacja użytego środowiska

Specyfikacja:

- Środowisko: Visual Studio Code,
- Język programowania: Python,
- System operacyjny: Microsoft Windows 11,
- Architektura systemu: x64.

3 Rozwiązanie problemu

3.1 Biblioteki

W realizacji rozwiązania wykorzystane zostały następujące biblioteki:

```
import torch
import torch.nn as nn
import numpy as np
import matplotlib.pyplot as plt
```

3.2 Model sieci neuronowej

Do zaimplementowania modelu sztucznej sieci neuronowej PINN wykorzystałem bibliotekę `PyTorch`. Dane treningowe to równomiernie rozłożone punkty na przedziale $[-2\pi, 2\pi]$ zawierające krańce przedziału oraz warunek początkowy, czyli $x = 0$. Natomiast jako dane testujące również wykorzystałem równomiernie rozłożone punkty na przedziale $[-2\pi, 2\pi]$, jednak bez krańców przedziału, żeby uniknąć powtórzeń punktów.

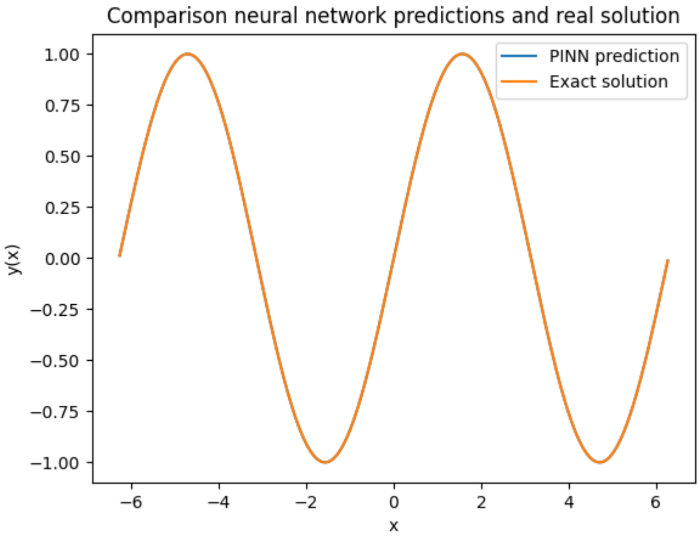
4 Przedstawienie wyników

4.1 Przypadek 1.

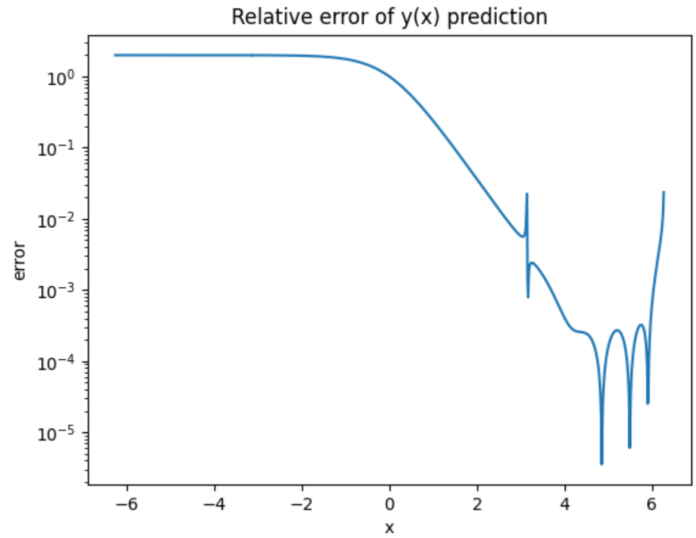
Parametry:

- $\omega = 1$
- 2 warstwy ukryte, 16 neuronów w każdej warstwie

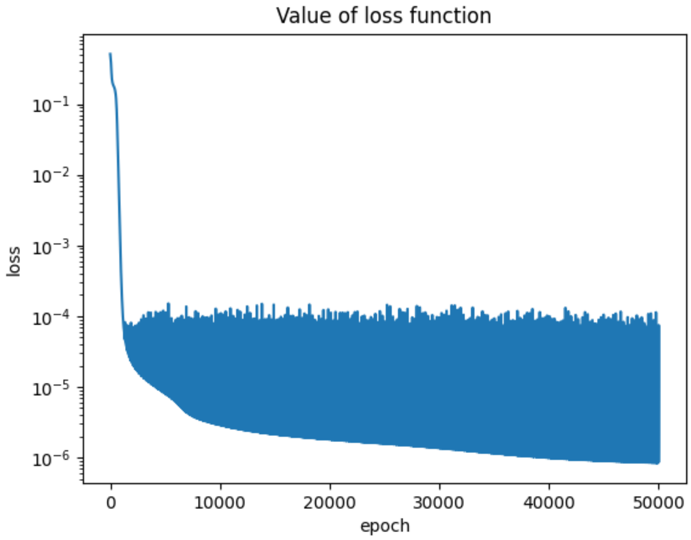
- 200 punktów treningowych
- 1000 punktów testowych



Rysunek 1: Dokładne rozwiązanie i rozwiązanie znalezione przez sieć, przypadek 1.



