Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра ІПІ

Звіт

з лабораторної роботи № 4 з дисципліни
«Методи та технології штучного інтелекту»

„Моделювання функції двох змінних з двома входами і одним виходом на основі нейронних мереж”

Виконав(ла) *ІП-14 Шляхтун Денис Михайлович*
(шифр, прізвище, ім'я, по батькові)

Перевірив *Шимкович Володимир Миколайович*
(прізвище, ім'я, по батькові)

Київ 2023

Мета: Дослідити структуру та принцип роботи нейронної мережі. За допомогою нейронної мережі змоделювати функцію двох змінних.

Постановка задачі.

За допомогою програмних засобів моделювання або мови програмування високого рівня створити та дослідити вплив кількості внутрішніх шарів та кількості нейронів на середню відносну помилку моделювання для різних типів мереж (feed forward backprop, cascade - forward backprop, elman backprop):

1. Тип мережі: feed forward backprop:
    a) 1 внутрішній шар з 10 нейронами;
    b) 1 внутрішній шар з 20 нейронами;
2. Тип мережі: cascade - forward backprop:
    a) 1 внутрішній шар з 20 нейронами;
    b) 2 внутрішніх шари по 10 нейронів у кожному;
3. Тип мережі: elman backprop:
    a) 1 внутрішній шар з 15 нейронами;
    b) 3 внутрішніх шари по 5 нейронів у кожному;
4. Зробити висновки на основі отриманих даних.

24.	$y = x \cdot \cos(2x) + \sin(x/2)$ $z = \sin(y) + \cos(x/2)$	24.	7.	18.

Виконання завдання.

Для виконання завдання була обрана мова програмування високого рівня Python.

Функції побудови моделей різних типів мереж:

```
def create_feed_forward_model(neurons_per_layer):
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Dense(neurons_per_layer, activation='relu'))
    model.add(tf.keras.layers.Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    return model

def create_cascade_forward_model(neurons_per_layer, hidden_layers=1):
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Dense(neurons_per_layer, activation='relu'))
```

```
    for _ in range(hidden_layers - 1):
        model.add(tf.keras.layers.Dense(neurons_per_layer, activation='relu'))
    model.add(tf.keras.layers.Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    return model

def create_elman_model(neurons_per_layer, hidden_layers=1):
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Dense(neurons_per_layer, activation='relu',
input_shape=(None, 2)))
    for _ in range(hidden_layers):
        model.add(tf.keras.layers.SimpleRNN(neurons_per_layer, activation='relu',
return_sequences=True))
    model.add(tf.keras.layers.Dense(1))
    model.compile(optimizer='adam', loss='mean_squared_error')
    return model
```

Функція відображення графіку результатів:

```
def plot_graph(x_train, x_test, z_train, z_test, z_predicted, label):
    plt.plot(x_train, z_train, 'o', label="Training value", markersize=1)
    plt.plot(x_test, z_test, label="True value")
    plt.plot(x_test, z_predicted, label="Predicted value")
    plt.title(label+"\nMSE: "+str(MSE(z_test, z_predicted)))
    plt.legend()
    plt.grid(True)
    plt.show()
    return

def MSE(true, pred):
    return np.mean((true - pred) ** 2)
```

Підготуємо дані для використання моделями:

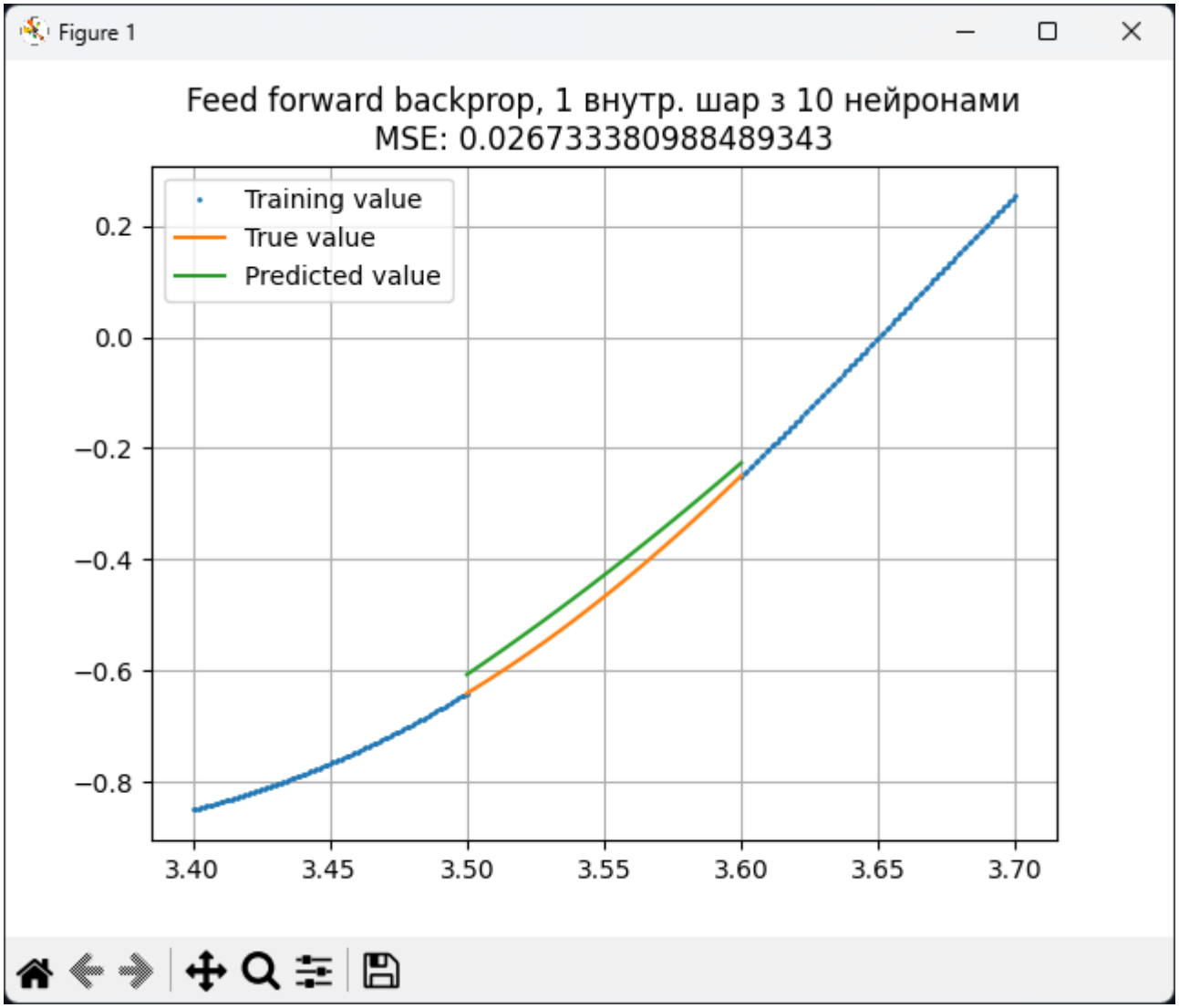
```
x_train = np.concatenate((np.arange(3.4, 3.5, 0.001), np.arange(3.6, 3.7, 0.001)))
x_test = np.arange(3.5, 3.6, 0.001)

y_train, y_test  = get_y(x_train), get_y(x_test)
z_train, z_test = get_z(x_train, y_train), get_z(x_test, y_test)
train_values = np.vstack((x_train, y_train)).T
test_values = np.vstack((x_test, y_test)).T
train_values_elman = train_values.reshape(train_values.shape[0], 1, 2)
test_values_elman = test_values.reshape(test_values.shape[0], 1, 2)
```

