Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №11-12**

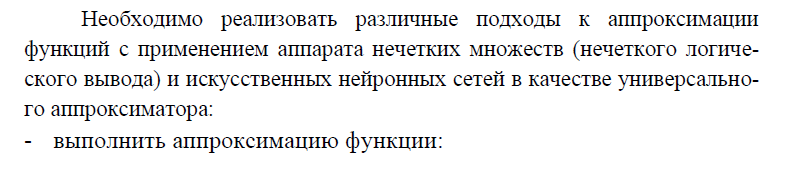
**Дисциплина: Нейросетевые и нечеткие модели**

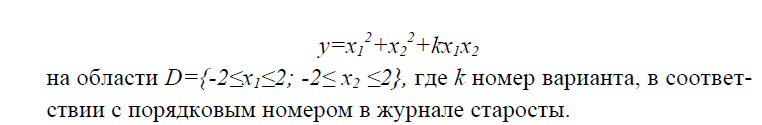
Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Мингазетдинов Р.Р.

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А. Крамаренко

Задание (Вариант 10):





1. Запускаем Matlab и выполняем одноименную команду NNTool, который позволит нам выбирать структуры NN из обширного перечня и предоставляет множество алгоритмов обучения для каждого типа сети. (смотреть рисунок 1)

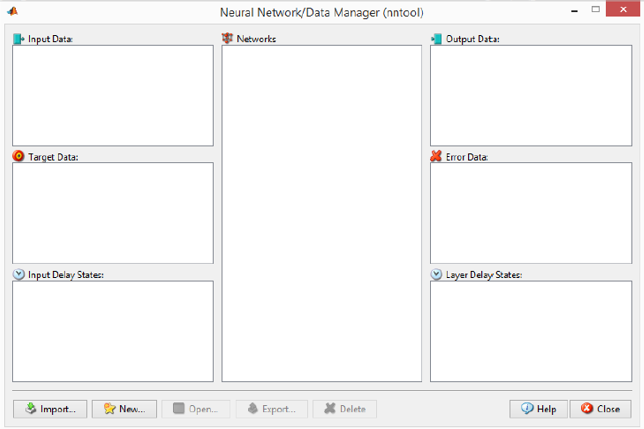


Рисунок 1 – Главное окно NNTool

1. Теперь рассмотрим создание нейронной сети с помощью NNTool на примере задачи аппроксимации. Это означает, что с помощью нейронных цепей можно аппроксимировать сколь угодно точно непрерывные функции многих переменных. Для начала необходимо получить обучающую выборку нашей функции , в области . Составим следующую программу:

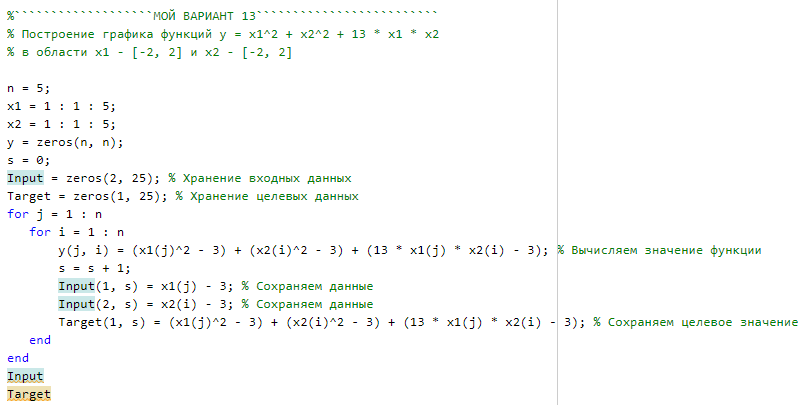


Рисунок 2 – Программа для нахождения обучающей выборки

Выполнение данной программы позволяет получить векторы входов, целевых значений. (смотреть рисунок 3)