Rysunek 2: Błąd względny rozwiązania, przypadek 1.

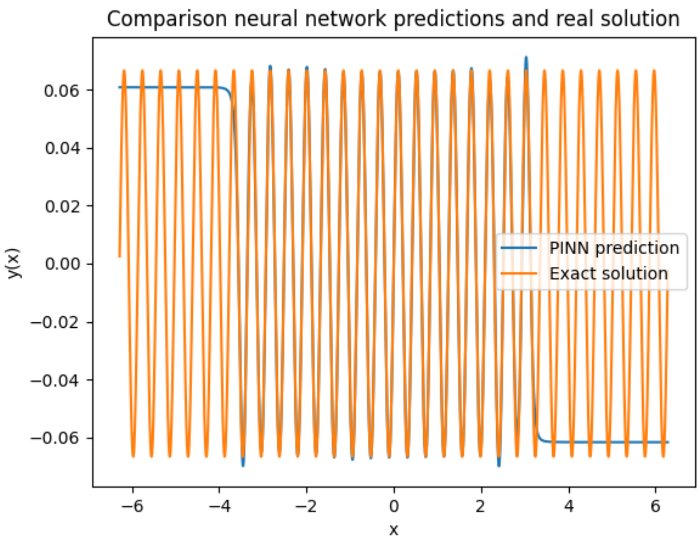


Rysunek 3: Wartość funkcji kosztu w zależności od liczby epok, przypadek 1.

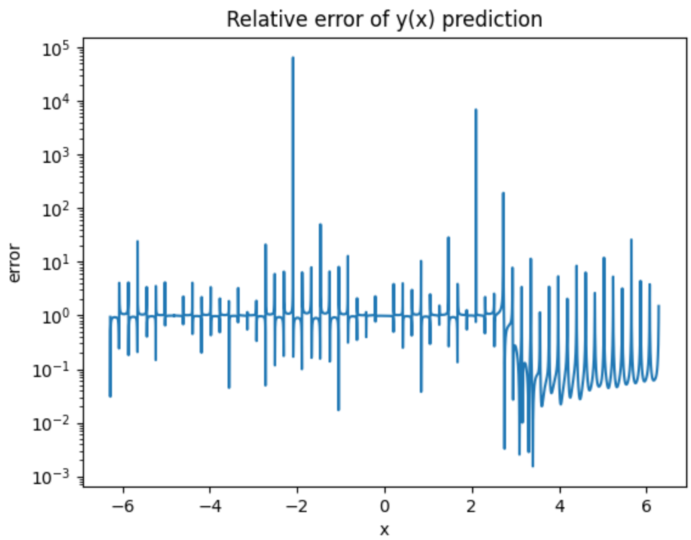
4.2 Przypadek 2.

Parametry:

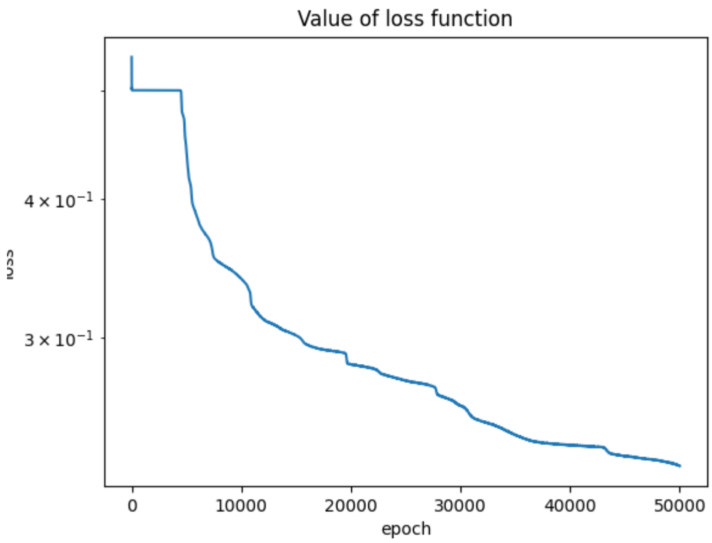
- $\omega = 15$
- 2 warstwy ukryte, 16 neuronów w każdej warstwie
- 3000 punktów treningowych
- 5000 punktów testowych



Rysunek 4: Dokładne rozwiązanie i rozwiązanie znalezione przez sieć, przypadek 2.



