Московский авиационный институт
(Национальный исследовательский университет)
Институт №8 «Информационные технологии и прикладная математика»

Кафедра вычислительной математики и программирования

Лабораторная работа № 7

по курсу «Нейроинформатика».

Тема: «Автоассоциативные сети с узким горлом».

Студент: Вельтман Л.Я.
Группа: 80-407Б
Преподаватели: Тюменцев Ю.В.
Аносова Н.П.

Вариант: 7

Оценка:

Москва, 2020

Цель работы.

Целью работы является исследование свойств автоассоциативных сетей с узким горлом, алгоритмов обучения, а также применение сетей для выполнения линейного и нелинейного анализа главных компонент набора данных.

Основные этапы работы.

1. Использовать автоассоциативную сеть с узким горлом для отображения набора данных, выделяя первую главную компоненту данных.

2. Использовать автоассоциативную сеть с узким горлом для аппроксимации кривой на плоскости, выделяя первую нелинейную главную компоненту данных.

3. Применить автоассоциативную сеть с узким горлом для аппроксимации пространственной кривой, выделяя старшие нелинейные главные компоненты данных.

Оборудование.

Операционная система: macOS Catalina version 10.15.5

Процессор: 2,3 GHz 2-ядерный процессор Intel Core i5

Оперативная память: 8 ГБ 2133 MHz LPDDR3

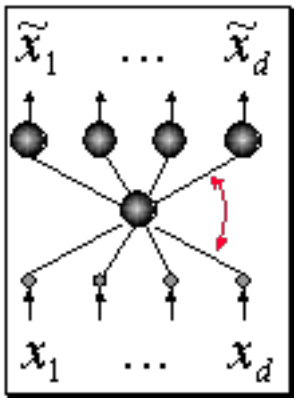
Программное обеспечение.

Работа выполнена на Python3 с применением библиотек numpy (для вычислений), pandas и matplotlib (для графиков) при помощи командной оболочки Jupyter Notebook.

Сценарий выполнения работы.

Автоассоциативной памятью — называют память, которая может завершить или исправить образ, но не может ассоциировать полученный образ с другим образом. Данный факт является результатом одноуровневой структуры ассоциативной памяти, в которой вектор

появляется на выходе тех же нейронов, на которые поступает входной вектор. Такие сети неустойчивы. Для устойчивой сети последовательные итерации приводят ко все меньшим изменениям выхода, пока в конце концов выход не становится постоянным. Для многих сетей процесс никогда не заканчивается. Неустойчивые сети обладают интересными свойствами и изучались в качестве примера хаотических систем. В определенном смысле, это может быть достигнуто и без обратных связей, например перцептроном для случаев когда устойчивость важнее изучения хаотических систем.