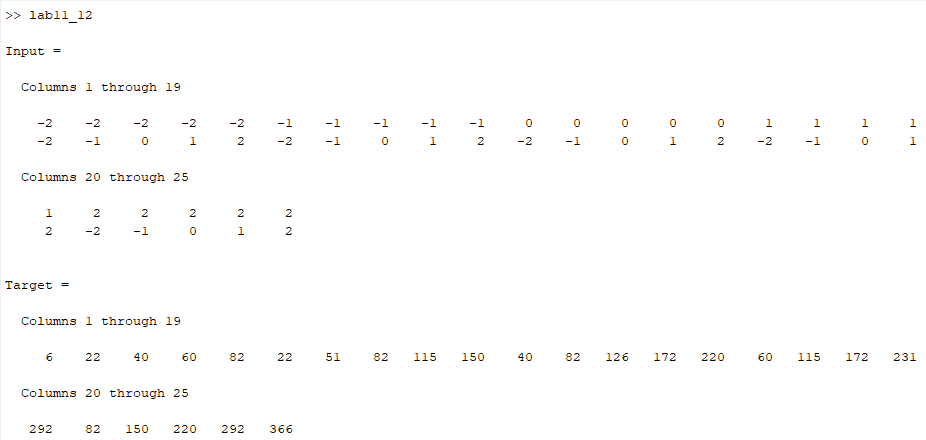


Рисунок 3 – Векторы входов, целевых значений

1. Возвращаемся к NNTool и продолжаем работать, создадим новую сеть, выберем персептрон (Feed-Forward backprop) с 25 сигмоидными нейронами (TANSIG) скрытого слоя и одним линейным (PURELIN) нейроном выходного слоя. (смотреть рисунок 4)

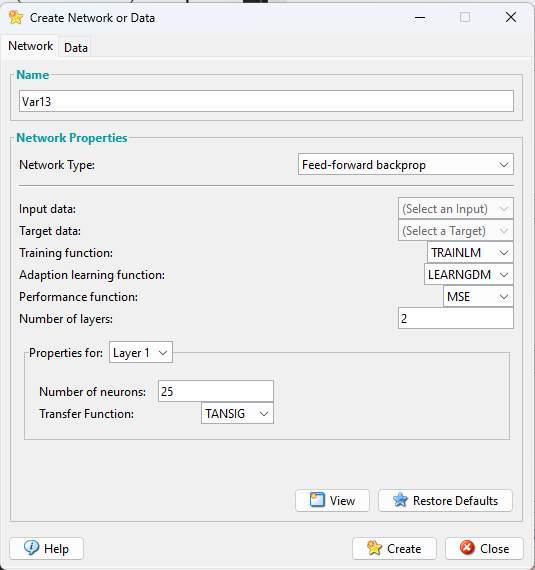
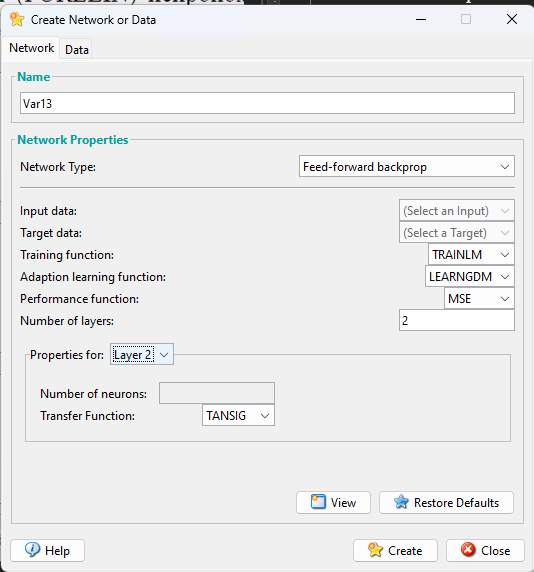
 

Рисунок 4 – Создание сети

Также зададим входные и целевые значения, полученные ранее. (смотреть рисунок 5)

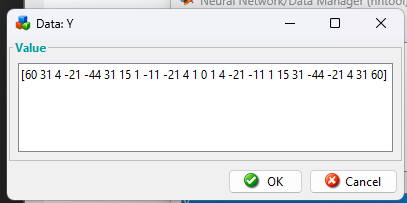
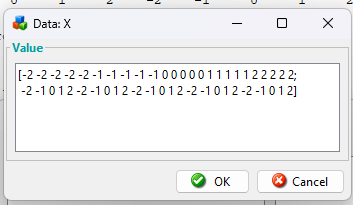


Рисунок 5 – Создание сети

Завершаем создание сети и получаем следующую структуру. (смотреть рисунок 6)

