МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №7
по курсу «Нейроинформатика»

Автоассоциативные сети с узким горлом.

Выполнил: Д. Д. Син
Группа: 8О-407Б
Преподаватели: Н.П Аносова

Москва, 2021

Постановка задачи

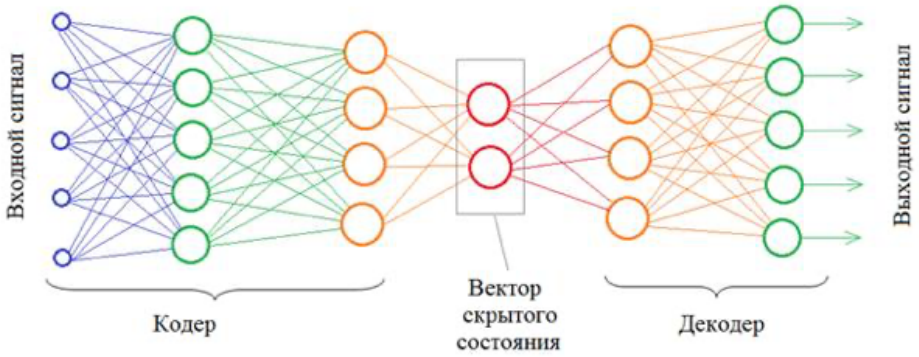
Целью работы является исследование свойств автоассоциативных сетей с узким горлом, алгоритмов обучения, а также применение сетей для выполнения линейного и нелинейного анализа главных компонент набора данных.

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 плос- кости, выделяя первую нелинейную главную компоненту данных.
3. Применить автоассоциативную сеть с узким горлом для аппроксимации пространственной кривой, выделяя старшие нелинейные главные компоненты данных.

Метод решения

Для решения задачи воспользуемся библиотекой pyrenn. Будем использовать автоассоциативную сеть или по другому автоэнкодеры. Такие сети помогают находить главную компоненту в данных. Будем строить автоасоциативную сеть и находить главную компоненту, а также распознавать данные.