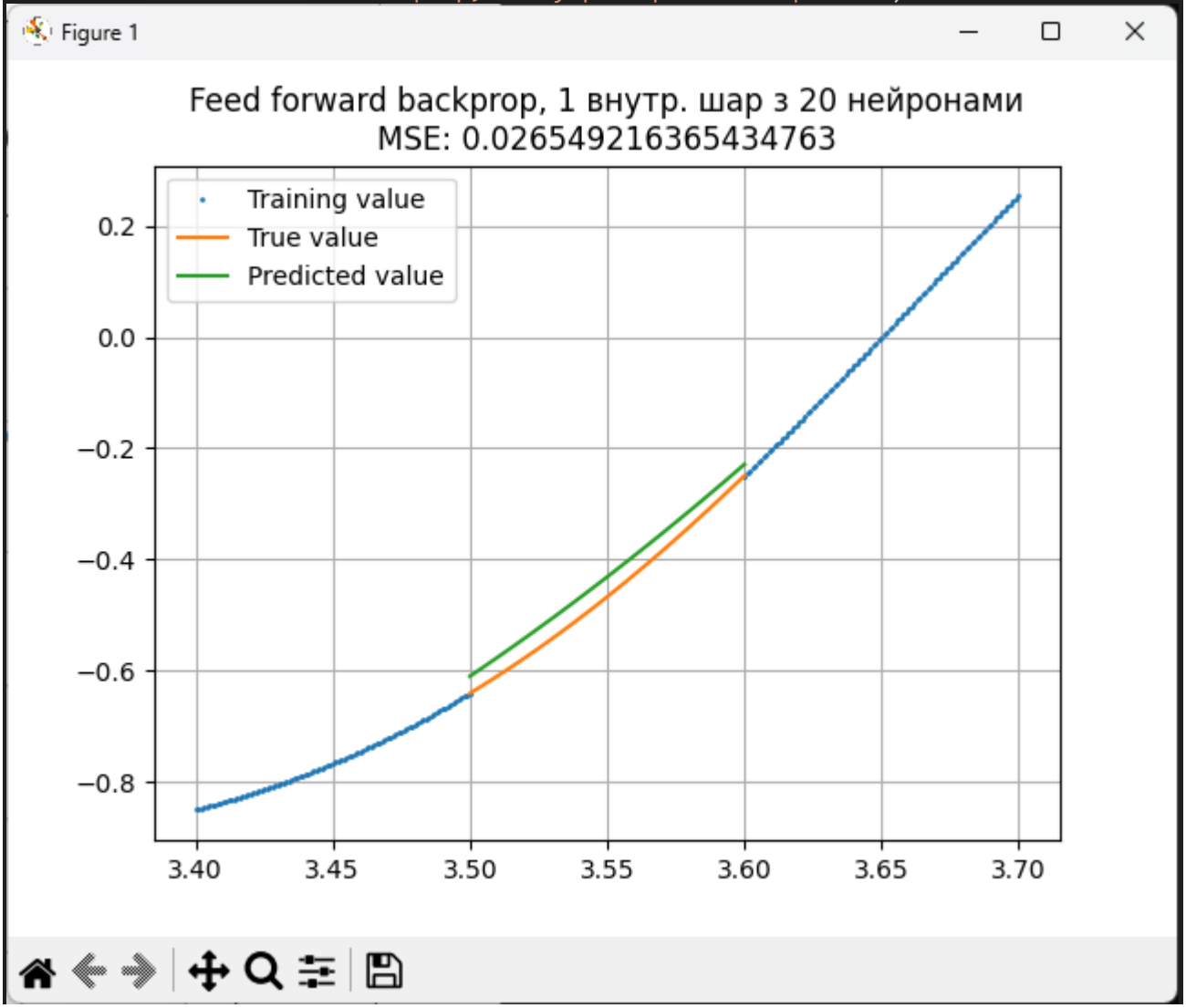
Створимо моделі мереж і протестуємо їх:

```
# Feed forward backprop, 1 внутрішній шар з 10 нейронами
model = create_feed_forward_model(10)
model.fit(train_values, z_train, epochs=500, verbose=0)
```

