Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №1

По дисциплине: «МиАПР»

Тема: «Линейная искусственная нейронная сеть. Правило Видроу-Хоффа»

Выполнил:

Студент 2 курса

Группы ПО-7(2)

Мозоль И.П.

Проверил:

Крощенко А.А.

Брест 2021

Вариант 1

**Цель работы:** изучить обучение и функционирование линейной ИНС при решении задач прогнозирования.

**Задание:** Написать на любом ЯВУ программу моделирования прогнозирующей линейной ИНС. Для тестирования использовать функцию:

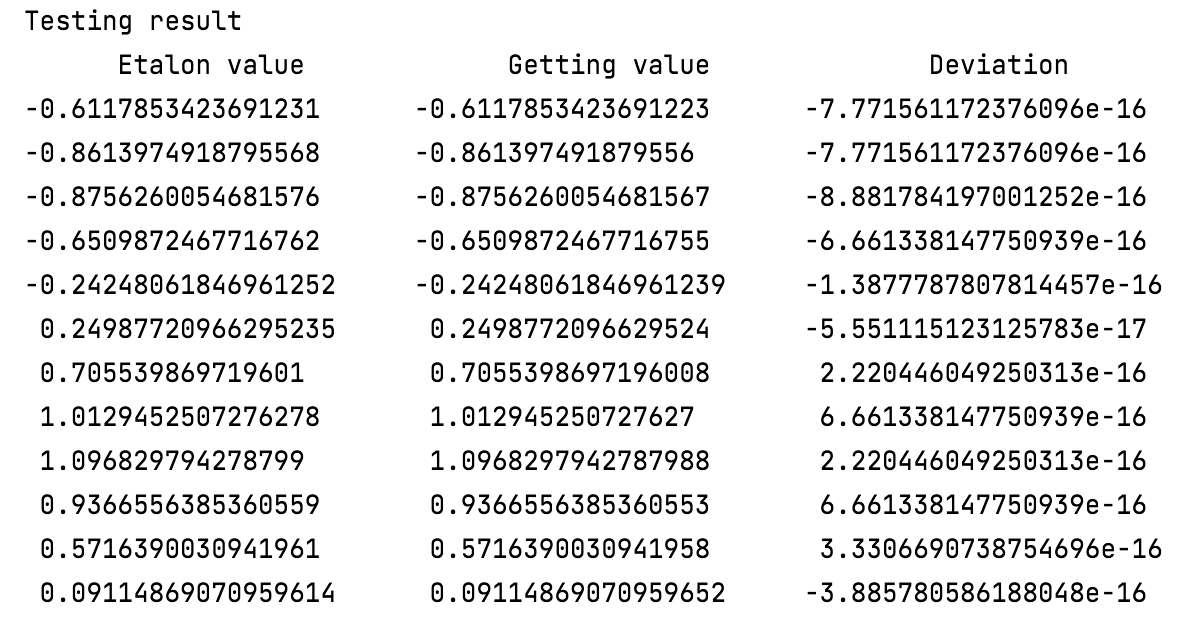
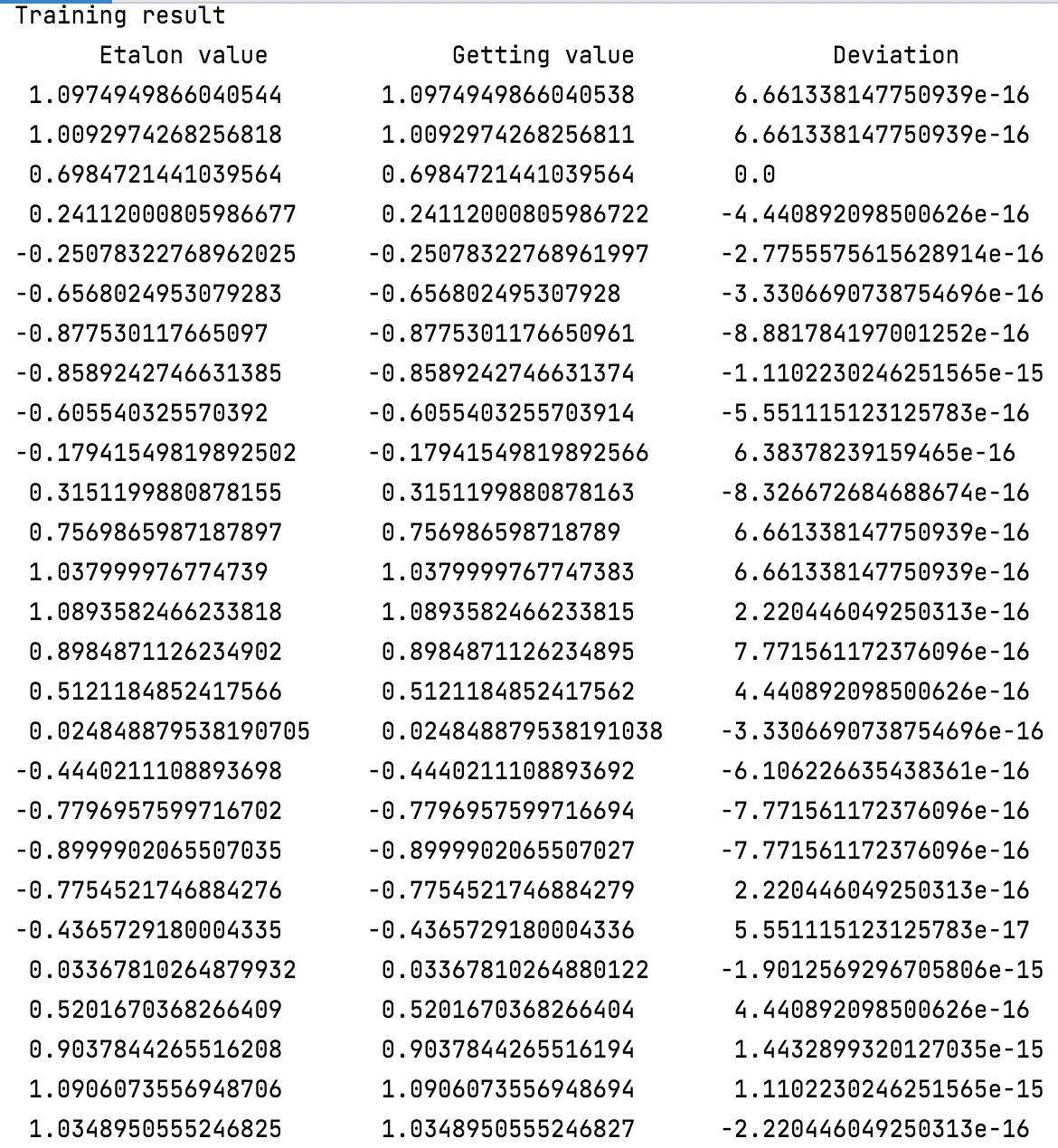
y = a\*sin(bx) + d