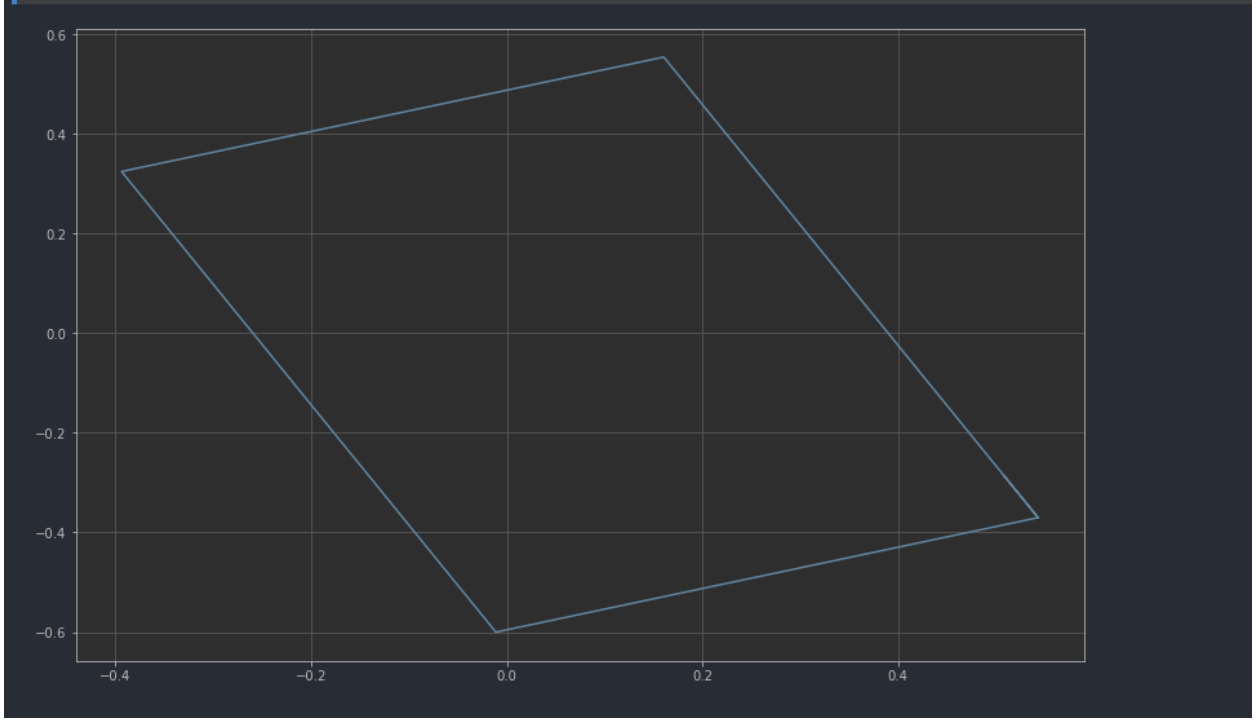
Описание работы программы

Задание 1.

Точки прямоугольника

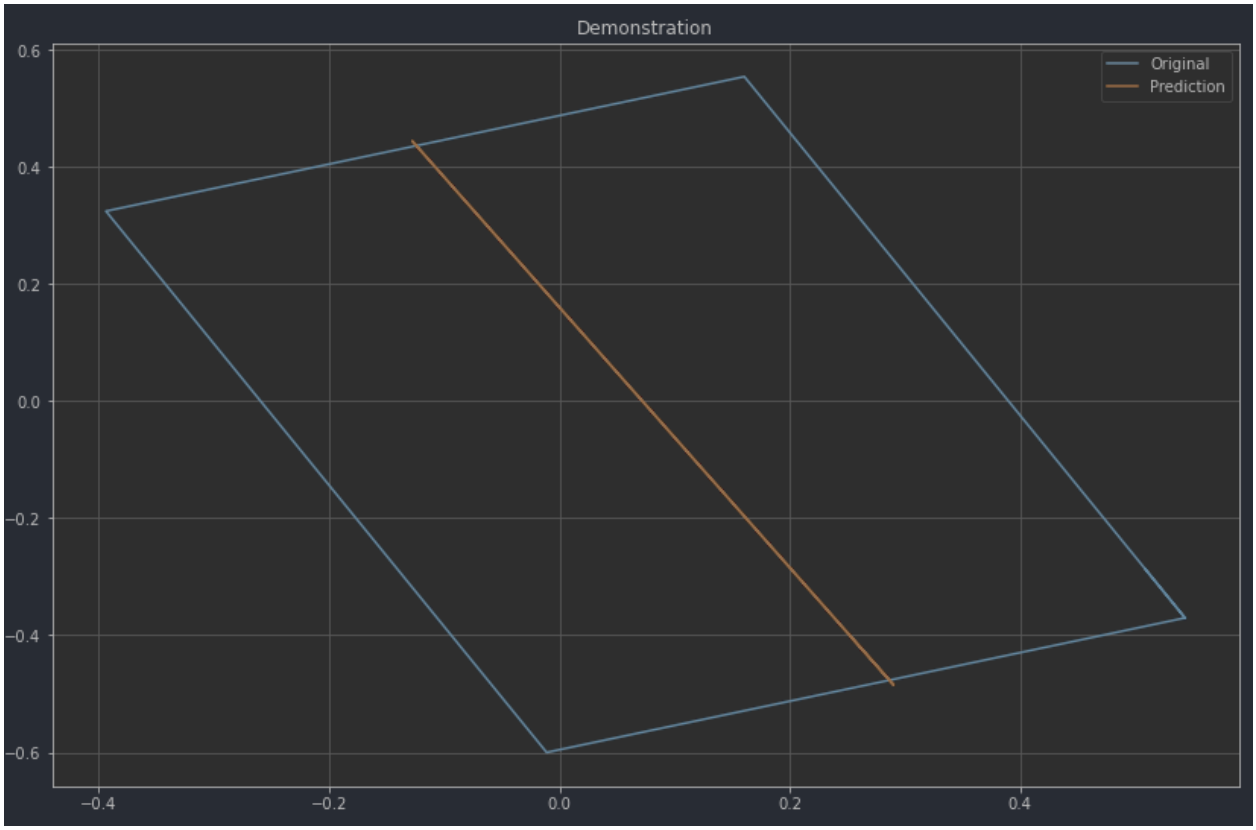
```
def f(t):
    return 0.3 * (np.cos(np.pi/2 * np.floor(t)) - (2 * t - 2 * np.floor(t) - 1) * np.sin(np.pi/2 * np.floor(t)) + 0.2)

def g(t):
    return 0.5 * (np.sin(np.pi/2 * np.floor(t)) + (2 * t - 2 * np.floor(t) - 1) * np.cos(np.pi/2 * np.floor(t)) - 0.1)
x = np.array([f(t) * np.cos(np.pi/8) - g(t)*np.sin(np.pi/8) for t in np.arange(0, 4.1, 0.01)])
y = np.array([f(t) * np.sin(np.pi/8) + g(t)*np.cos(np.pi/8) for t in np.arange(0, 4.1, 0.01)])
plt.figure(figsize=(14, 9))
plt.plot(x, y)
plt.grid(True)
plt.show()
```