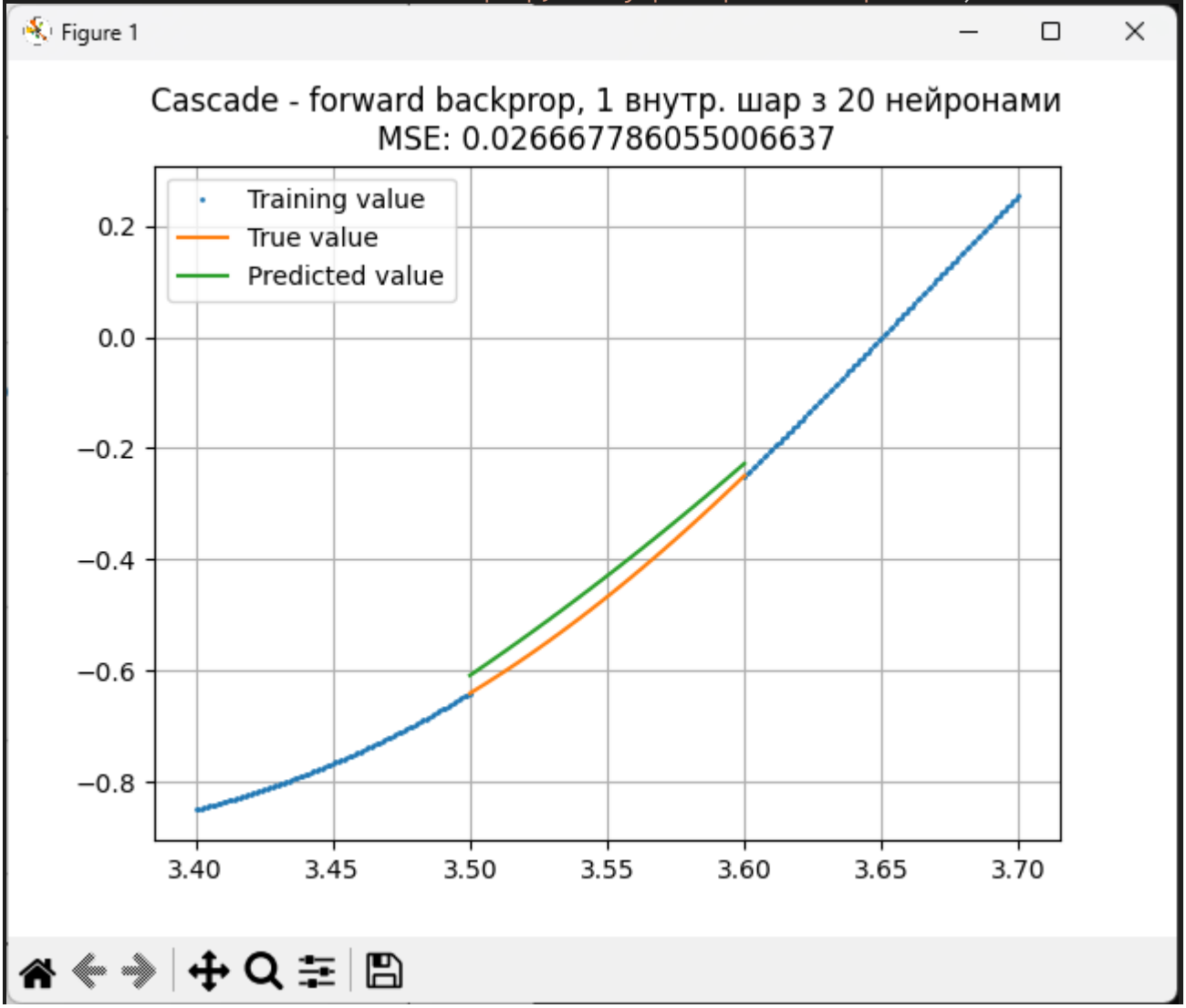
```
predicted_values = model.predict(test_values)
plot_graph(x_train, x_test, z_train, z_test, predicted_values,
           "Feed forward backprop, 1 внутр. шар з 10 нейронами")
```



