Министерство образования Республики Беларусь
Учреждение образования

«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №5
По дисциплине: «ОМО»
Тема: «Нелинейные ИНС в задачах регрессии»

Выполнил:
Студент 3-го курса
Группы АС-65
Гуща И.В.
Вариант 3
Проверил:
Крощенко А.А.

Брест 2025

Цель работы: Выполнить моделирование прогнозирующей нелинейной ИНС.

Ход работы

Вариант 3

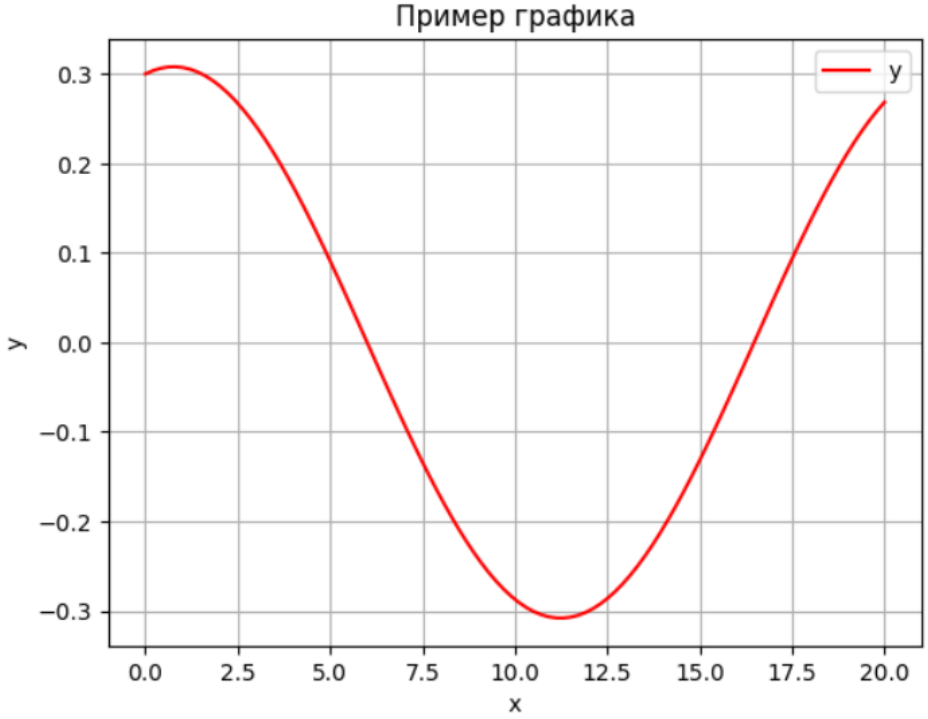
Для генерации обучающих и тестовых данных использовать функцию

$y = a\cos(bx) + c\sin(dx)$.

№ варианта	a	b	c	d	Кол-во входов ИНС	Кол-во НЭ в скрытом слое
3	0.3	0.3	0.07	0.3	10	4

Для прогнозирования использовать многослойную ИНС с одним скрытым слоем. В качестве функций активации для скрытого слоя использовать сигмоидную функцию, для выходного - линейную.

```
import numpy as np
import matplotlib.pyplot as plt
def y(x):
  return 0.3*np.cos(0.3*x)+0.07*np.sin(0.3*x)
x=np.linspace(0,20,100)
y=y(x)
plt.plot(x, y, label='y', color='red')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Пример графика')
plt.legend()
plt.grid(True)
plt.show()
```



```
#print(x)
#print(y)
input=10
hidden=4
train_size=80
def make_dataset(y, window):
    X, Y = [], []
    for i in range(len(y) - window):
        X.append(y[i:i+window])
        Y.append(y[i+window])
    return np.array(X), np.array(Y)

X, Y = make_dataset(y, input)
X_train, Y_train = X[:train_size], Y[:train_size]
X_test,  Y_test  = X[train_size:], Y[train_size:]
Y_train = Y_train.reshape(-1,1)
W1 = np.random.uniform(-0.5, 0.5, (input, hidden))
b1 = np.zeros((1, hidden))

W2 = np.random.uniform(-0.5, 0.5, (hidden, 1))
b2 = np.zeros((1, 1))



def sigmoid(x):
    return 1 / (1 + np.exp(-x))

def forward(X):
    z1 = X @ W1 + b1
    a1 = sigmoid(z1)
    z2 = a1 @ W2 + b2
    return z1, a1, z2

def train(X, Y):
    lr=0.01
    global W1, b1, W2, b2
    z1, a1, y_pred = forward(X)

    delta_output = y_pred - Y  # (batch_size, 1)

    W2 -= lr * a1.T @ delta_output      # (hidden, batch) @ (batch, 1) →
(hidden,1)
    b2 -= lr * np.sum(delta_output, axis=0, keepdims=True)

    delta_hidden = (delta_output @ W2.T) * (a1 * (1 - a1))  # (batch,1) @
(1,hidden) → (batch,hidden)
    W1 -= lr * X.T @ delta_hidden                            # (input,batch)
@ (batch,hidden) → (input,hidden)
    b1 -= lr * np.sum(delta_hidden, axis=0, keepdims=True)

    loss = np.mean((y_pred - Y)**2)
    return loss

epochs = 2000
```

```
loss_history = []

for epoch in range(epochs):
    loss = train(X_train, Y_train)
    loss_history.append(loss)

    if epoch % 200 == 0:
        print(f"Epoch {epoch}: loss = {loss:.6f}")
```