Rysunek 5: Błąd względny rozwiązania, przypadek 2.

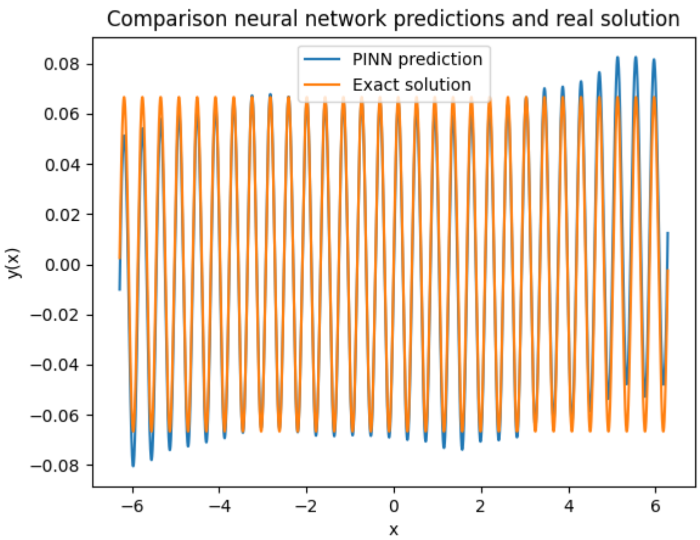


Rysunek 6: Wartość funkcji kosztu w zależności od liczby epok, przypadek 2.

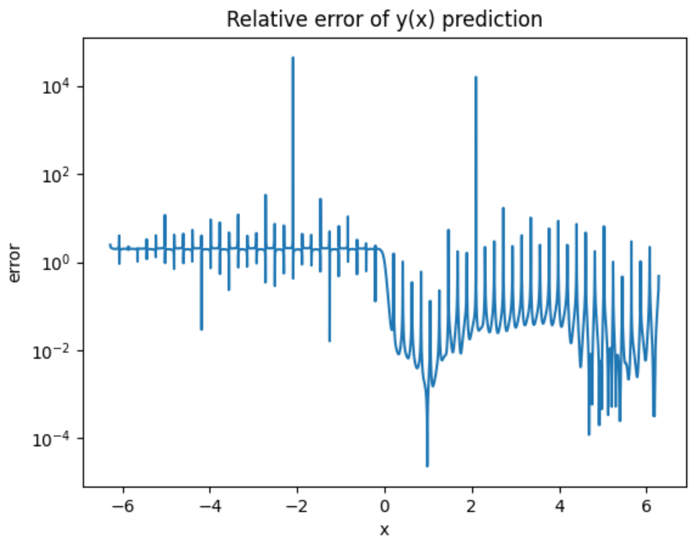
4.3 Przypadek 3.

Parametry:

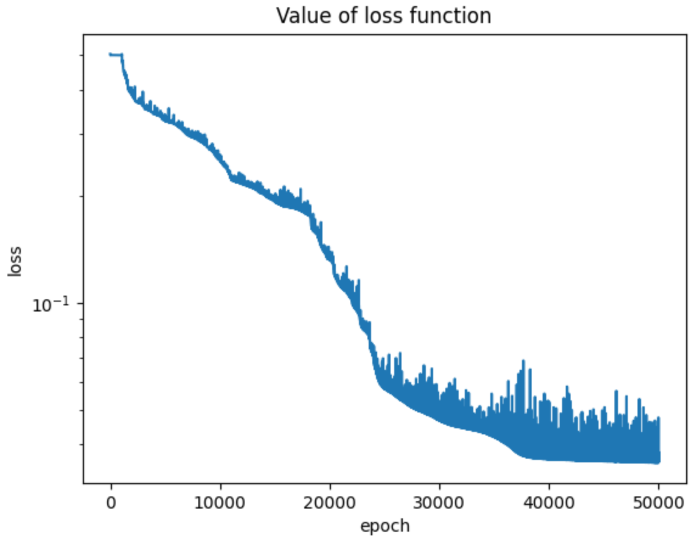
- $\omega = 15$
- 4 warstwy ukryte, 64 neurony w każdej warstwie
- 3000 punktów treningowych
- 5000 punktów testowych



Rysunek 7: Dokładne rozwiązanie i rozwiązanie znalezione przez sieć, przypadek 3.



Rysunek 8: Błąd względny rozwiązania, przypadek 3.