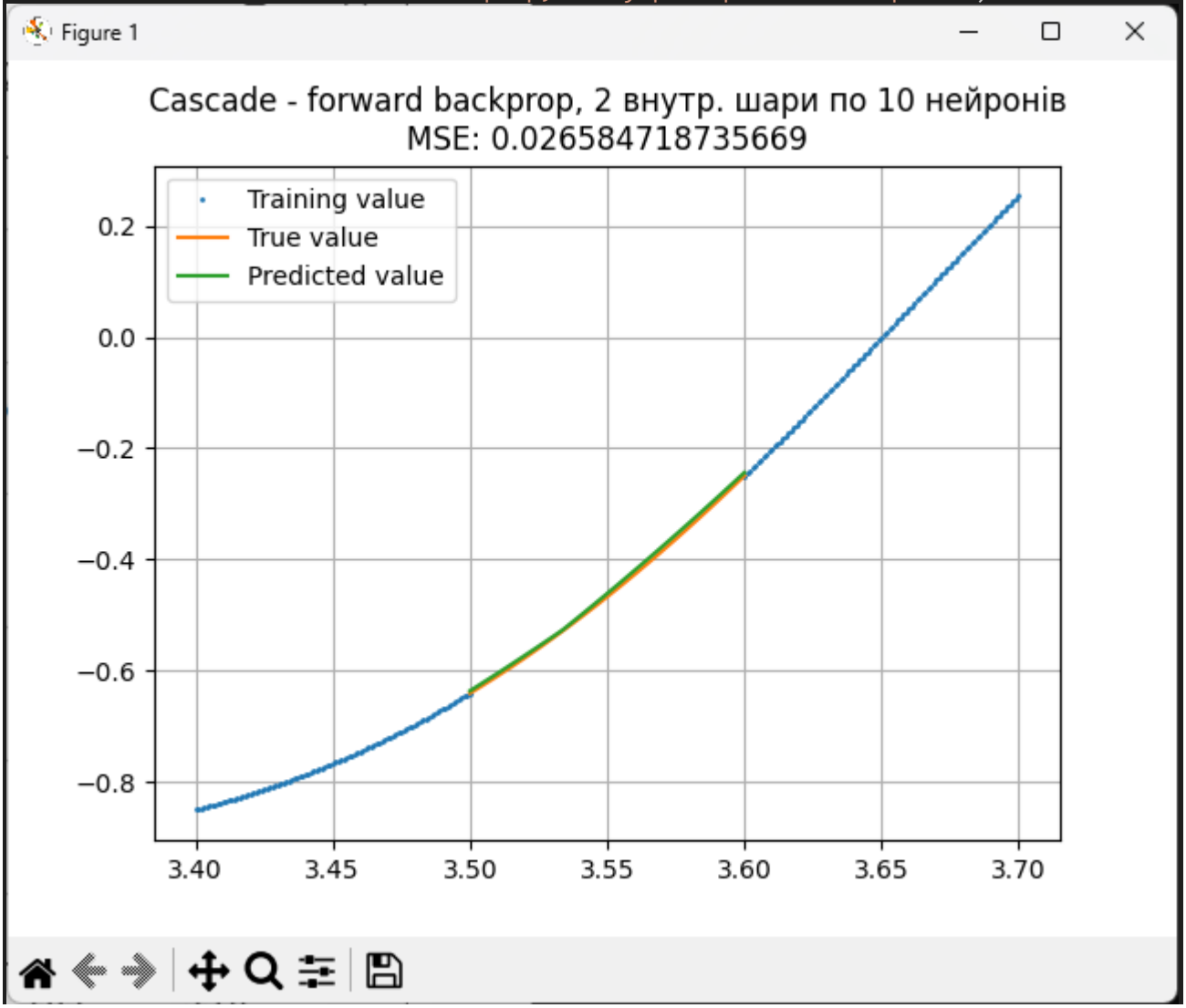
```python
# Feed forward backprop, 1 внутрішній шар з 20 нейронами
model = create_feed_forward_model(20)
model.fit(train_values, z_train, epochs=500, verbose=0)
predicted_values = model.predict(test_values)
plot_graph(x_train, x_test, z_train, z_test, predicted_values,
           "Feed forward backprop, 1 внутр. шар з 20 нейронами")
```