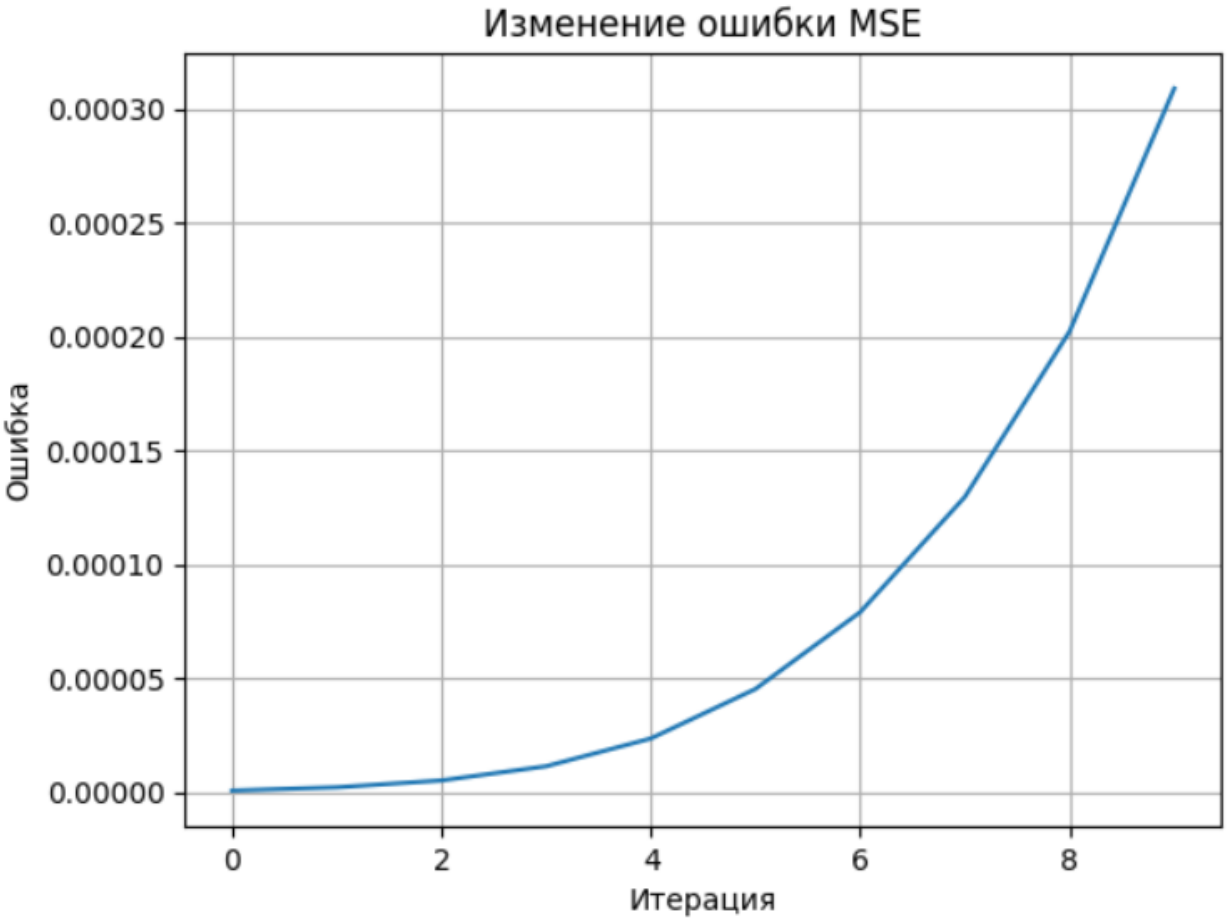
```
Epoch 0: loss = 0.361780
Epoch 200: loss = 0.003441
Epoch 400: loss = 0.002398
Epoch 600: loss = 0.001513
Epoch 800: loss = 0.000836
Epoch 1000: loss = 0.000399
Epoch 1200: loss = 0.000168
Epoch 1400: loss = 0.000064
Epoch 1600: loss = 0.000023
Epoch 1800: loss = 0.000009
```