a = 1, b = 5, d = 0.1, кол-во входов ИНС = 1.

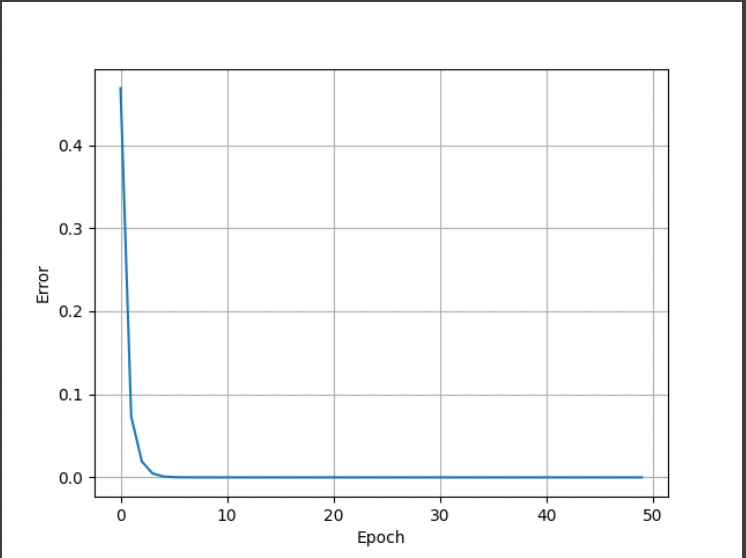
**Код программы:**

**import** math  
**import** random  
**import** matplotlib.pyplot **as** plt  
  
  
**def** etalonn(x):  
 a, b, d = 1, 5, 0.1 *# parametres for calculate reference values* **return** a \* math.sin(b \* x) + d  
  