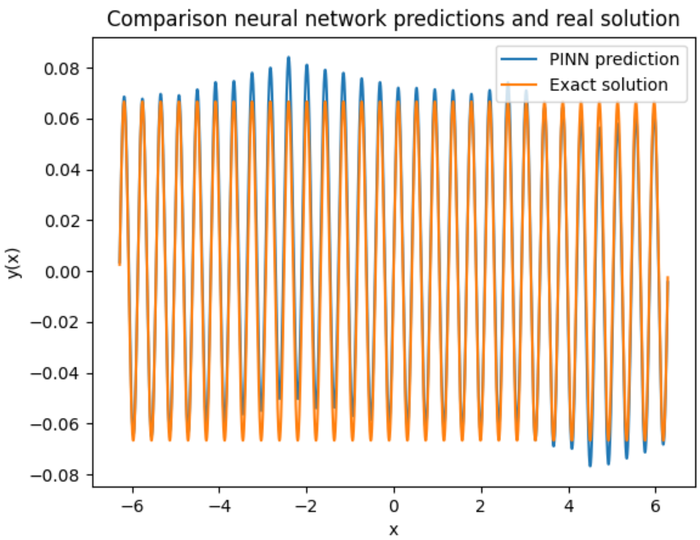


Rysunek 9: Wartość funkcji kosztu w zależności od liczby epok, przypadek 3.

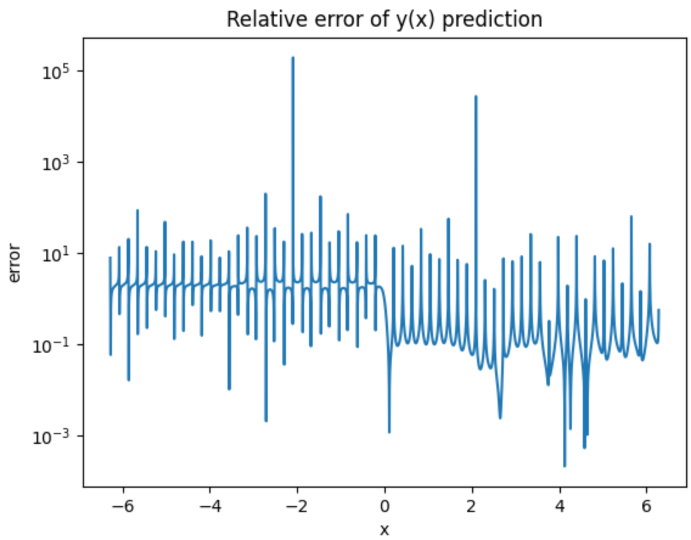
4.4 Przypadek 4.

Parametry:

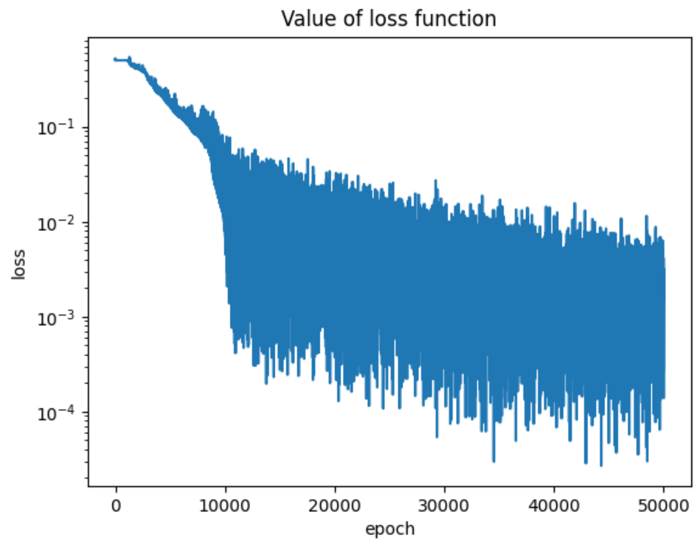
- $\omega = 15$
- 5 warstwy ukryte, 128 neuronów w każdej warstwie
- 3000 punktów treningowych
- 5000 punktów testowych



Rysunek 10: Dokładne rozwiązanie i rozwiązanie znalezione przez sieć, przypadek 4.



Rysunek 11: Błąd względny rozwiązania, przypadek 4.



Rysunek 12: Wartość funkcji kosztu w zależności od liczby epok, przypadek 4.

4.5 Przypadek 5.

Parametry:

- $\omega = 15$
- 4 warstwy ukryte, 64 neuronów w każdej warstwie
- 3000 punktów treningowych
- 5000 punktów testowych
- szukane rozwiązanie (*ansatz*) ma postać:

$$\hat{u}(x;\theta) = tanh(\omega x) \cdot NN(x;\theta)$$