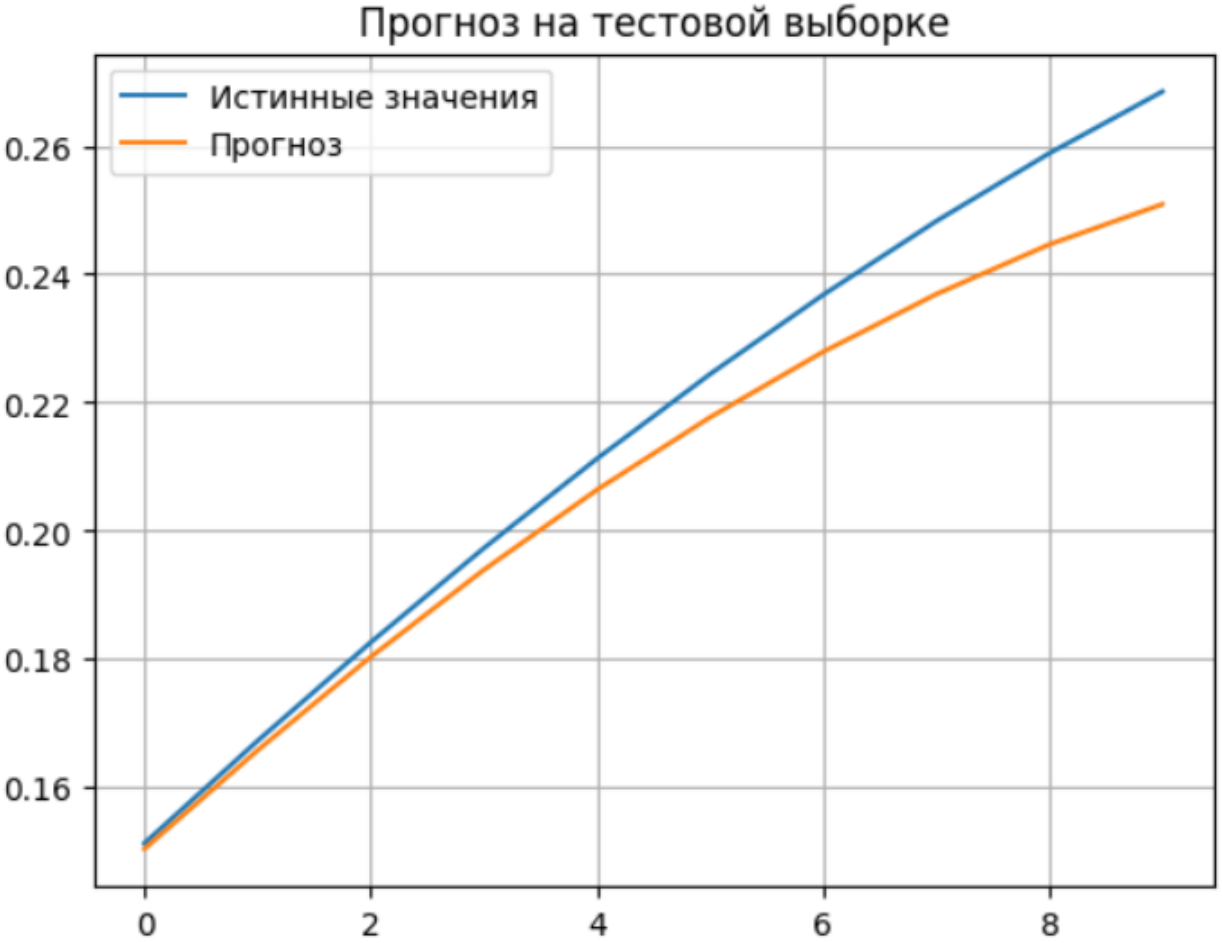
```
y_test_pred=[]
for i in Y_test:
  print(i)
  _, _, y_pred = forward(X_test)
  y_test_pred.append(y_pred.item())

  print(f"{X_test}{y_pred}")
  X_test = np.append(X_test[1:], y_pred.item())
print(Y_test,y_test_pred)
print("    Эталон | Предсказано | Отклонение")

for y,y_pred in zip(Y_test,y_test_pred):
  print(f"{y:.4f}", end="     |")
  print(f"{y_pred:.4f}", end="       |")
  print(f"{(y-y_pred):.4f}")
```

```
Эталон | Предсказано | Отклонение
0.1512     |0.1503       |0.0009
0.1672     |0.1657       |0.0015
0.1825     |0.1802       |0.0023
0.1972     |0.1938       |0.0034
0.2112     |0.2063       |0.0049
0.2244     |0.2176       |0.0067
0.2368     |0.2279       |0.0089
0.2483     |0.2369       |0.0114
0.2589     |0.2446       |0.0142
0.2685     |0.2509       |0.0176
```

```
mse=(Y_test-y_test_pred)**2
#for y, y_pred  in zip(Y_test)
print(mse)
plt.plot(mse)
plt.title("Изменение ошибки MSE")
plt.xlabel("Итерация")
plt.ylabel("Ошибка")
plt.grid()
plt.show()
```

```
plt.plot(Y_test, label="Истинные значения")
plt.plot(y_test_pred, label="Прогноз")
plt.title("Прогноз на тестовой выборке")
plt.legend()
plt.grid()
plt.show()
```





Вывод: В ходе работы была реализована нейронная сеть для решения задачи прогноза ряда данных. Проведённое обучение показало, что даже простая однослойная нелинейная ИНС способна достаточно точно аппроксимировать заданную функцию. С увеличением количества эпох обучения нейросеть давала все более точные значения.