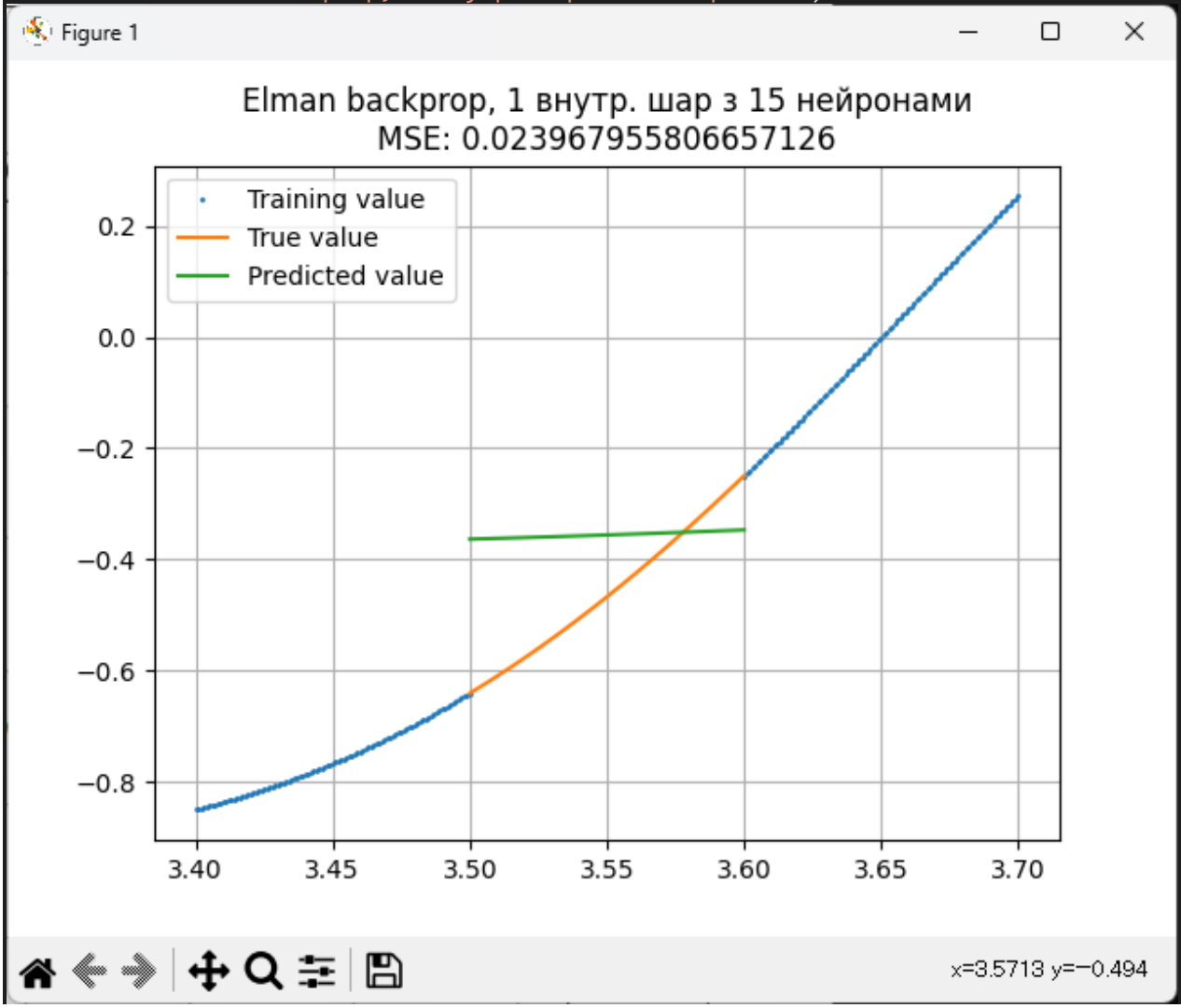
```python
# Cascade - forward backprop, 1 внутрішній шар з 20 нейронами
model = create_cascade_forward_model(20)
model.fit(train_values, z_train, epochs=500, verbose=0)
predicted_values = model.predict(test_values)
plot_graph(x_train, x_test, z_train, z_test, predicted_values,
           "Cascade - forward backprop, 1 внутр. шар з 20 нейронами")
```