Обучение сети, нахождение главной компоненты

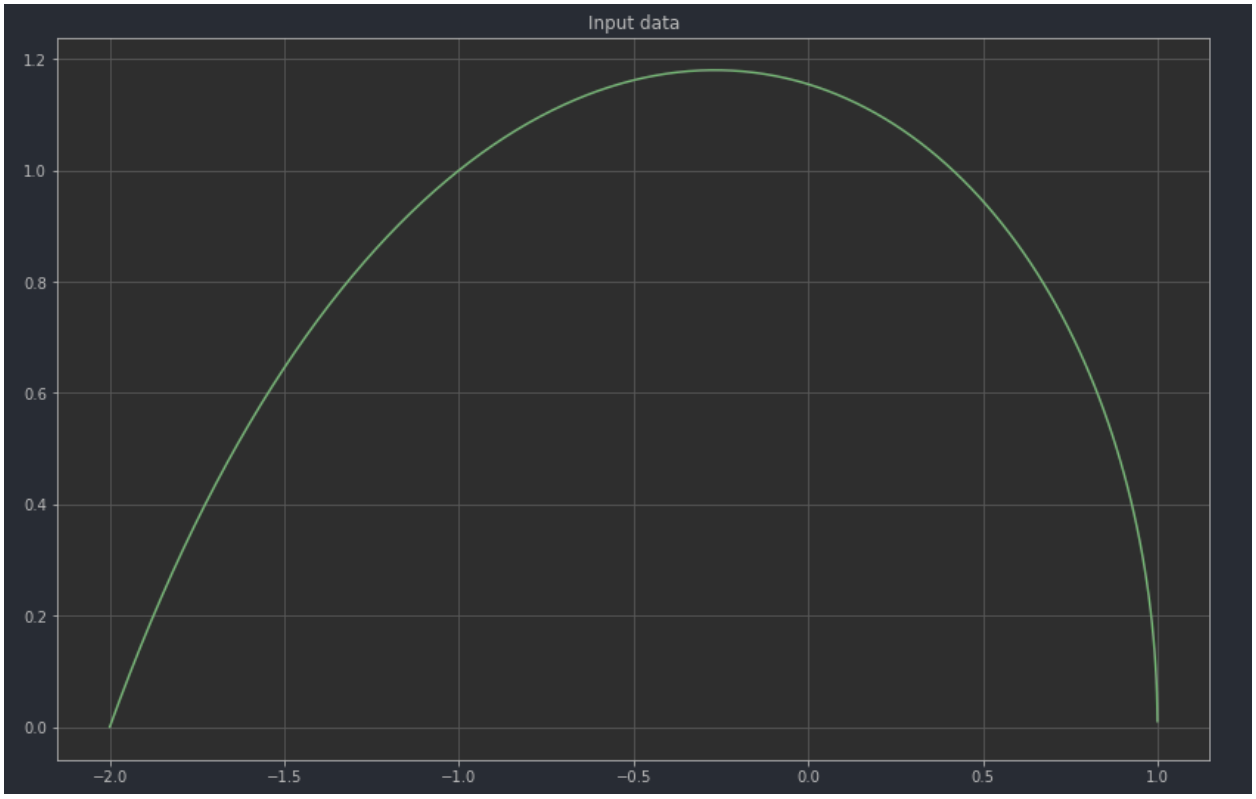
```
def get_curve(phi):
    r = 1 / np.cos(phi/3)
    return r*np.cos(phi), r*np.sin(phi)
phi = np.linspace(0.01, np.pi, int(np.pi / 0.025))
x, y = get_curve(phi)
plt.figure(figsize=(14, 9))
plt.plot(x, y, 'green')
plt.grid(True)
plt.title('Input data')
plt.show()
```