  
**def** getting\_value(w\_arr, etalon\_arr, T, i, amount\_inputs) -> float:  
 value = 0  
 **for** j **in** range(amount\_inputs):  
 value += w\_arr[j] \* etalon\_arr[j + i]  
 **return** value - T  
  
  
**def** calc\_w(w\_arr, alpha, etalon\_arr, value, i, amount\_inputs) -> list:  
 **for** j **in** range(len(w\_arr)):  
 w\_arr[j] -= alpha \* etalon\_arr[i + j] \* (value - etalon\_arr[i + amount\_inputs])  
 **return** w\_arr  
  
  
**def** calc\_T(T, alpha, etalon\_arr, value, i, amount\_inputs) -> float:  
 T += alpha \* (value - etalon\_arr[i + amount\_inputs])  
 **return** T  
  
  
**def** main():  
 amount\_inputs = 3 *# amount of input neuron* alpha = 0.1 *# training speed* min\_error = 1.0e-30  
 step = 0.1 *# tabulation step* amount\_train = 30 *# number training input* amount\_test = 15 *# number testing input* w\_arr = [  
 random.uniform(0, 1) **for** i **in** range(amount\_inputs)  
 ] *# generate weights in array* T = random.uniform(0, 1) *# generate limit* etalon\_arr = [  
 etalonn(i \* step) **for** i **in** range(amount\_train)  
 ] *# reference values for training* etalon\_for\_testing = [ *# reference values for testing* etalonn(i \* step) **for** i **in** range(amount\_train, amount\_test + amount\_train)  
 ]  
 arr\_chart = [] *# array for build chart* error = 10  
  
 **while** error > min\_error:  
 error = 0  
 **for** i **in** range(amount\_train - amount\_inputs):  
 value = getting\_value(w\_arr, etalon\_arr, T, i, amount\_inputs)  
 w\_arr = calc\_w(w\_arr, alpha, etalon\_arr, value, i, amount\_inputs)  
 T = calc\_T(T, alpha, etalon\_arr, value, i, amount\_inputs)  
 error += (value - etalon\_arr[i + amount\_inputs]) \*\* 2  
 error /= amount\_train - amount\_inputs  
 arr\_chart.append(error)  
  
 print(**"Training end\nTraining result"**)  
 print(**"{:^25}{:^25}{:^25}"**.format(**"Etalon value"**, **"Getting value"**, **"Deviation"**))  
 **for** i **in** range(amount\_train - amount\_inputs):  
 value = getting\_value(w\_arr, etalon\_arr, T, i, amount\_inputs)  
 print(  
 **"{:< 25}{:< 25}{:< 25}"**.format(  
 etalon\_arr[i + amount\_inputs],  
 value,  
 etalon\_arr[i + amount\_inputs] - value,  
 )  
 )  
 print(**"Testing result"**)  
 print(**"{:^25}{:^25}{:^25}"**.format(**"Etalon value"**, **"Getting value"**, **"Deviation"**))  
 **for** i **in** range(amount\_test - amount\_inputs):  
 value = getting\_value(w\_arr, etalon\_for\_testing, T, i, amount\_inputs)  
 print(  
 **"{:< 25}{:< 25}{:< 25}"**.format(  
 etalon\_for\_testing[i + amount\_inputs],  
 value,  
 etalon\_for\_testing[i + amount\_inputs] - value,  
 )  
 )  
 plt.plot(arr\_chart)  
 plt.xlabel(**"Epoch"**)  
 plt.ylabel(**"Error"**)  
 plt.grid()  
 plt.show()  
  
  
main()

**Результат:**



**График изменения ошибки в зависимости от количества эпох:**



**Вывод:** Я изучил обучение и функционирование линейной ИНС при решении задач прогнозирования. Программа написана на языке python.