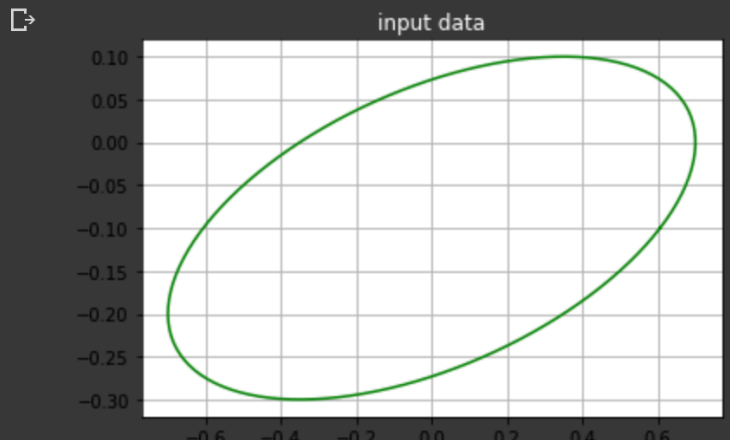
Входные данные и результаты

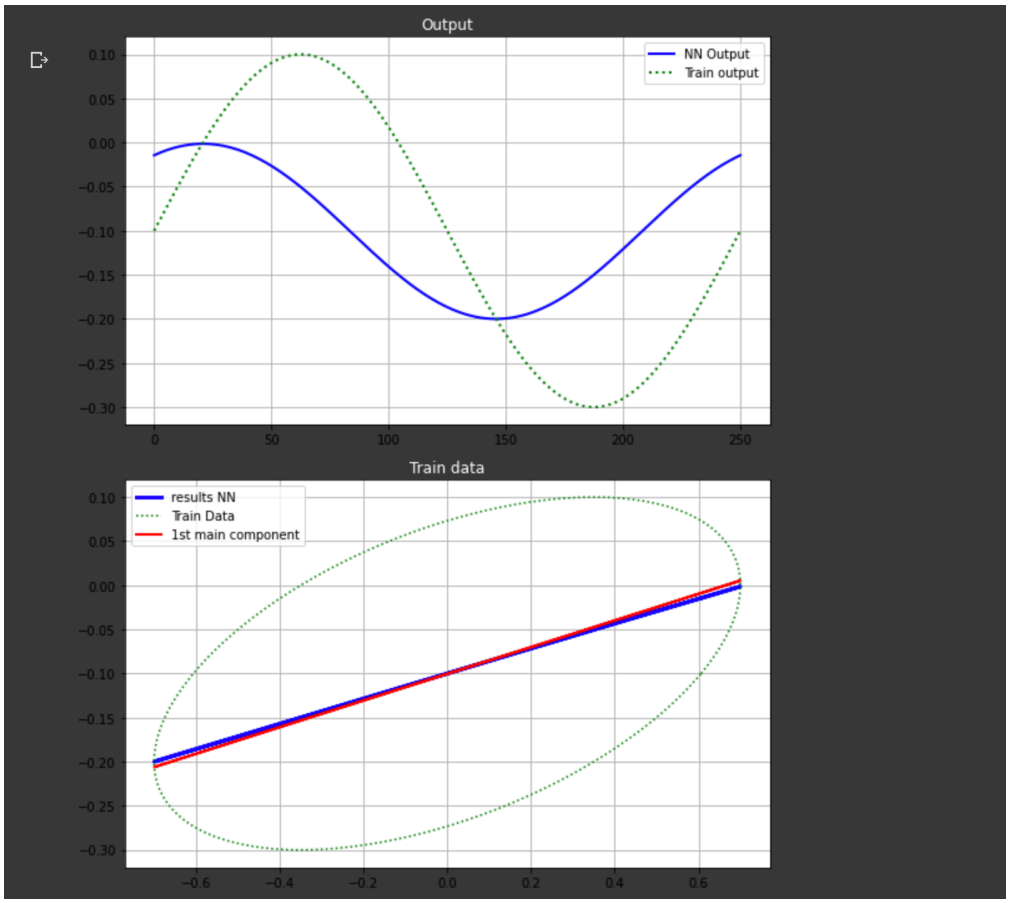
Задание 1

```
t = np.linspace(0, 2 * np.pi, int(2 * np.pi / 0.025), endpoint=True)

x = f(t)
y = g(t)

plt.plot(x, y, 'green')
plt.grid(True)
plt.title('input data', c='white')
plt.show()
```





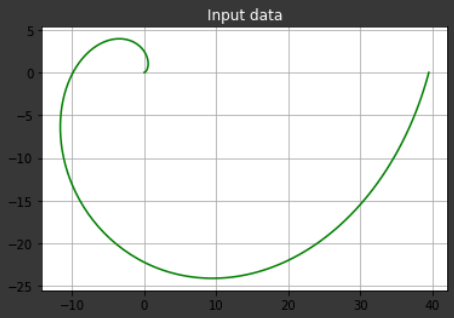
Задание 2

Задание 2

Использовать автоассоциативную сеть с узким горлом для аппроксимации кривой на плоскости, выделяя первую нелинейную главную компоненту данных.

```
phi = np.linspace(0, 2 * np.pi, int(2 * np.pi / 0.025), endpoint=True)
r = phi * phi
x2 = r * np.cos(phi)
y2 = r * np.sin(phi)

plt.plot(x2, y2, 'green')
plt.grid(True)
plt.title('Input data', c='white')
plt.show()
```