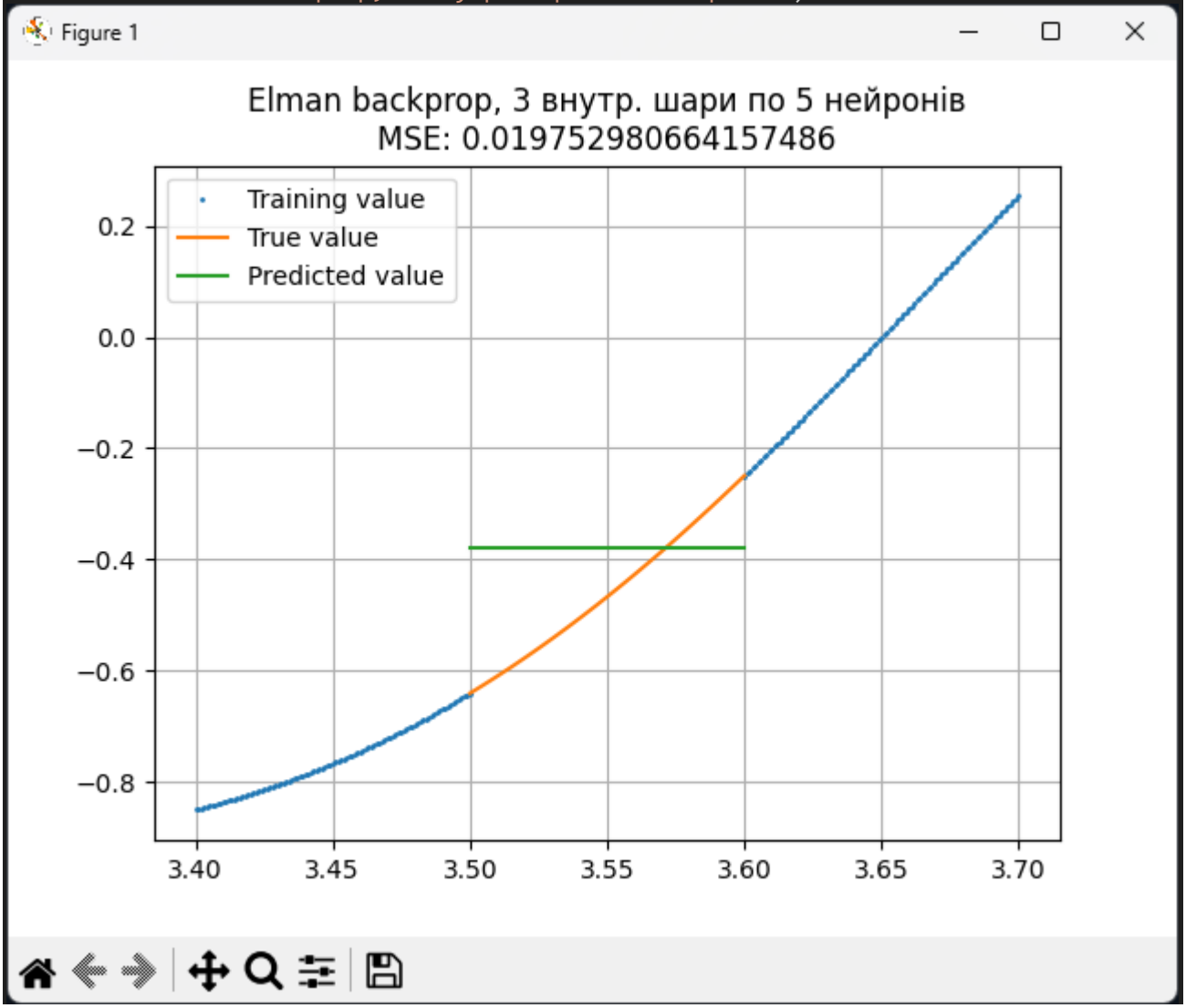
```
# Cascade - forward backprop, 2 внутрішніх шари по 10 нейронів у кожному
model = create_cascade_forward_model(10, 2)
model.fit(train_values, z_train, epochs=500, verbose=0)
predicted_values = model.predict(test_values)
plot_graph(x_train, x_test, z_train, z_test, predicted_values,
           "Cascade - forward backprop, 2 внутр. шари по 10 нейронів")
```



```
# Elman backprop, 1 внутрішній шар з 15 нейронами
model = create_elman_model(15)
model.fit(train_values_elman, z_train, epochs=500, verbose=0)
predicted_values = model.predict(test_values_elman)
plot_graph(x_train, x_test, z_train, z_test, predicted_values[:,0,:],
           "Elman backprop, 1 внутр. шар з 15 нейронами")
```



```
# Elman backprop, 3 внутрішніх шари по 5 нейронів у кожному
model = create_elman_model(5, 3)
model.fit(train_values_elman, z_train, epochs=500, verbose=0)
predicted_values = model.predict(test_values_elman)
plot_graph(x_train, x_test, z_train, z_test, predicted_values[:,0,:],
           "Elman backprop, 3 внутр. шари по 5 нейронів")
```



Висновок.

При виконанні лабораторної роботи було досліджено структуру та принцип роботи нейронної мережі. За допомогою нейронної мережі змодельовано функцію двох змінних. Було використано feed forward backprop, cascade - forward backprop та elman backprop, значення MSE усіх моделей майже не відрізнялося, але візуально найкраще впоралася cascade - forward backprop з 2 внутрішніми шарами по 10 нейронів.