Задание 2.

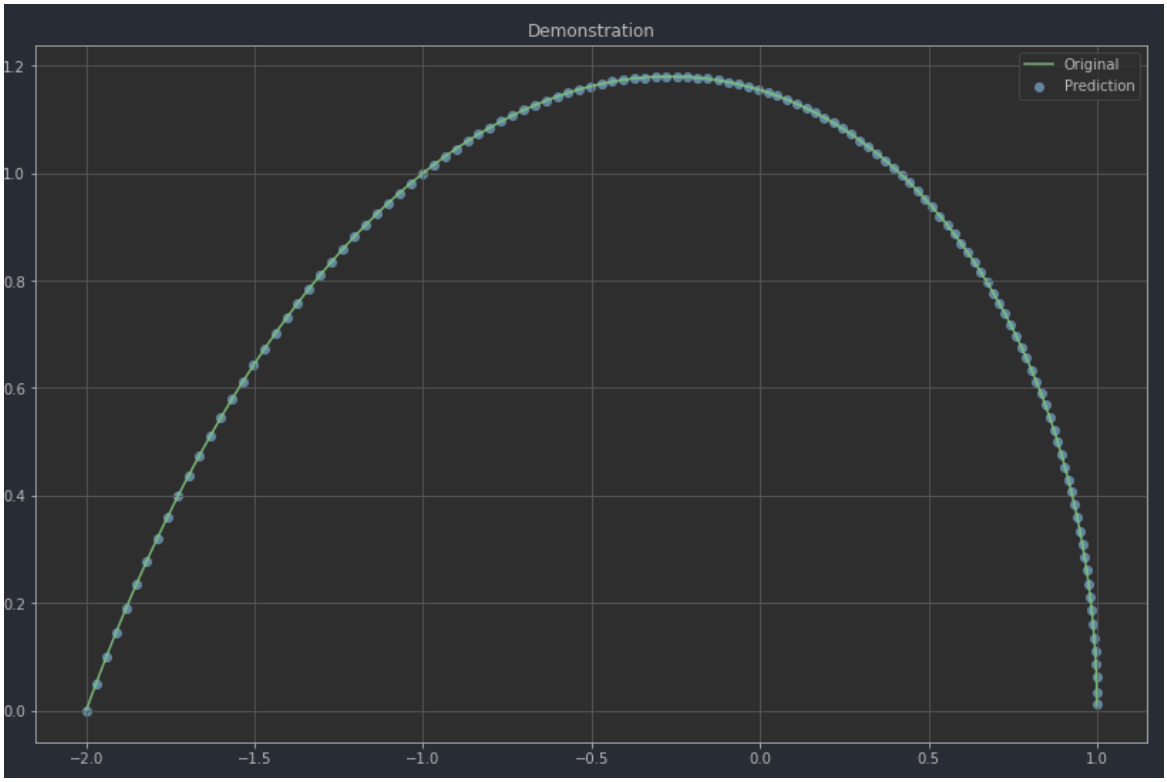
Использовать автоассоциативную сеть с узким горлом для аппроксимации кривой на плоскости, выделяя первую нелинейную главную компоненту данных.

```
def get_curve(phi):
    r = 1 / np.cos(phi/3)
    return r*np.cos(phi), r*np.sin(phi)
phi = np.linspace(0.01, np.pi, int(np.pi / 0.025))
x, y = get_curve(phi)
plt.figure(figsize=(14, 9))
plt.plot(x, y, 'green')
plt.grid(True)
plt.title('Input data')
plt.show()
```