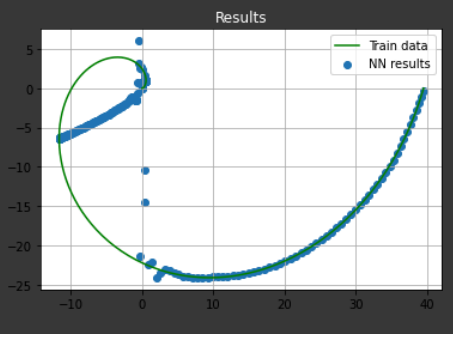


Число нейронов скрытого слоя задать равным [10,1,10]. Используем метод Левенберга-Марквардта в качестве алгоритма обучения.

```
[12] nn2 = pyrenn.CreateNN([1, 10, 1, 10, 1])
```

```
Iteration:  1992                Error:  14.2825674998594l3      scale factor:  0.003
Iteration:  1993                Error:  14.282504964108094      scale factor:  0.003
Iteration:  1994                Error:  14.282442428186702      scale factor:  0.003
Iteration:  1995                Error:  14.282379892154779      scale factor:  0.003
Iteration:  1996                Error:  14.28231735607187       scale factor:  0.003
Iteration:  1997                Error:  14.28225481999882       scale factor:  0.003
Iteration:  1998                Error:  14.282192283995654      scale factor:  0.003
Iteration:  1999                Error:  14.282129748123575      scale factor:  0.003
Iteration:  2000                Error:  14.282067212443684      scale factor:  0.003
Maximum number of iterations reached
```

```
plt.plot(x2, y2, 'green', label='Train data')
plt.scatter(x2, output2, label='NN results')
plt.grid(True)
plt.title('Results', c='white')
plt.legend()
plt.show()
```



Задание 3

Задание 3

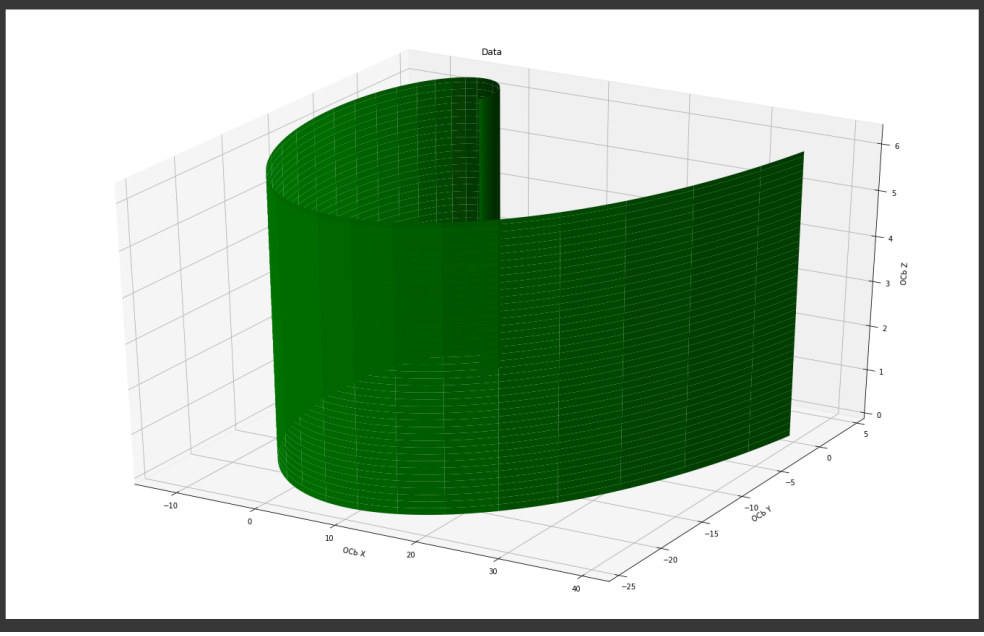
Применить автоассоциативную сеть с узким горлом для аппроксимации пространственной кривой, выделяя старшие нелинейные главные компоненты данных.

Задан обучающий набор, точки набора лежат на пространственной кривой.

Модифицировать обучающее множество из задания 2, добавив в каждой точке третью координату по формуле z = ϕ

```
phi = np.linspace(0, 2 * np.pi, int(2 * np.pi / 0.025), endpoint=True)
r = phi * phi
x3 = r * np.cos(phi)
y3 = r * np.sin(phi)
z = phi


fig = plt.figure(num=1, figsize=(19, 12), clear=True)
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.plot_surface(x3, y3, z.reshape(-1, 1), color='green')
ax.set_title('Data')
ax.set_xlabel('Ось X')
ax.set_ylabel('Ось Y')
ax.set_zlabel('Ось Z')
fig.tight_layout()
```



