ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Ордена трудового красного знамени федеральное государственное бюджетное образовательное учреждение высшего образования

МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ

Практическая работа № 3

Применение однослойной нейронной сети  
 с линейной функцией активации  
 для прогнозирования временных рядов

                                                                             Выполнил студент

Группы М092101(71)

Ландызин А.Н.

Москва, 2021

**Цель работы —** изучить возможности однослойных НС в задачах прогнозирования временных рядов методом скользящего окна (авторегрессия).

***Постановка задачи.***

На временном интервале  задан дискретный набор значений функции *x(t)*. Количество точек *N = 20*, расположение — равномерное.

* Методом «скользящего окна» спрогнозировать поведение функции *x(t)*на *N* точках последующего интервала Для решения использовать однослойную НС с количеством нейронов *р* и линейной функцией активации. Исходное количество нейронов (длина окна) . Обучение проводить методом Видроу — Хоффа.
* Исследовать влияние количества эпох *М* обучения и коэффициента обучения на среднеквадратичную погрешность приближения .
* Исследовать процесс прогнозирования при постепенном изменении (уменьшении/увеличении) размера окна *р*. Сделать выводы по результатам численного эксперимента.

***Варианты заданий***

**Варианты функций**

|  |  |  |  |
| --- | --- | --- | --- |
| № Варианта | Функция x(t) | a | b |
| 6 |  | 1 | 5 |

# Выполнение:

## Исследование процесса прогнозирования при постепенном изменении размера окна р

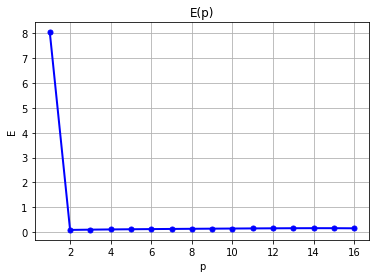


График 1.

На графике 1 представлена зависимость среднеквадратичной ошибки от размера окна. Окно изменяется с шагом 1 в промежутке [1;16].

## Исследование влияния коэффициента обучения n на среднеквадратичную погрешность приближения

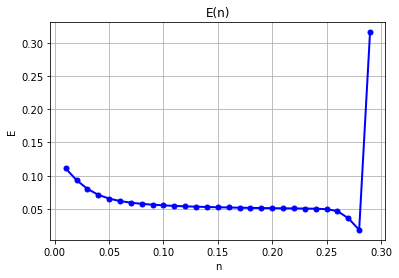


График 2.

На графике 2 представлена зависимость среднеквадратичной ошибки от коэффициента обучения. Коэффициент обучаемости изменяется с шагом 0.01 в промежутке [0.01; 0.3]

## Исследование влияния количества эпох М обучения на среднеквадратичную погрешность приближения

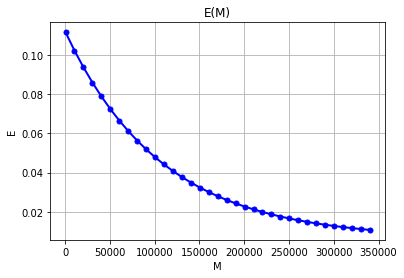


График 3.

На графике 3 представлена зависимость среднеквадратичной ошибки от количества эпох. Количество эпох обучения изменяется с шагом в 10000 в промежутке [400; 340400].

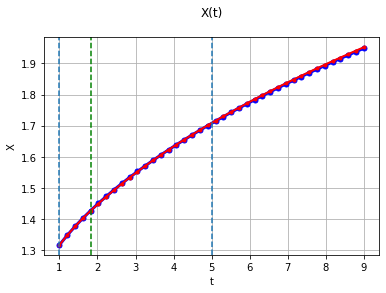


График 4.

На графике 4 представлены графики исходной и обученной функции в зависимости от временного ряда. Синим цветом обозначен график исходной функции, красным – полученной. Интервал, на котором обучалась НС, обозначен голубым пунктиром. Зеленым пунктиром показана ширина окна. НС в данном примере обучалась 350000 эпох, с шагом обучения n = 0.01, при ширине окна p = 4

# Выводы:

В процессе лабораторной работы были изучены возможности однослойных НС в задачах прогнозирования временных рядов методом скользящего окна. Было установлено, что при увеличении ширины окна среднеквадратичная погрешность остается практически неизменной.

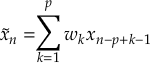
При увеличении нормы обучения значения среднеквадратичной погрешности уменьшалась, однако после преодоления порога 0.3 резко возрастает.

Сравнивая результаты прогноза при различном количестве эпох, следует отметить, что с увеличением количества эпох, среднеквадратичная ошибка уменьшается. И после преодоления порога в 350000 эпох, прогнозные значения практически полностью совпадают с изначальной функцией.

# Контрольные вопросы

1. В чем состоит принцип прогнозирования на основе авторегрессии?

Авторегрессионная модель основана на выражении прогнозируемого значения ряда в момент времени n>m через его известные значения в предыдущие моменты времени:



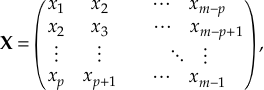
где *р* ***—*** размер «окна» данных, по которому производится прогноз; ***—*** некоторые весовые коэффициенты. Выражение используется для центрированных временных рядов, среднее значение которых равно 0.

Ошибка прогноза (локальная) равна апостериорной разнице спрогнозированного и реального значений временного ряда:

.

1. Объясните методику обучения НС прогноза.