Аппроксимация кривой

```
nn = pyrenn.CreateNN([1, 10, 1, 10, 1])
nn2 = pyrenn.train_LM(x, y, nn, E_stop=1e-5, k_max=2000)
a = pyrenn.NNOut(x, nn)
plt.figure(figsize=(14, 9))
plt.plot(x, y, 'green', label='Original')
plt.scatter(x, a, label='Prediction')
plt.grid(True)
plt.title('Demonstration')
plt.legend()
plt.show()
```



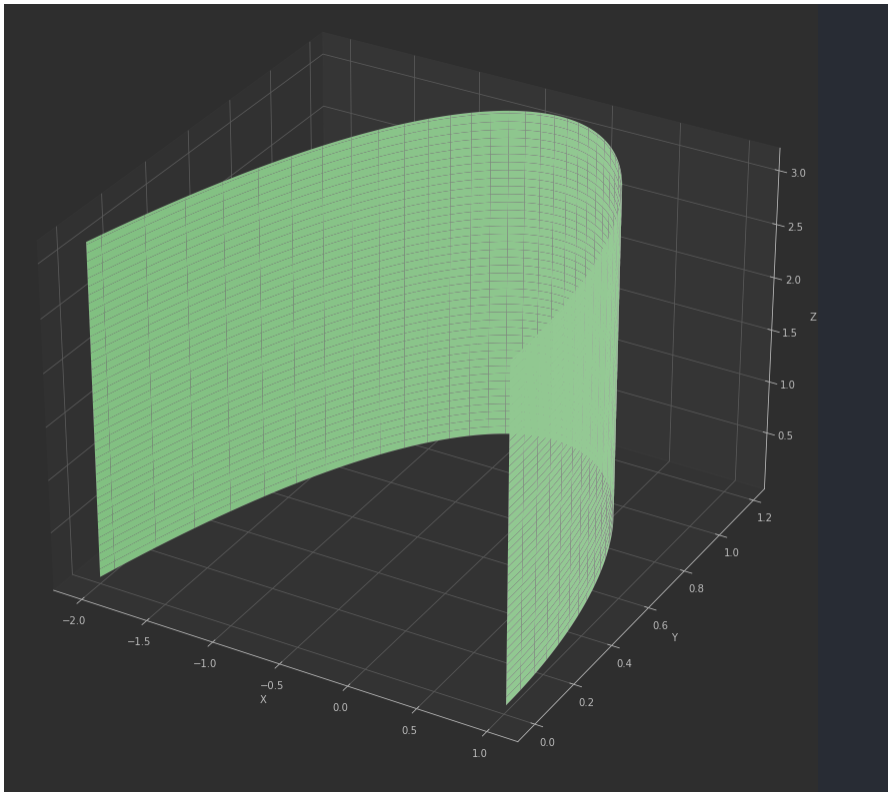
Задание 3

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

Пространственная аппроксимации

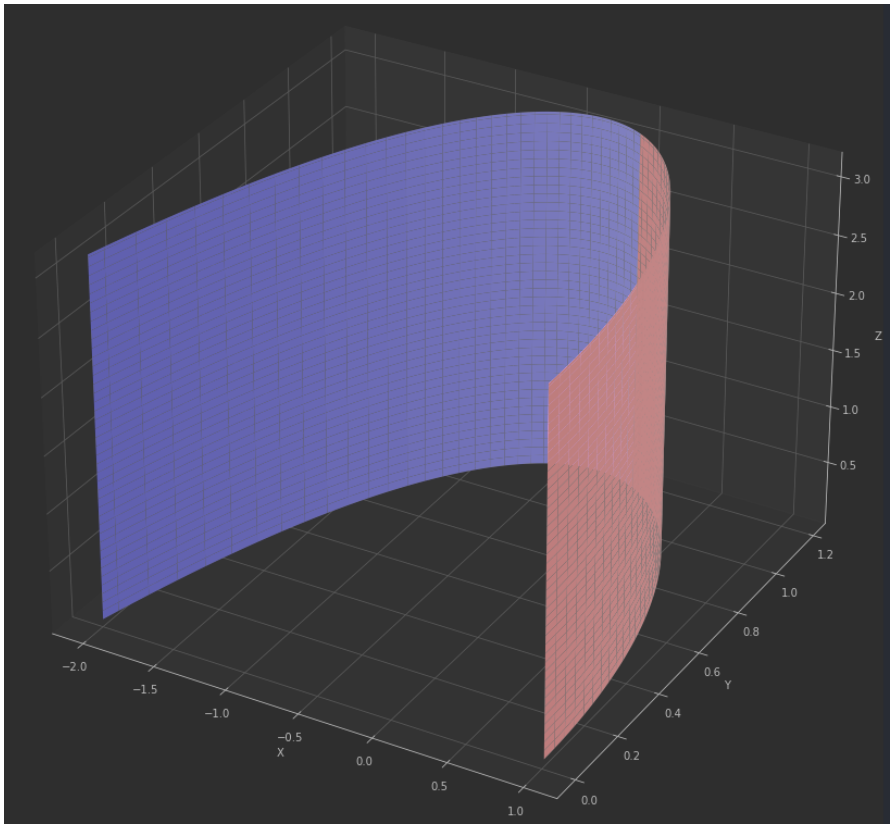
```
phi = np                    int(np.pi / 0.025))
x, y = get_curve(phi)
fig = plt.figure(num=1, figsize=(19, 12), clear=True)
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.plot_surface(x, y, phi.reshape(-1, 1), color='green')
ax.set_title('Input Data')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
fig.tight_layout()
```

Multiple implementations



Аппроксимация кривой

```python
nn = pyrenn.CreateNN([2, 10, 2,  10, 1])
nn = pyrenn.train_LM(np.array([x, phi]), y, nn, E_stop=1e-5, k_max=500)
a = pyrenn.NNOut(np.array([x, phi]), nn)
fig = plt.figure(num=1, figsize=(19, 12), clear=True)
ax = fig.add_subplot(1, 1, 1, projection='3d')
ax.plot_surface(x[:x.size//2], y[:y.size//2], phi.reshape(-1, 1), color='red')
ax.plot_surface(x[x.size//2-1:], a[y.size//2-1:], phi.reshape(-1, 1), color='blue')
ax.set_title('Demonstration')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
fig.tight_layout()
```



Выводы

В данной лабораторной работе проанализировали и применили автоассоциативные сети для нахождения главной компоненты и для аппроксимации кривых. Данные сети хорошо справляются с нахождением главных компонент, а также с аппроксимацией. Очевидно, раз можно находить главную компоненту, то можно снижать размерность данных, а также находить зависимости в данных при их анализе.