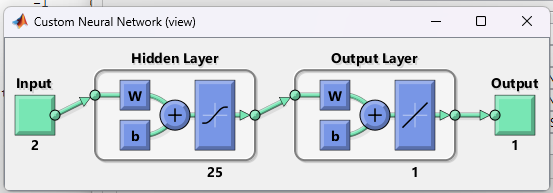


Рисунок 6 – Вид созданной сети

1. Следующий этап — это обучение созданной сети. Обучение будем производить, используя алгоритм обратного распространения ошибки, с минимизации функции ошибки по методу Левенберга-Маркардта, который реализует функция TRAINLM. Функция ошибки – MSE. (смотреть рисунок 7)

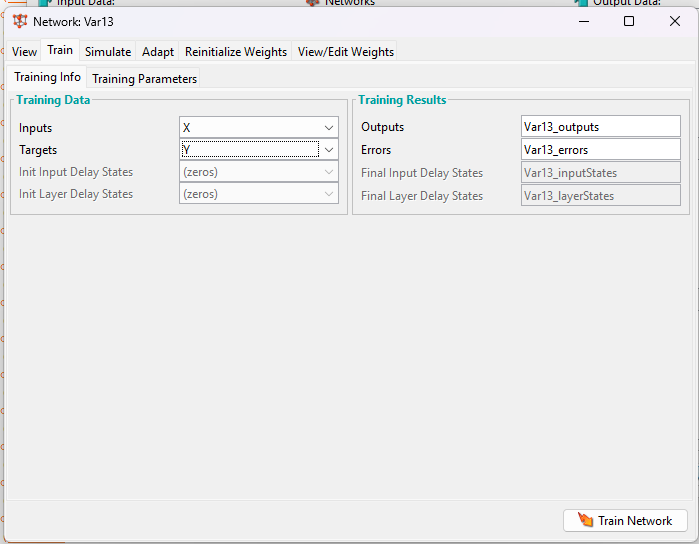


Рисунок 7 – Обучение сети

Нужно указать в качестве обучающей выборки значения векторов входов и целей X, Y и нажать на «Train Network» и далее мы получим результаты. (смотреть рисунок 8 – 9)

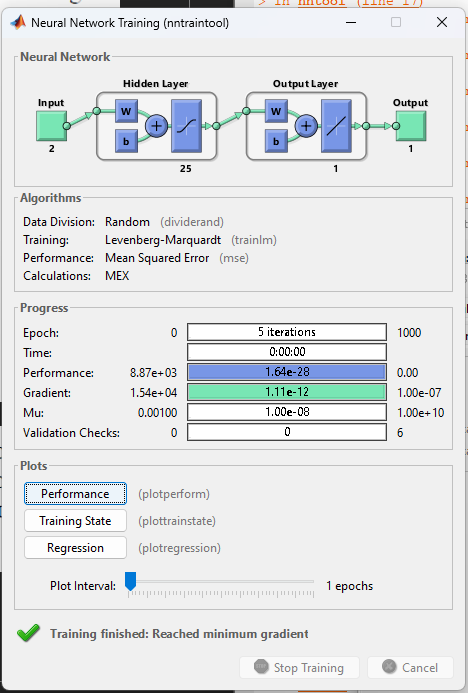


Рисунок 8 – Результаты обучения сети

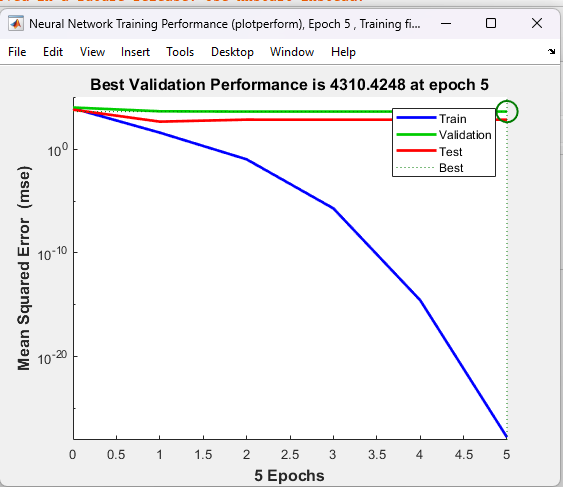


Рисунок 9 – Результаты обучения сети

После обучения сети необходимо проверить насколько хорошо сеть приближает функцию, для этого укажем в «Simulate» вектор входных значений и затем нажмём «Simulate Network». (смотреть рисунок 10)