В режиме обучения последовательно, от эпохи к эпохе, на вход сети подаются элементы векторов-столбцов обучающей выборки:



а на выходе соответственно получаются прогнозируемые значения:



Эти значения сравниваются с реальными



и по формуле  оценивается ошибка. Коррекция весов на каждой эпохе производится по правилу Видроу — Хоффа:



где — норма обучения, .

Если по достижении правого края выборки суммарная среднеквадратичная ошибка



останется достаточно большой, следует продолжить обучение, снова вернувшись к первому столбцу выборки и т. д.

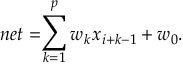
1. Поясните принцип функционирования НС прогноза.

Нахождение неизвестных весовых коэффициентов  осуществляется по известной обучающей выборке значений временного ряда  Для этого необходимо определить окно длиной *.* Далее, начиная с левого края временного ряда, следует прогнозировать его значения в моменты, идущие непосредственно за окном (справа от него). Сравнивая этот прогноз с реальными значениями, можно оценить ошибку , а на ее основе — скорректировать весовые коэффициенты , например, с помощью методов обучения НС.

Функция активации — линейная, т. е.



где net — комбинированный вход единственного нейрона,



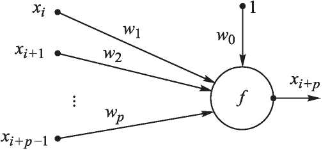


Рис. 1 – Простейшая архитектура НС.

# Приложения

import numpy as np

import matplotlib.pyplot as plt

*def* X(*T*): # возвращает результат моделируемой функции от значения времени

    return [np.sqrt(0.1 \* t) + 1  for t in T]

N = 20 # количество точек

a = 1 # интервал

b = 5

T = *list*(np.linspace(a, 2 \* b - a, 2 \* N))

RightX = X(T)

TryX = [0 for i in range(N)]

*def* DeltaW(*x*, *q*, *n*): # находит величину, на которую изменятся Wi, для пороговой ФА

    return n \* q \* x

*def* Net(*x*, *w*): # находит значение сетевого входа НС

    return sum([w\_i \* x\_i for w\_i, x\_i in zip(w, x)])

*def* MeanSquareError(*RightX*, *TryX*, *p*):

    summa = 0

    for rx\_i, tx\_i in zip(RightX[p:], TryX[p:]):

        summa += (rx\_i - tx\_i) \*\* 2

    return summa \*\* 0.5

*def* Learning(*p*, *n*, *m*): # обучение НС методом скользящего окна

    for k in range(p): TryX[k] = RightX[k]

    w = [0] \* p

    era = 0

    while(era < m):

        for l in range(p, N): # 16 шагов эпохи

            TryX[l] = Net(RightX[l - p:l-1], w)

            q = RightX[l] - TryX[l]

            for k in range(0, p):

                w[k] += DeltaW(RightX[l - p + k], q, n)

        era += 1

        #print("\nera = ",  np.round(era, 3))

        #print("TryX : ", np.around(TryX, 3))

        #print("w : ", np.around(w, 3))

        #print("e = ", np.round(e, 3))

    print(np.around(TryX, 3))

    return *list*(TryX), w

*def* Graph(*TryX*, *p*, *arg* = "", *name* = ""): # строит 2 графика : исходной и полученной функции в зависмости от временного ряда

    fig, ax = plt.subplots()

    ax.plot(T, RightX, 'bo-', linewidth=3, markersize=5)

    ax.plot(T, TryX, 'ro-', linewidth=2, markersize=3)

    plt.title("X(t)\n" + name + *str*(arg))

    plt.xlabel('t')

    plt.ylabel('X')

    plt.axvline(x=a,  linestyle='--')

    plt.axvline(x=b, linestyle='--')

    plt.axvline(x=T[p], linestyle='--', color = 'g')

    plt.grid(True)

    plt.show()

*def* Graph\_E(*e*, *arg*, *name*):  # строит график зависимости ошибки от arg : n, p, m

    plt.plot(arg, e, 'bo-', linewidth=2, markersize=5)

    plt.title("E(" + name + ")")

    plt.xlabel(name)

    plt.ylabel('E')

    plt.grid(True)

    plt.show()

*def* Forecast(*E*, *n*, *p*, *m*):

    TryX, w = Learning(p, n, m)

    TryX.extend(np.zeros(N))

    for l in range(N, 2 \* N):

        TryX[l] = Net(RightX[l - p: l - 1], w)

    E.append(MeanSquareError(RightX[N:], TryX[N:], p))

    return TryX

if \_\_name\_\_=="\_\_main\_\_":

   # med\_n = 0.3  # норма обучения

   # range\_n = np.around(np.linspace(0.2, 1, 10), 1)

    med\_n = 0.01

    range\_n = np.around(np.linspace(0.01, 0.3, 30), 2)

    med\_p = 4 # размер "окна" данных

    #range\_p = range(1, 17)

    range\_p = range(1, 17, 1)

    med\_m = 1000 # количество эпох

    range\_m = range(400, 350000, 10000)

    E = []

    for p in range\_p:    # исследование относительно размера окна

        print("\n\np = ", p)

        Forecast(E, med\_n, p, med\_m)

    Graph\_E(E, range\_p, "p")

    E.clear()

    for n in range\_n:  # исследование относительно нормы обучения

        print("\n\nn = ", n)

        Forecast(E, n, med\_p, med\_m)

    Graph\_E(E, range\_n, "n")

    E.clear()

    for m in range\_m:    # исследование относительно количества эпох

        print("\n\nM = ", m)

        Forecast(E, med\_n, med\_p, m)

    Graph\_E(E, range\_m, "M")

    Graph(Forecast(E, med\_n, med\_p, 350000), med\_p)