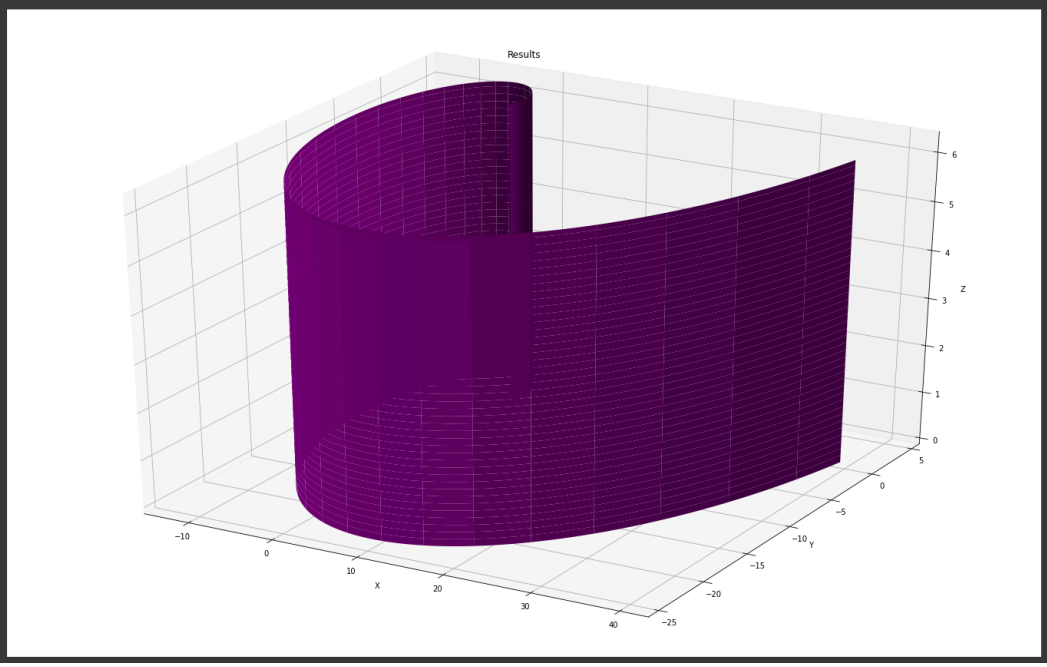
Построим автоассоциативную сеть с узким горлом, реализующую нелинейный метод главных компонент.

Число нейронов скрытого слоя равно [10,2,10]. Задаем метод Левенберга-Марквардта в качестве алгоритма обучения.

```
[20] nn3 = pyrenn.CreateNN([2, 10, 2,  10, 1])
     nn3 = pyrenn.train_LM(np.array([x3, z]), y3, nn3, E_stop=1e-5, k_max=500)
     # Рассчитать выход сети для обучающего множества.
     output3 = pyrenn.NNOut(np.array([x3, z]), nn3)
```

```
Termination Error reached
```

```
fig = plt.figure(num=1, figsize=(19, 12), clear=True)
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.plot_surface(x3, output3, z.reshape(-1, 1), color='purple', label='Output NN')
ax.set_title('Results')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
fig.tight_layout()
```