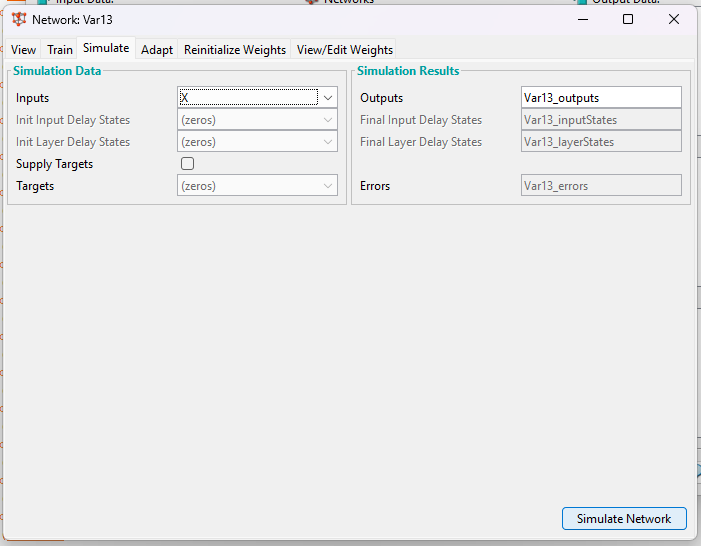
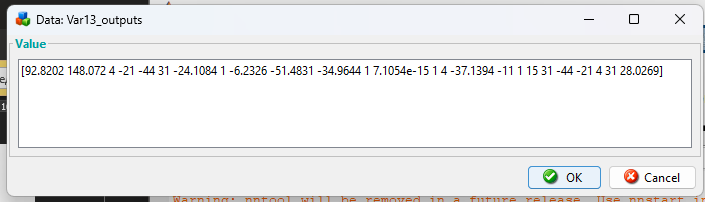


Рисунок 10 – Проверка работы сети

Вернемся к главному окну NNTool и посмотрим результаты работы сети. (смотреть рисунок 11)



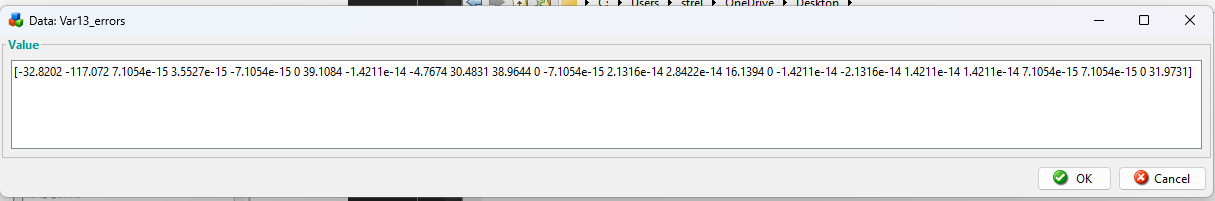
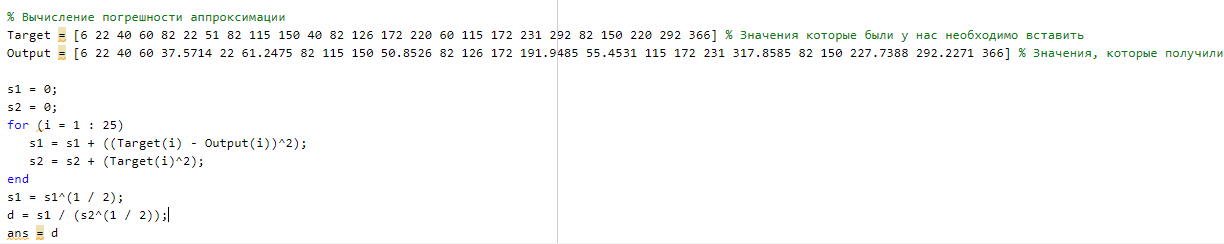


Рисунок 11 – Результаты обучения сети: выходы и ошибки

1. Теперь необходимо найти оценку погрешности аппроксимации, для этого надо найти корень из суммы квадратов разностей между желаемыми и получаемыми значениями. Напишем код для вычисления погрешности аппроксимации и запустим его. (смотреть рисунок 12)



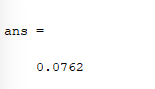


Рисунок 12 – Результат относительной погрешности примерно 7.6%

Результат указывает на хороший уровень точности т.е. модель стабильна.

**Вывод:** ознакомился с графическим интерфейсом пользователя NNTool, создал и обучил модель, нашел оценку погрешности согласно варианту.