Код программы.

```
# -*- coding: utf-8 -*-

import pyrenn
from matplotlib import pyplot as plt
import math
import numpy as np
import neurolab as nl
import seaborn as sns
import random
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split

from mpl_toolkits.mplot3d import Axes3D


def f(t):
    return 0.7 * np.cos(t - np.pi / 6)

def g(t):
    return 0.2 * np.sin(t) - 0.1
```

```
t = np.linspace(0, 2 * np.pi, int(2 * np.pi / 0.025), endpoint=True)

x = f(t)
y = g(t)

plt.plot(x, y, 'green')
plt.grid(True)
plt.title('input data', c='white')
plt.show()

"""Создать линейную многослойную сеть прямого распространения

Число нейронов  скрытого слоя задать равным 1.
"""

# Система с 1 входом и выходом
nn = pyrenn.CreateNN([1, 1, 1])

"""Используем метод Левенберга-Марквардта в качестве алгоритма обучения.

Параметры обучения: число эпох обучения 100, предельное значение критерия
обучения $10^{-5}$.
"""

nn = pyrenn.train_LM(x, y, nn, E_stop=1e-5, k_max=100)

"""Рассчитать выход сети для обучающего множества"""

output = pyrenn.NNOut(x, nn)

"""Метод главных компонент — один из основных способов уменьшить размерность
данных, потеряв наименьшее количество информации.

Используем метод главных компонент, выделяя первую главную компоненту данных.
"""

pca = PCA(n_components=1)
data = np.array([[i, j] for i, j in zip(x, y)])
#Fit the model with X and apply the dimensionality reduction on X.
data = pca.fit_transform(data)
#Transform data back to its original space.
data = pca.inverse_transform(data)

"""С помощью сети восстановили набор данных, учитывая информацию только о первой
главной компоненте."""

fig = plt.figure(figsize=(8,10))
ax0 = fig.add_subplot(211)
ax1 = fig.add_subplot(212)

ax0.set_title('Output', c='white')
ax0.plot(output, color='b', lw=2, label='NN Output')
ax0.plot(y, color='g', linestyle=':', lw=2, label='Train output')
ax0.legend()
ax0.grid()

ax1.set_title('Train data', c='white')
ax1.plot(x, output, color='b', lw=3, label='results NN')
ax1.plot(x, y, color='g', linestyle=':', label='Train Data')
ax1.plot(data[:, 0], data[:, 1], color='r', lw=2, label='1st main component')
ax1.legend()
ax1.grid()

fig.tight_layout()
plt.show()
```

```
"""#### Задание 2

Использовать автоассоциативную сеть с узким горлом для аппроксимации кривой на
плоскости, выделяя первую нелинейную главную компоненту данных.
"""

phi = np.linspace(0, 2 * np.pi, int(2 * np.pi / 0.025), endpoint=True)
r = phi * phi
x2 = r * np.cos(phi)
y2 = r * np.sin(phi)

plt.plot(x2, y2, 'green')
plt.grid(True)
plt.title('Input data', c='white')
plt.show()

"""Число нейронов скрытого слоя задать равным [10,1,10]. Используем метод
Левенберга-Марквардта в качестве алгоритма обучения."""

nn2 = pyrenn.CreateNN([1, 10, 1, 10, 1])

nn2 = pyrenn.train_LM(x2, y2, nn2, verbose=True, E_stop=1e-5, k_max=2000)
output2 = pyrenn.NNOut(x2, nn2)

plt.plot(x2, y2, 'green', label='Train data')
plt.scatter(x2, output2, label='NN results')
plt.grid(True)
plt.title('Results', c='white')
plt.legend()
plt.show()

"""#### Задание 3

Применить автоассоциативную сеть с узким горлом для аппроксимации
пространственной кривой, выделяя старшие нелинейные главные компоненты данных.

Задан обучающий набор, точки набора лежат на пространственной кривой.

Модифицировать обучающее множество из задания 2, добавив в каждой точке третью
координату по формуле z = $\phi$
"""

phi = np.linspace(0, 2 * np.pi, int(2 * np.pi / 0.025), endpoint=True)
r = phi * phi
x3 = r * np.cos(phi)
y3 = r * np.sin(phi)
z = phi


fig = plt.figure(num=1, figsize=(19, 12), clear=True)
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.plot_surface(x3, y3, z.reshape(-1, 1), color='green')
ax.set_title('Data')
ax.set_xlabel('ОСЬ X')
ax.set_ylabel('ОСЬ Y')
ax.set_zlabel('ОСЬ Z')
fig.tight_layout()

"""Построим автоассоциативную сеть с узким горлом, реализующую нелинейный метод
главных компонент.

Число  нейронов  скрытого  слоя равно [10,2,10]. Задаем метод Левенберга-
Марквардта в качестве алгоритма обучения.
"""

nn3 = pyrenn.CreateNN([2, 10, 2,  10, 1])
```

```
nn3 = pyrenn.train_LM(np.array([x3, z]), y3, nn3, E_stop=1e-5, k_max=500)
# Рассчитать выход сети для обучающего множества.
output3 = pyrenn.NNOut(np.array([x3, z]), nn3)

fig = plt.figure(num=1, figsize=(19, 12), clear=True)
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.plot_surface(x3, output3, z.reshape(-1, 1), color='purple', label='Output
NN')
ax.set_title('Results')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
fig.tight_layout()
```

Выводы:

Выполнив лабораторную работу, я научилась применять автоассоциативную сеть с узким горлом для аппроксимации функций и отображения данных, выделяя линейные и нелинейные компоненты данных.

Итеративную автоассоциативную сеть с узким горлом используют, когда необходимо на ходу адаптироваться к меняющемуся потоку данных. Также нейроалгоритмы легко обобщаются на случай нелинейного сжатия информации, когда никаких явных решений уже не существует. Можно заменить линейные нейроны нелинейными. С минимальными видоизменениями нейроалгоритмы будут работать и в этом случае, всегда находя оптимальное сжатие при наложенных нами ограничениях.